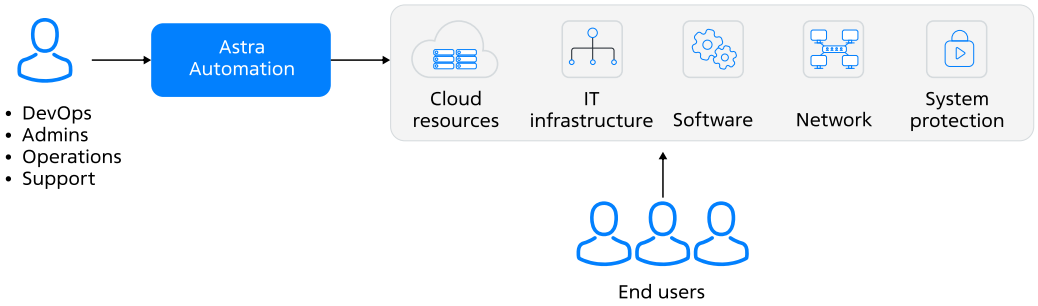
Astra Automation 2.0

27.11.2025 г.

Оглавление

Платформа для автоматизации множества рутинных операций по управлению информационной инфраструктурой заказчика, позволяющая существенно сократить затраты на ее поддержку и администрирование.

Решения базируются на использовании известного открытого проекта Ansible со свободно распространяемым исходным кодом.



Реализуя принцип инфраструктурного кода *IaC*, платформа позволяет автоматизировать решение следующих задач:

- Развертывание IT-инфраструктуры на базе различных операционных систем, используя как физическое оборудование, так и виртуальное окружение в необходимых сочетаниях.
- Управление объектами облачной инфраструктуры, включая их создание, настройку и обновление.
- Установка и настройка программного обеспечения, включая одиночные приложения и комплексы.
- Управление сетевым оборудованием от различных производителей.
- Настройка системы безопасности для приведения ее в соответствие с требованиями регуляторов.

Преимущества

Преимущества использования Astra Automation по отношению к последовательным ручным способам развертывания сложных информационных систем:

- существенное упрощение и ускорение процесса развертывания, обеспечивающие кратное снижение трудозатрат и времени на весь процесс;
- обеспечение безопасности в соответствии с существующими требованиями;
- обеспечение безошибочной непротиворечивой конфигурации устанавливаемых продуктов, значительно снижающее риски возникновения инцидентов и связанных с ними простоев систем;
- развертывание информационной инфраструктуры любой сложности.

Пользователи

Данная платформа ориентирована на крупные компании и интеграторов сложных систем и обеспечивает необходимые потребительские качества.

С применением данной платформы *крупные компании* решают следующие важные проблемы:

- снижение стоимости владения информационной инфраструктурой;
- высвобождение высококвалифицированных инженеров для решения внутренних сложных задач;
- ускорение развертывания инфраструктуры и снижение рисков.

Платформа удовлетворяет следующим насущным потребностям *интеграторов*:

- обеспечение безопасного развертывания продуктов и отсутствия ошибок в их настройках;

- уменьшение затрат на весь процесс развертывания;
- сокращение времени развертывания комплексной инфраструктуры.

Знакомство с платформой

Узнайте для чего это, как устроено и как работает.

Документация содержит начальные и углубленные сведения о структуре платформы, включая ее основные компоненты, их взаимодействие между собой и с управляемой инфраструктурой.

Возможности платформы Назначение платформы в целом и основные области применения.

Обзор архитектуры Структура платформы и общие сведения о ее составных частях.

Интерфейс платформы Организация доступа к платформе через веб-интерфейс и API.

Глобальная библиотека контента Структура и концепции облачного реестра инфраструктурного кода. Способы получения содержимого из этого реестра.

Управление контентом Организация приватного реестра инфраструктурного кода в составе вашей платформы Astra Automation.

Автоматизация процессов Назначение, структура и настройка Automation Controller для автоматизации процессов в инфраструктуре IT.

Обработка событий Способы и средства автоматического реагирования на изменения в инфраструктуре IT.

Обновления Astra Automation Жизненный цикл продукта и документации, объявления о выпусках новых версий.

Развертывание и техническое обслуживание

Как развернуть и поддерживать Astra Automation.

Вы познакомитесь с полным жизненным циклом платформы и ее поддержкой, начиная с планирования и подготовки развертывания платформы, а также с ее последующим техническим обслуживанием. Здесь все расписано по условным дням: day-0, day-1 и day-2.

Быстрый старт Быстрое развертывание тестового варианта Astra Automation и начало знакомства с платформой путем выполнения простейших операций администратора.

Day 0: Планирование и подготовка Приняв решение о внедрении Astra Automation, начните с тщательного планирования и подготовки ресурсов. Здесь про выбор модели развертывания (на виртуальных машинах или в Kubernetes) и топологии (базовая или уровня предприятия), а также про подготовку аппаратных и программных средств.

Day 1: Развертывание и тестирование Все готово – значит можно начинать. Пошаговые инструкции приведут вас к созданию запланированной инфраструктуры.

Day 2: Операционные задачи А это уже повседневная жизнь, начиная с условного day-2 и далее. Необходимо следить за состоянием платформы, обеспечивать работоспособность ее компонентов и безопасность эксплуатации. Эффективными способами являются периодическое резервное копирование и своевременное обновление версии Astra Automation.

Специальные главы

Обеспечение безопасности, производительности и примеры решений.

Здесь вы найдете ответы на типовые вопросы по развертыванию и эксплуатации Astra Automation:

- как платформа обеспечивает безопасность управления и что мы рекомендуем для повышения безопасности;
- как рассчитать ориентировочную нагрузку на узлы платформы и ресурсы, необходимые для поддержания работоспособности;

- примеры применения платформы.

Обеспечение безопасности Компиляция описания встроенных методов и средств обеспечения безопасности, используемых на разных этапах жизненного цикла платформы. Рекомендации по использованию дополнительных возможностей по усилению защищенности.

Настройка производительности Оценка нагрузки на различные компоненты платформы применительно к вашим задачам. Ориентировочный расчет количества узлов и их ресурсов для обеспечения работоспособности платформы под такой нагрузкой.

Примеры решений Применение Astra Automation для решения типовых задач автоматизации, включая настройку механизмов защиты в соответствии с рекомендациями отечественных регуляторов и мировых стандартов.

Дополнительные материалы Вспомогательные информационные ресурсы, позволяющие понять рабочее окружение, используемые инструменты и применение Astra Automation для развертывания некоторых продуктов ПАО Группа Астра.

Разработчикам в помощь

Как это усилить и приспособить.

Ansible Core Без разработки собственного кода сложно обойтись. В основе его лежит Ansible. Знание его основ потребуется на разных стадиях разработки, начиная с создания простейших сценариев и до разработки собственных коллекций, модулей и расширений Ansible.

Ресурсы разработчика Назначение, состав и применение средств разработки собственного инфраструктурного кода.

Справочники

То, что всегда пригодится профессионалу.

Документация содержит справочную информацию, необходимую для профессиональной работы с платформой.

Глобальная библиотека контента Содержимое репозитория Automation Hub.

Параметры развертывания платформы Параметры утилиты развертывания `aa-setup` и описания инвентаря платформы.

Справочник по EDA Список источников событий и действий, встроенных в Event-Driven Automation.

Справочник по CDK Справочник разработчика, включая параметры утилит CDK (Content Development Kit) и состав образов среды разработки.

Программный интерфейс (API)

Необходимо для управления платформой программным путем.

Единой точкой входа для запросов API является шлюз платформы. При этом каждый компонент предоставляет собственную спецификацию в формате OpenAPI.

Platform Gateway API Управление организационной инфраструктурой в рамках RBAC (Role-Based Access Control) и общие настройки платформы.

Private Automation Hub API Управление контентом Ansible и образами среды исполнения в приватном реестре.

Automation Controller API Управление процессами автоматизации, включая подготовку проектов, описания инвентаря, управление шаблонами, планирование и запуск заданий и другое.

Event-Driven Automation API Создание процессов обработки событий и управление ими.

Типографский стиль

Для улучшения читаемости в документации используются различные стили для выделения частей, которые по смыслу отличаются от основного текста.

Жирный (Bold) – слова, к которым требуется привлечь внимание, а также названия элементов пользовательского интерфейса (за исключением ссылок и кнопок).

Наклонный (Italic) – термины при первом употреблении и значения, которые необходимо указать в полях формы.

`Моноширинный (Monospace)` – названия параметров, названия ядер и их версии, названия пакетов, названия и значения переменных, названия утилит, имена файлов и каталогов.

`Enter` – названия клавиш и их сочетаний.

File ▸ *Exit* – названия пунктов меню и последовательность их выбора.

Запустить – названия кнопок и ссылок.

Совет

Полезная информация, которая может упростить работу.

Примечание

Описание какой-либо особенности работы с платформой.

Важно

Важная информация, которую необходимо учитывать при использовании платформы.

Предупреждение

Важная информация, игнорирование которой может привести к проблемам в работе платформы, утечке или потере данных и другим неприятным последствиям.

Последнее обновление: 27.11.2025

ГЛАВА 1

Быстрый старт

1.1 Быстрое развертывание платформы

Быстрое развертывание платформы Astra Automation предназначено для получения оперативного доступа к системе и ознакомления с ее базовыми возможностями. Для этого в качестве рабочей среды предлагаем использовать Kubernetes Minikube[1].

Предупреждение

Использование Minikube для развертывания Astra Automation допустимо только в ознакомительных целях и не предназначено для эксплуатации в промышленных рабочих окружениях.

1.1.1 Предварительные требования

Для развертывания платформы потребуются дополнительные аппаратные и программные ресурсы.

Программное обеспечение платформы

Для использования платформы необходима подписка. Чтобы получить пробную подписку, перейдите на сайт Astra Automation[2] и нажмите кнопку *Запросить пробную версию платформы бесплатно*.

Развертывание выполняется с помощью архивированного пакета, доступного в Личном кабинете[3].

Рабочее окружение

Для развертывания платформы понадобится узел в виде ВМ (виртуальная машина) или физического компьютера, обладающий характеристиками не ниже указанных:

[1] https://minikube.sigs.k8s.io/docs/start/
[2] https://astra-automation.ru/
[3] https://lk.astra.ru/

Характеристика	Значение
Количество ядер CPU	8
Объем RAM, ГБ	16
Объем хранилища, ГБ	30

Вы можете использовать любую операционную систему, в которой можно установить Minikube. Для данной демонстрации использовалась ВМ с установленной операционной системой Astra Linux Special Edition 1.8.2.UU1 в базовом режиме защищенности («Орел»). Для настройки узла необходим доступ к нему по протоколу SSH.

1.1.2 Подготовка узла

Подготовьте узел следующим образом:

1. Подключитесь к узлу по SSH.
2. Установите и настройте систему Docker:

```
sudo apt update && sudo apt install -y make git curl docker.io docker-compose␣
↪docker-buildx
```

```
sudo usermod -aG docker $USER && newgrp docker
```

3. Настройте уровень безопасности для загрузки контейнерных образов в Minikube:

Предупреждение

Решение подходит только для отладочного режима и не предназначено для эксплуатации в промышленной среде.

- Используя привилегии суперпользователя (команда sudo), создайте каталог /etc/docker/ и в нем файл daemon.json со следующим содержимым:

```
{
   "astra-sec-level" : 6
}
```

- Повторно запустите службу docker:

```
sudo systemctl restart docker
```

4. Установите систему оркестрации контейнеров Minikube:

```
curl -Lo minikube https://storage.googleapis.com/minikube/releases/latest/minikube-
↪linux-amd64
chmod +x minikube
sudo install minikube /usr/local/bin/
```

5. Установите утилиту управления kubectl:

```
curl -LO "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.
↪txt)/bin/linux/amd64/kubectl"
chmod +x kubectl
sudo mv kubectl /usr/local/bin/
```

1.1.3 Подготовка Kubernetes

Переведите рабочую среду в активное состояние:

1. Убедитесь, что вы подключены к целевому узлу по SSH.

2. Запустите сервис Minikube и установите контроллер `ingress` для обеспечения доступа к интерфейсу платформы:

```
minikube start --cpus=6 --memory=8192 --disk-size=30g --addons=ingress
```

3. Проверьте состояние Minikube:

```
minikube status
```

Команда выводит на экран терминала следующие данные:

```
minikube
type: Control Plane
host: Running
kubelet: Running
apiserver: Running
kubeconfig: Configured
```

1.1.4 Развертывание платформы

Выполните следующие шаги для развертывания платформы:

1. Распакуйте архив платформы в рабочем каталоге:

```
mkdir astra-automation-setup
```

```
tar -xvzf <offline_bundle.tgz> -C astra-automation-setup/ && cd astra-automation-
↪setup
```

Структура пакета имеет следующий вид:

```
├── examples/
│   ├── kubernetes/
│   └── minikube/
└── operators/
    ├── aa-operator-deploy.yaml
    ├── ac-operator-deploy.yaml
    ├── eda-operator-deploy.yaml
    └── pah-operator-deploy.yaml
```

2. Установите операторы платформы:

```
kubectl apply -f operators/
```

Создаются контейнеры и сервисы Kubernetes в пространстве `astra-automation`. Это потребует несколько минут. Для проверки состояния подов используйте следующую команду:

```
kubectl -n astra-automation get pods -w
```

Поды должны быть в состоянии `running`.

3. Запустите развертывание платформы в минимальном составе:

```
kubectl apply -n astra-automation -f examples/minikube/minimal/minikube-minimal-aa-
↪demo.yaml
```

На развертывание потребуется несколько минут. Дождитесь готовности подов платформы:

```
kubectl -n astra-automation get pods -w
```

Примечание

Если по какой-либо причине надо повторить все сначала, удалите все объекты, созданные в Minikube:

```
minikube stop && minikube delete --all --purge
```

1.1.5 Получение доступа к интерфейсу платформы

Для доступа к единому интерфейсу платформы процедура развертывания создает учетную запись admin, имеющую случайным образом сгенерированный пароль. Для получения пароля воспользуйтесь следующей командой:

```
kubectl -n astra-automation get secret aa-demo-admin-password \
        -o jsonpath="{.data.password}" | base64 -d
```

Сервис (контроллер Kubernetes) ingress доступен в пространстве ingress-nginx. Получите сведения о нем:

```
minikube service ingress-nginx-controller -n ingress-nginx
```

Таблица в выходных данных показывает соответствие между URL, по которому доступен сервис, и внутренним портом пользовательского интерфейса:

```
|-------------|-------------------------|-------------|-----------------------|
|  NAMESPACE  |          NAME           | TARGET PORT |          URL          |
|-------------|-------------------------|-------------|-----------------------|
| ingress-nginx | ingress-nginx-controller | http/80     | http://192.168.49.2:32390 |
|             |                         | https/443   | http://192.168.49.2:31668 |
|-------------|-------------------------|-------------|-----------------------|
[ingress-nginx ingress-nginx-controller http/80
https/443 http://192.168.49.2:32390
http://192.168.49.2:31668]
```

В данном примере установлены следующие перенаправления:

- http://192.168.49.2:32390 -> http://<gateway>
- https://192.168.49.2:31668 -> https://<gateway>

192.168.49.2 является адресом узла Minikube. Если вы устанавливали все на локальной машине, то можете использовать эти URL в вашем браузере. Если это внешняя ВМ, то проверьте доступность интерфейса с помощью утилиты curl, выполнив следующие команды внутри ВМ:

```
curl http://192.168.49.2:32390
```

или

```
curl -k http://192.168.49.2:31668
```

Для доступа к интерфейсу платформы, развернутой на внешней ВМ, из локальной машины, на которой вы работаете, необходимо туннельное соединение. Создайте его с помощью одной из следующих команд на вашей локальной машине:

- Для доступа по HTTP:

  ```
  ssh -L 8080:192.168.49.2:32390 admin@158.160.201.18
  ```

  В приведенной команде 158.160.201.18 – IP-адрес ВМ. В браузере интерфейс будет доступен по URL http://localhost:8080.

- Для доступа по HTTPS:

```
ssh -L 8043:192.168.49.2:31668 admin@158.160.201.18
```

В браузере интерфейс будет доступен по URL https://localhost:8043. При этом надо подтвердить, что вы доверяете этому ресурсу, у которого не установлен подтвержденный сертификат TLS.

1.1.6 Первоначальная настройка платформы

Первоначальная настройка состоит из следующих шагов:

1. При первом подключении браузер выводит предупреждение о том, что подключение не защищено. Это нормальное поведение, поскольку для защиты подключения контроллер использует самоподписанный сертификат. При использовании браузера на основе Chromium выполните следующие действия:
   a. Нажмите кнопку *Дополнительные*.
   b. Нажмите ссылку *Перейти на сайт <IP> (небезопасно)*.
2. Для аутентификации в интерфейсе платформы введите название учетной записи `admin` и пароль, определенный ранее.
3. Активируйте лицензию на продукт, следуя *инструкции*.

1.1.7 Заключение

В этом сценарии вы познакомились с основными шагами по развертыванию платформы и его использованию для настройки управляемых узлов. Из всей последовательности шагов важно выделить следующие действия:

- Обеспечение предварительных требований.
- Установка операторов в окружении Kubernetes.
- Развертывание платформы с помощью операторов.
- Обеспечение доступа к интерфейсу платформы.
- Активизация лицензии на использование платформы.

1.2 Быстрый старт для администратора

Пройдите полный путь от подготовки окружения и создания проекта до выполнения задания. Процесс автоматизации состоит из следующих этапов:

1. *Описание инвентаря* – выбор и описание целевых узлов (инвентаря), к которым будет применяться автоматизация.
2. *Создание проекта* – создание проекта, как внутренней сущности платформы, которая использует готовый проект автоматизации. Обычно проекты размещаются в репозиториях систем управления исходным кодом (SCM, Source Control Management). Они включают один или несколько наборов сценариев автоматизации (playbooks).
3. *Создание шаблона заданий* – создание шаблона, который определяет параметры для запуска сценариев автоматизации.
4. *Выполнение заданий* – запуск заданий автоматизации с использованием созданного шаблона.

1.2.1 Описание инвентаря

Этот простейший сценарий ориентирован на выполнение задания на локальном узле, который будет единственным элементом инвентаря. Создайте описание инвентаря следующим образом:

1. На панели навигации выберите **Автоматизация процессов –> Инфраструктура –> Инвентарные списки** (Automation Execution –> Infrastructure –> Inventories) .

2. Нажмите кнопку *Создать инвентарный список* (Create inventory) и выберите **Создать инвентарный список** (Create inventory).
3. В форме **Создать инвентарный список** (Create inventory) заполните следующие поля:
   - в поле **Название** (Name) введите строку *Quick Start Inventory*;
   - в поле **Организация** (Organization) выберите организацию Default.
4. Нажмите кнопку *Создать инвентарный список* (Create inventory). Откроется окно просмотра свойств созданного инвентарного списка.
5. Выберите вкладку **Узлы** (Hosts).
6. Нажмите кнопку *Создать узел* (Create host).
7. Заполните форму **Создать узел** (Create host):
   - **Название** (Name): *localhost*.
   - **Переменные** (Variables):

```
---
ansible_host: localhost
ansible_connection: local
```

8. Нажмите кнопку *Создать узел* (Create host).

1.2.2 Создание проекта

Простейший проект автоматизации размещён в системе SCM. С помощью следующих шагов добавьте соответствующий проект, как внутреннюю сущность платформы:

1. На панели навигации выберите **Автоматизация процессов -> Проекты** (Automation Execution –> Projects).
2. Нажмите кнопку *Создать проект* (Create project).
3. Заполните форму **Создать проект** (Create project):
   - **Название** (Name): *Quick Start Project*;
   - **Организация** (Organization): Default;
   - **Тип системы управления исходным кодом** (Source Control Type): Git;
   - **URL системы управления исходным кодом** (Source Control URL):

```
https://gitflic.ru/project/astra_team/astra-automation-quick-start.git
```

4. Нажмите кнопку *Создать проект* (Create project).

Дождитесь завершения синхронизации проекта: убедитесь, что в поле **Статус последнего задания** (Last job status) отображается значение Успех (Success).

1.2.3 Создание шаблона заданий

С помощью следующих шагов создайте шаблон для запуска заданий автоматизации, используя один из файлов сценариев автоматизации, входящих в созданный ранее проект:

1. На панели навигации выберите **Автоматизация процессов -> Шаблоны** (Automation Execution –> Templates).
2. Нажмите кнопку *Создать шаблон* (Create template) и в открывшемся меню выберите *Создать шаблон задания* (Create job template).
3. Заполните форму **Создать шаблон задания** (Create Job Template):
   - **Название** (Name): Quick Start Template.
   - **Тип задания** (Job Type): **Выполнить** (Run).
   - **Инвентарный список** (Inventory): Quick Start Inventory.

- **Проект** (Project): Quick Start Project.
- **Playbook**: `playbooks/quick-start.yml`.

4. Нажмите кнопку *Создать шаблон задания* (Create job template)|. Откроется окно просмотра свойств шаблона.

С помощью шаблона задания вы сможете запускать задания вручную (см. далее) или по расписанию.

1.2.4 Выполнение заданий

Запустите на выполнение задание, используя созданный ранее шаблон:

1. Для запуска задания нажмите кнопку *Запустить шаблон* (Launch template). Откроется окно с журналом выполнения задания.
2. Дождитесь перехода задания в статус Успех (Success), после чего обновите страницу браузера.

В окне вывода будет отображен журнал выполнения задания, например:

```
PLAY [Quick Start Playbook] ****************************************************

TASK [Print Quick Start] *******************************************************
ok: [localhost] => {
      "msg": "Quick Start with Astra Automation on localhost"
}

PLAY RECAP *********************************************************************
localhost                  : ok=1    changed=0    unreachable=0    failed=0    skipped=0␣
↪    rescued=0    ignored=0
```

1.2.5 Заключение

Выполненное руководство позволило вам получить первоначальный опыт по основным административным операциям в системе автоматизации. В процессе выполнения вы освоили ключевые действия:

- Описание инвентаря с добавлением локального узла и определением необходимых переменных подключения.
- Создание проекта, привязанного к внешнему репозиторию, с готовым сценарием автоматизации.
- Формирование шаблона заданий, который связывает инвентарь, проект и требуемый playbook.
- Запуск задания с контролем успешного завершения и анализом журнала выполнения.

Приведенные здесь инструкции предназначены для быстрого освоения базовых приемов работы с платформой:

- развертывание платформы в минимальном составе;
- использование платформы для установки программ;
- разработка простейшего контента Ansible для управления инфраструктурой с помощью платформы.

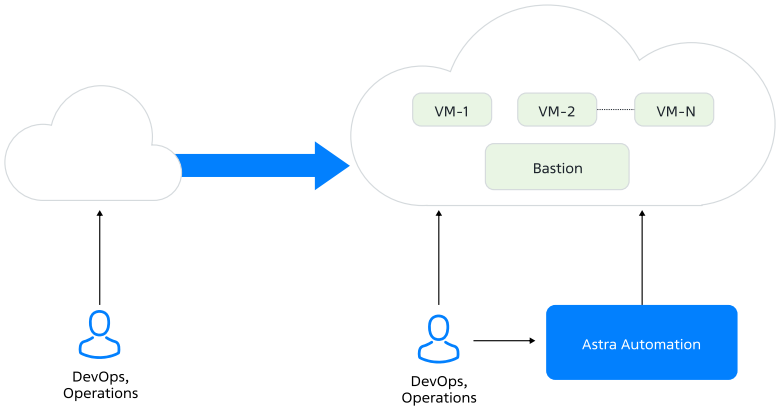
ГЛАВА 2

Возможности платформы

Платформа Astra Automation решает множество задач по управлению IT-инфраструктурой, решение которых описано в следующих секциях. Многие задачи можно решать в произвольной последовательности и в различных сочетаниях. Жизненный цикл инфраструктуры содержит множество повторяющихся операций развертывания, установки, замены и обновления различных ее компонентов.

2.1 Управление облачными ресурсами

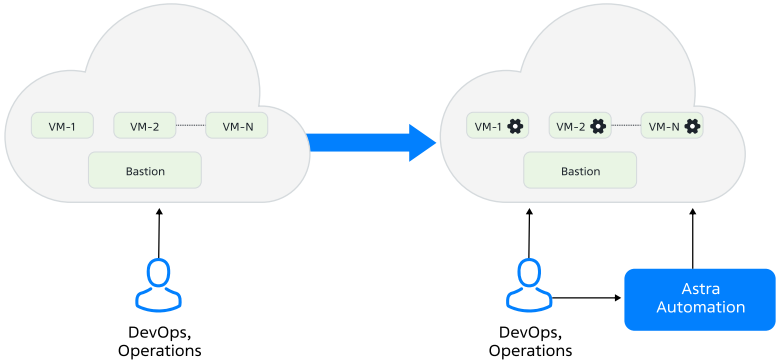
Обладая доступом к облачному пространству, публичному или частному, организация может использовать Astra Automation для создания базовой инфраструктуры, используя технологии виртуализации.



Для создания облачных ресурсов и управления ими предлагается использовать модули Ansible.

2.2 Установка программного обеспечения

Astra Automation предоставляет развитый реестр инфраструктурного кода, позволяющего устанавливать разнообразные одиночные системы или комплексы систем на оборудовании организации или в облачном пространстве.



Сертифицированный инфраструктурный код ориентирован на использование операционной системы Astra Linux Special Edition.

2.3 Управление безопасностью

Astra Automation предоставляет специальную коллекцию Ansible для автоматизации применения требуемых мер безопасности в соответствии с рекомендациями CIS[4], ФСТЭК России[5] и других сообществ и организаций. Коллекция ориентирована на использование операционной системы Astra Linux Special Edition.

2.4 Управление сетевым оборудованием

Сетевое оборудование (маршрутизаторы и коммутаторы), составляющие основу любой IT-инфраструктуры, в зависимости от сложности сети подлежат достаточно частой настройке под нужды этой инфраструктуры. К таким настройкам, например, относятся следующие составляющие:

- таблицы маршрутизации;
- состав виртуальных сетей;
- входные и выходные фильтры.

Astra Automation предоставляет необходимые модули и коллекции для настройки отечественного и зарубежного оборудования.

2.5 Рутинные операции по управлению инфраструктурой

IT-инфраструктура, подобно живому организму, подлежит частым изменениям, требующим каждодневной настройки, связанной с организационными изменениями, изменениями в ролевых моделях доступа к различным сервисам и другими. Подобные операции требуют значительных трудозатрат, которые можно значительно снизить, используя возможности Astra Automation по автоматизации рутинных операций. Для этого необходимо подготовить шаблоны для запуска типовых заданий или потоков заданий, которые может использовать персонал с минимальными знаниями платформы.

2.6 Разработка собственного инфраструктурного кода

Несмотря на значительный объем инфраструктурного кода, предоставляемого платформой, практически в каждой организации найдутся специфичные задачи, для выполнения которых недостаточно создания сценариев использования готовых решений. При необходимости разработчики организации могут воспользоваться набором инструментов для разработки контента Ansible (CDK, content development kit) от Astra Automation для эффективного решения специфичных задач. CDK позволяет разрабатывать необходимые компоненты:

[4] https://www.cisecurity.org/
[5] https://fstec.ru/

- контейнерные образы для расширения возможностей среды исполнения, поставляемой в составе Astra Automation;
- коллекции Ansible, включающие роли, модули, расширения и другие элементы инфраструктурного кода.

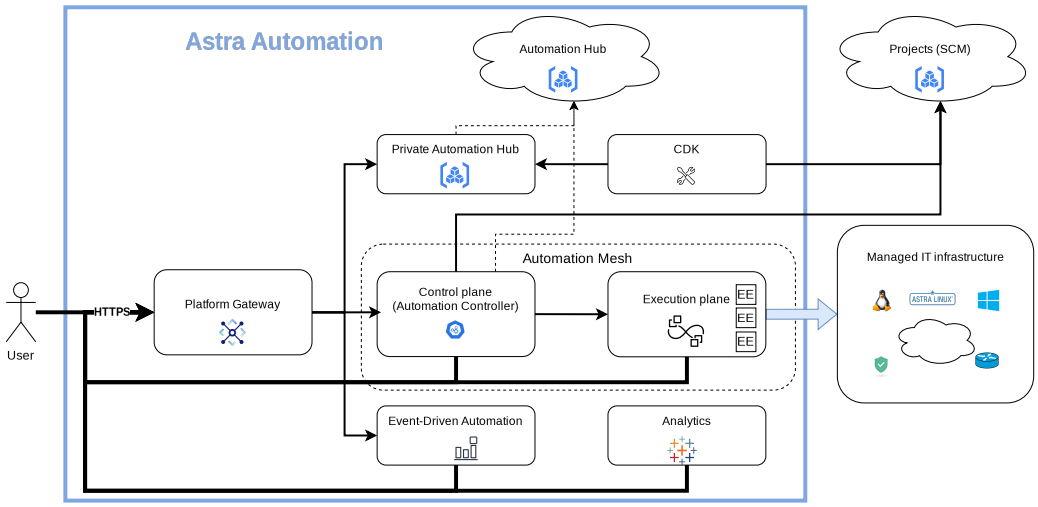
ГЛАВА 3

Обзор архитектуры

Astra Automation является платформой автоматизации, которая включает компоненты управления, устанавливаемые в пространстве организации и облачный сервис, предоставляющий базовый контент для проектов и образы среды исполнения для всех организаций.

3.1 Структура платформы

Состав платформы, взаимодействие ее компонентов и интеграция с внешней средой представлены на следующей схеме:



К платформе относятся следующие компоненты:

- *Объединенный интерфейс (шлюз) платформы* (Platform Gateway) предоставляет единую точку доступа к графическому веб-интерфейсу и API основных компонентов платформы (Automation Controller, Private Automation Hub и Event-Driven Automation) по протоколу HTTPS (при необходимости можно HTTP) для пользователей и программ. Внутреннее взаимодействие компонентов платформы также осуществляется через этот шлюз.
- *Контроллер автоматизации* (Automation Controller) образует плоскость управления (control plane) на основе кластера. Он предоставляет пользователю удобные средства управления через графический веб-интерфейс, вызовы API, командный интерфейс (CLI, command-line interface) и специальную коллекцию Ansible.

- *Плоскость исполнения* (Execution plane) является частью общего кластера, объединяющей узлы, которые выполняют задания контроллера. Особенности плоскости исполнения:
  - Она обеспечивает эффективное управление *распределенной* инфраструктурой путем расположения узлов в одном сегменте сети с узлами управляемой инфраструктуры.
  - Непосредственным исполнителем сценариев автоматизации является среда исполнения (EE, Execution Environment), выполненная в виде контейнера Podman.
  - Контроллер позволяет назначать различные задания на исполнение определенным группам узлов плоскости исполнения, а также выполнять их с помощью указанных образов EE.
- *Сеть прикладного уровня* (Automation Mesh, overlay) объединяет плоскость управления и плоскость исполнения в единый кластер распределенного управления. Настройки платформы позволяют гибко определять связи (топологию) между компонентами этих плоскостей.
- *Automation Hub* является облачным реестром инфраструктурного кода для реализации требуемых задач автоматизации. В состав реестра входят различные типы инфраструктурного кода:
  - сертифицированные коллекции Ansible для выполнения различных задач автоматизации;
  - образы среды исполнения Ansible.
- *Собственный (приватный) реестр* инфраструктурного кода (Private Automation Hub) позволяет организации использовать контент, предоставляемый через Automation Hub или другие источники, а также хранить и использовать собственный контент и образы среды исполнения. При создании собственного реестра в процессе развертывания платформы он получает контент из Automation Hub.
- *Комплект средств разработки* (CDK, Content Development Kit) содержит необходимые инструменты для организации полноценного процесса разработки собственного инфраструктурного кода и образов среды исполнения. Созданные артефакты следует сохранять в собственном реестре инфраструктурного кода.
- *Служба управления событиями* (Event-Driven Automation) позволяет отслеживать события, возникающие в различных компонентах инфраструктуры пользователя. Применение этой службы приводит к существенному снижению объема ручной работы, а также к быстрому и своевременному реагированию на изменившуюся ситуацию в инфраструктуре. Контроллер EDA активирует специальные контейнеры DE (Decision Environment) для активации сводов правил по обработке событий.
- *Средства аналитики* (Analytics) собирают данные о работе контроллера и формируют отчеты для оценки эффективности процессов автоматизации.

Примечание

Компонент находится в стадии разработки и временно недоступен.

3.2 Интеграция

Компоненты платформы взаимодействуют с различными внешними системами:

- *Источники проектов* (Projects) – это внешние системы управления исходным кодом (SCM, Source Code Management), из которых компоненты платформы получают проекты автоматизации. Поддерживаются *VCS* Git (рекомендуется) и Subversion. Проекты содержат основное описание процессов автоматизации:
  - наборы сценариев (playbooks) для Automation Controller;
  - своды правил (rulebooks) для Event-Driven Automation.

- *Внешние источники коллекций* (не приведены на схеме) могут понадобиться для загрузки дополнительных коллекций в приватный реестр. Наиболее известным источником является общедоступный реестр Ansible Galaxy[6].
- *Управляемая инфраструктура* (Managed IT infrastructure) объединяет все управляемые узлы, к которым планируется применение сценариев автоматизации. Она может быть структурно и географически распределенной. В соответствии с этим узлы плоскости исполнения распределяются так, чтобы быть ближе к управляемым узлам, то есть внедряются в локальные сети, где расположены управляемые узлы.

3.3 Объединенный интерфейс

Независимо от способа развертывания платформа предоставляет единый набор интерфейсов для различных типов операторов платформы.

3.3.1 Виды интерфейсов

Для администраторов и конечных пользователей платформа предоставляет следующие способы взаимодействия:

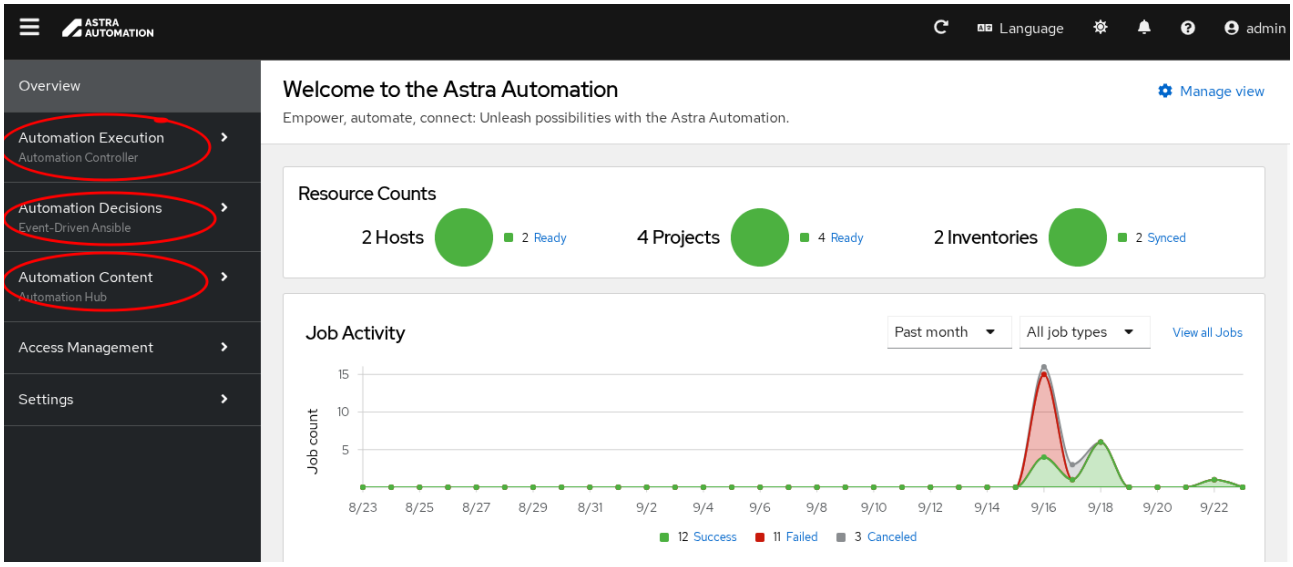
- Графический веб-интерфейс удобен для выполнения повседневных задач, требующих ручного вмешательства.
- API (Application Programming Interface) является базовым интерфейсом, через который реализуются все остальные виды интерфейсов. Прямое управление платформой через API необходимо для программного управления без вмешательства операторов.
- CLI (Command Line Interface) позволяет использовать командную строку для управления платформой с помощью клиента CLI – утилиты awx.

Для службы технической поддержки доступны дополнительные утилиты:

- `awx-manage` предоставляет ряд возможностей по внутренним настройкам, диагностике и другим функциям поддержки локально на узле, входящем в Automation Controller.
- `pulpcore-manager` предоставляет возможности по управлению Private Automation Hub, такими как техническая поддержка базы данных, управление привилегиями пользователей, очистка контента, проверка работоспособности и генерация токена доступа.

3.3.2 Обзор графического интерфейса

Графический интерфейс построен в соответствии со структурной схемой платформы. Он имеет явно выраженное деление на разделы:



[6] https://galaxy.ansible.com/

- Автоматизация процессов (Automation Execution) – создание и управление заданиями по автоматизации процессов в управляемых узлах с помощью проектов автоматизации, предоставляющих сборники сценариев автоматизации (playbooks). Создание ресурсов автоматизации и их применение реализуется через Automation Controller.
- Обработка событий (Automation Decision) – автоматизация процессов по событиям с помощью проектов, предоставляющих своды правил (rulebooks). Создание ресурсов по обработке событий и их применение реализуется через контроллер Event-Driven Automation.
- Контент автоматизации (Automation Content) – хранение и распределение коллекций Ansible и образов сред исполнения. Эти действия реализуются путем настройки Private Automation Hub и загрузки в него необходимого контента.

3.4 Сеть

С помощью сети прикладного уровня с автоматическим обнаружением узлов (Automation Mesh) кластер Astra Automation объединяет узлы разных типов в две плоскости – плоскость управления (control plane) и плоскость исполнения (execution plane). При объединении узлов в сеть происходит настройка связей между узлами плоскостей напрямую или через промежуточные узлы.

Плоскость управления представляет собой Automation Controller и состоит из управляющих и гибридных (объединение функций управления и исполнения) узлов, а плоскость исполнения – из промежуточных и исполняющих узлов.

Узлы кластера объединяются в сеть с помощью службы рецепторов (receptor), установленной на каждом узле.

Рекомендуется распределять узлы плоскости исполнения в соответствии с географическим и иным распределением управляемой инфраструктуры для обеспечения масштабируемости, надежности, снижения задержек и оптимальной производительности платформы в целом.

3.5 Организация данных

Платформа обрабатывает множество данных как для внутреннего управления, так и для передачи в управляемую инфраструктуру.

3.5.1 Базы данных

Каждый основной компонент платформы (Automation Controller совместно с плоскостью исполнения, Private Automation Hub и контроллер EDA) имеет собственную базу данных. Все они создаются на базе СУБД PostgreSQL. Поскольку узлы плоскости исполнения могут быть распределены по сетям управляемой инфраструктуры, необходимо уделять особое внимание доступности базы данных для них.

По умолчанию СУБД создается вместе с базами данных в процессе развертывания платформы, однако, можно использовать и уже существующий сервер или кластер PostgreSQL, развернутый на собственном оборудовании или в одном из облачных сервисов управляемых баз данных.

При развертывании СУБД средствами платформы необходимо выделить для нее отдельный узел, который не должен использоваться для других целей.

Astra Automation не имеет встроенной поддержки кластеризации СУБД. Это значит, что высокую доступность СУБД следует обеспечить самостоятельно.

Использование внешней СУБД накладывает следующие ограничения:

- Astra Automation не поддерживает автоматическое переключение на новый мастер-узел в случае переключения мастера в кластере СУБД.

- Не рекомендуется использовать менеджеры соединений типа pgBouncer[7].
- Нельзя использовать менеджер соединений в режиме «Transaction pooling».

3.5.2 Репозитории

Помимо базы данных, используемой для внутренних целей, Private Automation Hub хранит также данные, необходимые для других компонентов платформы, в отдельных реестрах. К этим реестрам относятся следующие:

- Реестр коллекций обеспечивает Automation Controller коллекциями Ansible для использования их в различных проектах. Кроме того, потребителями коллекций являются различные утилиты Ansible, среди которых следует выделить Ansible Builder, который создает образы сред исполнения с включенными в них коллекциями.
- Реестр образов контейнеров обеспечивает Automation Controller и Event-Driven Automation соответственно образами сред исполнения и сред принятия решений.

Для хранения данных можно использовать одно из следующих решений:

- сервер NFS с монтированием устройства хранения к каждому узлу кластера Private Automation Hub;
- объектное хранилище S3.

3.6 Рабочее окружение

Astra Automation можно развернуть в двух различных окружениях:

- на виртуальных машинах или физических серверах (последний вариант практически не используется);
- в контейнерной среде под управлением Kubernetes.

Существуют ключевые различия между развертыванием на виртуальных машинах и в контейнерах:

[7] https://www.pgbouncer.org/

Критерий	Виртуальные машины (VM)	Kubernetes
Архитектура развертывания	Монолитная; сервисы платформы размещаются на отдельных VM	Микросервисная; сервисы платформы работают в подах на основе контейнеров
Масштабируемость	Требует ручного управления масштабированием VM	Автоматическое масштабирование через механизмы Kubernetes
Обновление платформы	Проверенные Ansible playbooks, медленная миграция	Обновления выполняются через операторы Kubernetes, быстрая миграция с быстрым откатом при необходимости
Управление и мониторинг	Используются стандартные средства Linux и сторонние агенты	Встроенные средства мониторинга (Prometheus, Grafana), централизованное управление журналами
Устойчивость к сбоям	Основывается на архитектуре высокой доступности (HA, High Availability) с применением внешних балансировщиков	Встроенный перезапуск и размещение подов в автоматическом режиме (self-healing)
Управление ресурсами	Явное выделение CPU, RAM, Disk для каждой VM	Динамическое выделение ресурсов, политика обеспечения качества обслуживания (QoS, Quality of Service), политика предпочтений размещения подов на узлах кластера (node affinity)
Интеграция с DevOps	Требует настройки CI/CD вне платформы (Jenkins, GitLab)	Легкая интеграция с GitOps, WebHooks, конвейерами CI/CD Kubernetes
Сетевые особенности	Виртуальные сети, ручная настройка сетевых экранов (firewall)	Сетевые политики, автоматическое обнаружение сервисов (service discovery)

3.6.1 Топология на виртуальных машинах

Виртуальные машины являются традиционными ресурсами для развертывания Astra Automation. В зависимости от требований бизнеса рекомендуется выбрать наиболее подходящую проверенную топологию.

Сравнение топологий

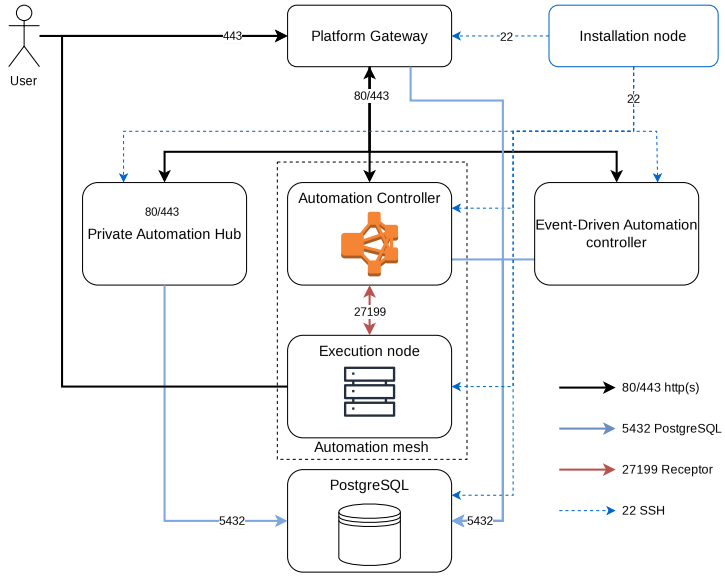
Сравнение топологий развертывания платформы поможет принять правильное решение для вашего бизнеса.

Топология	Количество ВМ	Ключевые компоненты	Область применения
Базовая	7+	Каждый компонент на отдельной ВМ: - Platform Gateway - Automation Controller - Private Automation Hub - Event-Driven Automation - PostgreSQL - Execution node	- Малые производственные среды - Удаленное выполнение задач - Распределенная автоматизация - Начальное внедрение
Уровень предприятия (HA)	13+	Базовая топология + - Дублирование всех компонентов - Внешний кластер PostgreSQL - Внешние балансировщики нагрузки	- Критически важные системы - Высокая доступность - Крупномасштабная автоматизация - Соответствие корпоративным стандартам

Базовая топология

Эта топология не обеспечивает отказоустойчивости, однако может быть удобна для начальной стадии развития процессов автоматизации информационных технологий в компании. Ее можно расширить до топологии масштаба предприятия.

Благодаря подключению исполняющих узлов через сеть Mesh, такую топологию можно использовать в распределенных инфраструктурах.



Особенности реализации платформы:

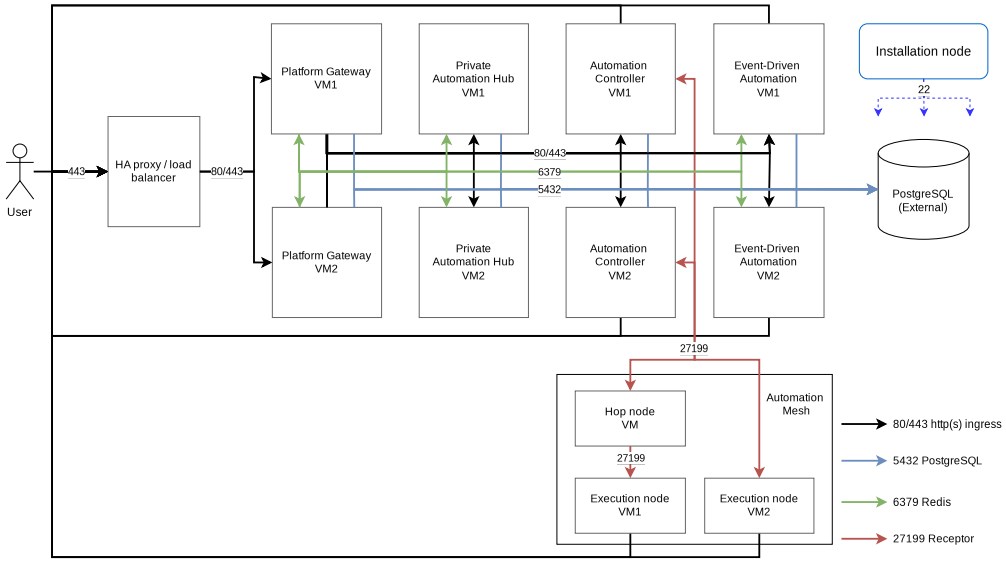
- каждый компонент платформы установлен на отдельной ВМ;
- узел Platform Gateway содержит сервис кеширования данных Redis, используемый совместно им и контроллером Event-Driven Automation;
- Automation Controller и Private Automation Hub используют собственный локальный сервис Redis;

- Automation Controller представлен управляющим вариантом узла, то есть реализует функции управления;
- для исполнения заданий автоматизации используется сеть Mesh с подключением нужного количества исполняющих узлов (Execution node), распределенных по управляемой инфраструктуре;
- СУБД PostgreSQL со всеми требуемыми базами данных устанавливается в процессе развертывания платформы на отдельном узле;
- в состав платформы входит установочный узел (installation node), который не участвует в работе платформы, но выполняет развертывание и обновление всех ее компонентов.

Топология уровня предприятия

Топология уровня предприятия необходима для автоматизации процессов IT в критически важных производственных средах с соблюдением соответствующих требований:

- высокая доступность;
- крупномасштабная автоматизация;
- соответствие корпоративным стандартам отказоустойчивости.

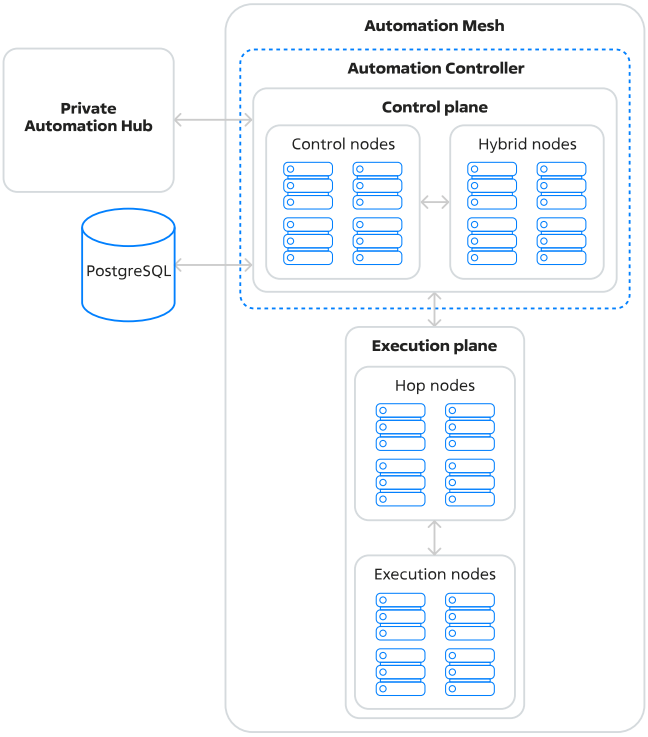


Особенности реализации платформы:

- дублирование всех критических компонентов – минимум 2 экземпляра каждого компонента;
- кластеризация Redis для Platform Gateway и контроллеров EDA – минимум 4 узла для обеспечения высокой доступности с автоматическим переключением при отказах;
- внешняя СУБД PostgreSQL – отдельный кластер управления базами данных;
- балансировщик нагрузки для Platform Gateway, например на основе HAProxy Load Balancer;
- резервирование узлов сети Mesh – множественные промежуточные (hop) и исполняющие (execution) узлы.

Кластер управления

Центральное место в структуре Astra Automation, особенно в топологии уровня предприятия, занимает кластер управления, состоящий из набора узлов, объединенных сетью Mesh.



Сеть Mesh обеспечивает взаимодействие узлов четырех типов:

- *Управляющие узлы* (control nodes) управляют работой контроллера и формируют задания автоматизации.
- *Исполняющие узлы* (execution nodes) используются для непосредственного запуска заданий автоматизации, сформированных управляющими узлами.
- *Промежуточные* (переходные) узлы (hop nodes) используются в качестве посредников между управляющими и исполняющими узлами. Их назначение – связывать узлы, когда прямой доступ управляющих узлов к исполняющим по какой-либо причине невозможен, например, из-за технических ограничений или соображений безопасности.
- *Гибридные узлы* (hybrid nodes) выполняют функции управляющих и исполняющих узлов.

3.6.2 Топология в Kubernetes

Платформа Astra Automation может быть целиком или частично реализована как комплексное приложение внутри среды Kubernetes. Жизненный цикл платформы обеспечивается с помощью *операторов Kubernetes*. Возможны различные варианты топологии, как представлено далее.

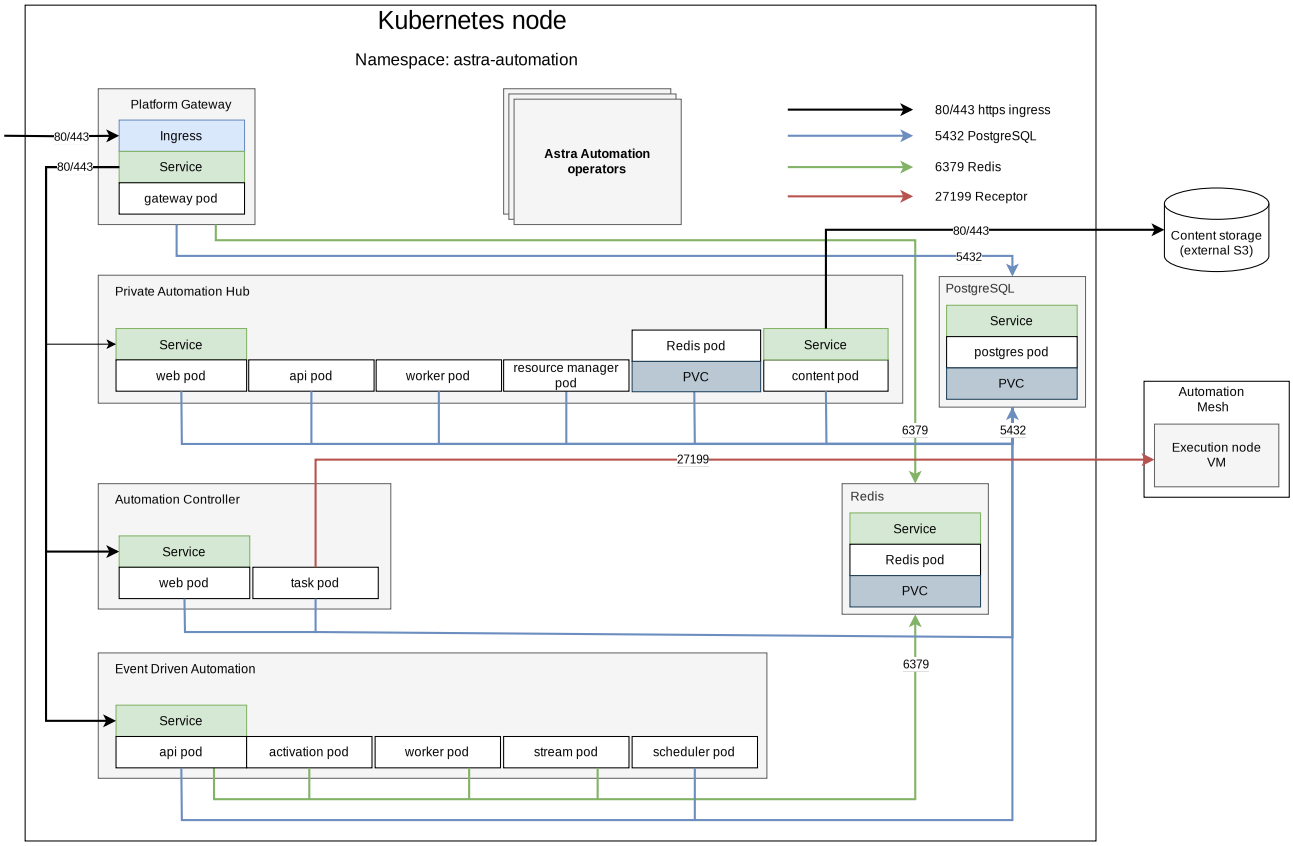
Сравнение топологий

Сравнение топологий развертывания платформы поможет принять правильное решение для вашего бизнеса. По сравнению с реализацией на ВМ переход от одной топологии к другой в рамках Kubernetes представляется гораздо проще.

Топология	Дублирование с помощью ReplicaSet	Внешние ресурсы	Область применения
Базовая (Growth)	Отсутствует	- Хранилище контента в S3 - Automation Mesh	Распределенная автоматизация
Уровень предприятия (Enterprise)	3+ replicas	- Хранилище контента в S3 - PostgreSQL - Automation Mesh - Load Balancers	- Критически важные системы - Крупномасштабная автоматизация - Высокая доступность (24/7) - Обеспечение особых требований

Базовая топология

Эта топология часто используется для начального развертывания платформы с прицелом на ее дальнейшее развитие и переходом на топологию уровня предприятия. Она обеспечивает быстрое развертывание с использованием минимальных ресурсов.



Операторы (Astra Automation operators) создают компоненты платформы в пространстве имен `astra-automation`:

- *Шлюз платформы (Platform Gateway)* реализован в `gateway` pod со следующими контроллерами Kubernetes:
  - Приемник сетевых пакетов `ingress` обеспечивает маршрутизацию запросов из API/UI на соответствующие компоненты платформы.

  - Контроллер `service` обеспечивает связь внутри кластера Kubernetes.

  Оператор, управляющий этим компонентом, также создает поды с соответствующими сервисами для других компонентов платформы:

  - `postgres`, реализующий СУБД PostgreSQL, использует ресурс PVC (Persistent Volume Claim) для запроса дискового пространства в виде постоянного тома (PV, не показан на схеме). В этом томе он создает СУБД, которую используют все компоненты платформы (Platform Gateway, Automation Controller, Private Automation Hub и контроллер EDA) для создания собственных баз данных и управления ими. Доступ к базам данных обеспечивают контроллеры `service` в среде Kubernetes.
  - `redis`, реализующий кеш для Platform Gateway и контроллера EDA, запрашивает дисковое пространство через PVC. Полученный том он использует в качестве надежного хранилища, обеспечивающего восстановление данных в случае пересоздания пода. Доступ к данным обеспечивает контроллер `service`.

- *Automation Controller* – плоскость управления (control plane):
  - `web` и соответствующий `service` обрабатывают входные запросы от веб-интерфейса и API. Они также обеспечивают интеграцию с другими компонентами платформы и внешними ресурсами. Внутри пода реализован Redis для обмена данными в пределах контроллера.
  - Для выполнения заданий выделен отдельный `task` pod.
- *Private Automation Hub* – реестр инфраструктурного кода, реализованный с помощью набора компонентов:
  - `web` и соответствующий `service` принимают запросы от веб-интерфейса и API и выполняют их первичную обработку.
  - `api` обрабатывает запросы REST API.
  - `worker` обрабатывает фоновые задачи и асинхронные операции.
  - `resource manager` управляет ресурсами и их распределением.
  - `redis` обеспечивает кеширование промежуточных данных.
  - `content` управляет контентом, включая коллекций Ansible и артефакты в виде образов среды исполнения и среды принятия решений. Он обеспечивает хранение контента в файловой системе отдельного хранилища.
  - Content storage – хранилище контента. Выбор устройства хранения зависит от того, планируете ли вы развивать платформу в дальнейшем, увеличивая количество узлов приватного реестра. Если нет, то можно использовать PV. Если планируете, то следует выбирать хранилище, которое обеспечивает доступ к нему со стороны нескольких узлов в режиме чтения-записи (RWX, ReadWriteMany). Не все облачные системы, предоставляющие сервис Managed Kubernetes, обеспечивают это. Поэтому в общем случае рекомендуется использовать внешнее хранилище в виде S3, которое обеспечивает такой режим.
- *Event-Driven Automation* – сервис, отслеживающий события в инфраструктуре и инициирующий автоматические сценарии. Он реализуется в виде следующих компонентов:
  - `api` обеспечивает обработку запросов от Platform Gateway по протоколу HTTP/HTTPS.
  - `activation` управляет активациями правил по обработке событий.
  - `worker` обрабатывает фоновые задачи и асинхронные операции с помощью среды принятия решений (DE).
  - `stream` обеспечивает связь с внешними источниками событий типа Kafka, Webhook и другие.
  - `scheduler` планирует обработку событий с помощью других компонентов, перечисленных ранее, в определенной последовательности. В его обязанности входят также формирование и посылка запросов на выполнение заданий в Automation

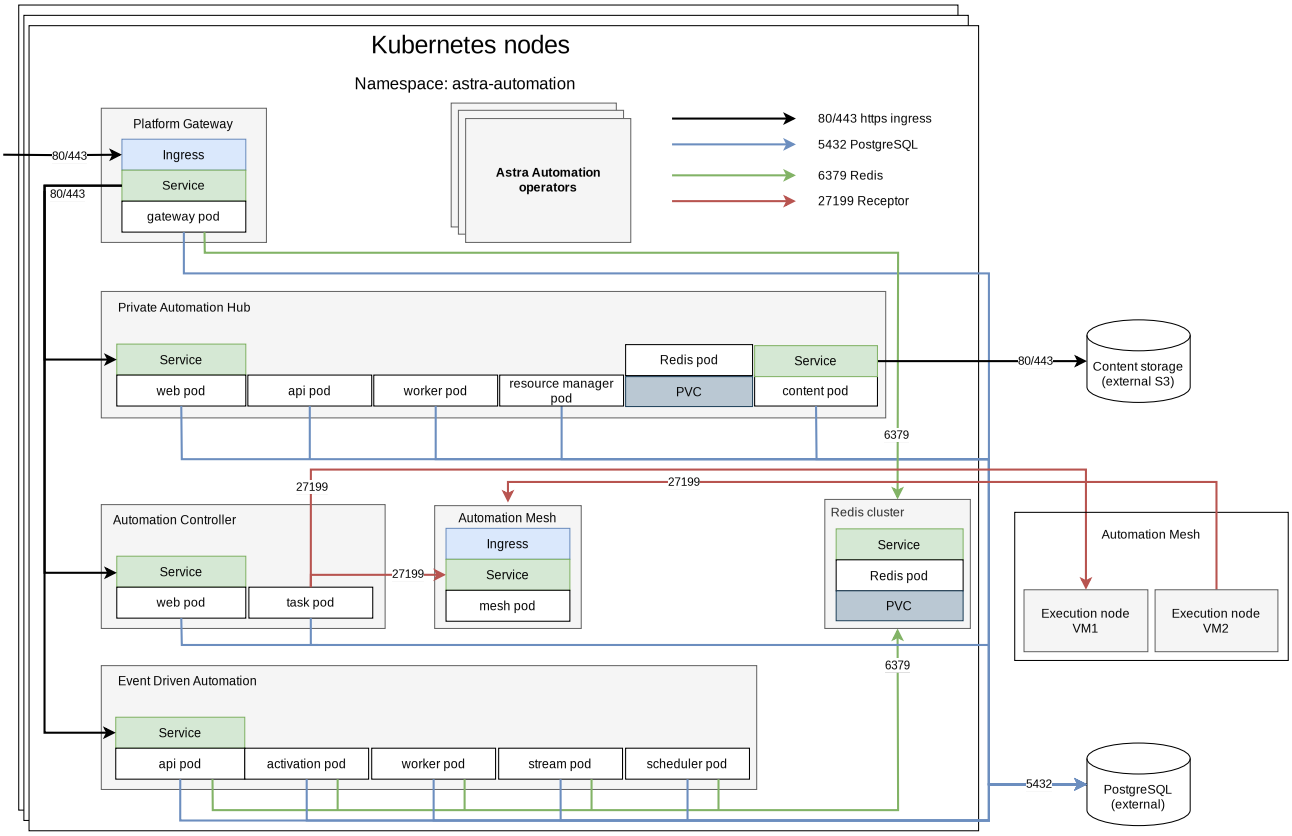
Controller, управление очередями событий в Redis и хранение состояния активаций и истории выполнения заданий в базе данных.

К центральным компонентам добавляется сеть Mesh с делением системы автоматизации на две плоскости: плоскость управления (Automation Controller, Control Plane) и плоскость исполнения (Execution Plane). Последняя содержит исполняющие узлы (Execution nodes) для создания распределенной системы управления в соответствии с топологией инфраструктуры, подлежащей автоматизации. Появляется дополнительное сетевое соединение через сеть Mesh, которое обеспечивает взаимодействие между управляющим и исполняющими узлами с помощью службы рецепторов. Необходима настройка сетевых фильтров на связь по порту TCP 27199 для взаимодействия рецепторов. При необходимости в топологию можно добавить переходные узлы (hop nodes) для маршрутизации трафика и обеспечения связи с исполняющими узлами, расположенными в сложных или изолированных сетевых сегментах.

Как правило, исполняющие и переходные узлы реализуют в виде ВМ или физических серверов, приближенных к сегментам инфраструктуры, которыми они управляют.

Топология уровня предприятия

Топология уровня предприятия необходима для инфраструктуры с высокими требованиями к устойчивости платформы автоматизации.



По сравнению с базовой топологией уровень предприятия добавляет множество ресурсов и возможностей Kubernetes:

- Три или более рабочих узлов (worker nodes) необходимы для обеспечения доступности компонентов платформы в случае выхода из строя одного из них.
- Развитая масштабируемая сеть Mesh обеспечивает необходимую производительность и надежность процессов автоматизации для сетей практически любой сложности. Для обеспечения лучшей масштабируемости добавлен промежуточный узел Automation Mesh Ingress (hop node), который принимает запросы от исполняющих узлов (execution nodes) на установление соединения с Automation Controller посредством сервиса receptor. Таким образом обеспечивается возможность установления соединений по инициативе контроллера (от task pod) или по инициативе исполняющих узлов.

- Для еще более высокой устойчивости рекомендуется центральную часть платформы, реализуемую через Kubernetes, развертывать на рабочих узлах, распределенных географически. В облачных сервисах Managed Kubernetes это обеспечивается использованием двух и более зон доступности.
- Базы данных компонентов платформы необходимо создавать с использованием корпоративного кластера СУБД PostgreSQL, обеспечивающего его высокую доступность по определенному IP-адресу и порту TCP.
- Служба Ingress в Kubernetes обеспечивает балансировку нагрузки. При географически распределенном кластере Kubernetes необходим внешний балансировщик нагрузки для Platform Gateway, например на основе HAProxy Load Balancer.

Операторы Kubernetes для Astra Automation

Развертывание платформы в среде Kubernetes осуществляется с помощью операторов Astra Automation. В соответствии с функциональным делением платформы на компоненты используются следующие операторы:

- `aa-operator-aa-manager` является корневым оператором, управляющим шлюзом платформы (Platform Gateway) и общими компонентами – Redis и PostgreSQL.
- `ac-operator-ac-manager` управляет Automation Controller.
- `pah-operator-pah-manager` управляет Private Automation Hub и хранилищем контента.
- `eda-operator-eda-manager` управляет контроллером Event-Driven Automation.

Название оператора определяется в его манифесте (см. пояснение далее) через название основного компонента, которым обычно является `Deployment` или `StatefulSet`. В определении основного компонента это будет поле `metadata.name`.

Принцип действия

Оператор Kubernetes – это постоянная вспомогательная служба, которая обслуживает жизненный цикл приложения, включая его создание, периодические обновления и удаление. Для развертывания оператора используют файл настроек (манифест) в формате YAML, который содержит как описание компонентов самого оператора (различные поды, сервисы поверх них и др.), так и определение ресурсов приложения – CRD (Custom Resource Definition).

Примечание

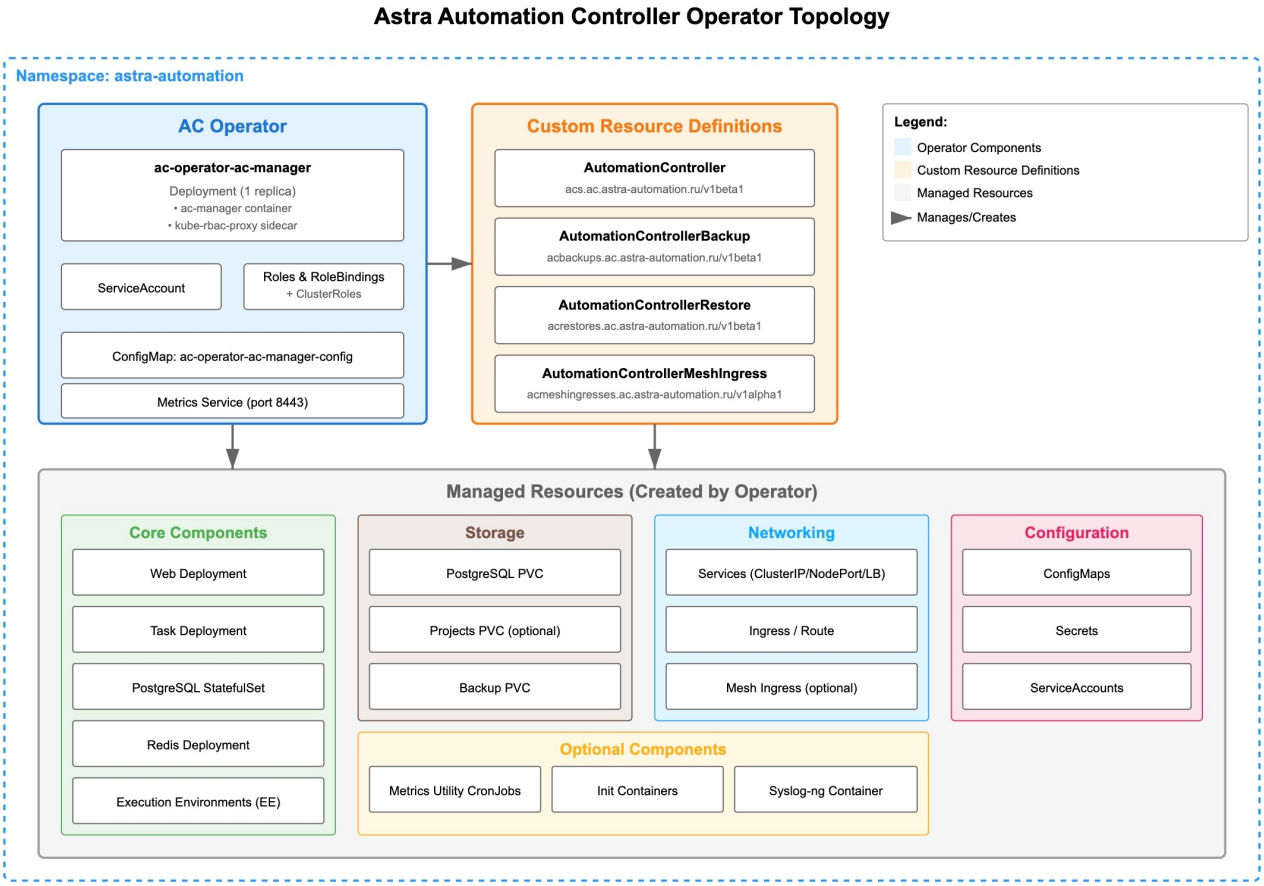
Манифест оператора является универсальным для всех рассмотренных ранее вариантов топологии платформы Astra Automation.

Управление через оператор представляется в виде следующих шагов, выполняемых по инициативе сотрудников, отвечающих за жизненный цикл приложения:

1. Установка оператора с помощью его манифеста.
2. Подготовка манифеста для развертывания приложения через оператор.
3. Развертывание приложения с помощью манифеста, в котором используются ресурсы оператора. Приложение базируется на собственных типах ресурсов (CR, Custom Resource), определенных через CRD оператора. На этом шаге вы определяете необходимую вам топологию платформы.
4. Поддержка жизненного цикла приложения:
   - мониторинг состояния и нагрузки компонентов приложения;
   - автоматическое поддержание компонентов приложения в рабочем состоянии;
   - обновление приложения по запросу.
5. Удаление приложения.

Пример

Рассмотрим структуру установленного оператора и управляемого им приложения на примере Automation Controller.



Для краткости компоненты представлены в обобщенном виде без подробностей. Манифест оператора представляет собой сложную структуру, состоящую из нескольких тысяч строк.

Компоненты оператора

После развертывания оператор находится в режиме ожидания запросов от сотрудников, управляющих жизненным циклом платформы. В рабочем состоянии находятся следующие компоненты в пространстве `astra-automation`:

- **AC Operator** deployment – менеджер оператора `ac-operator-ac-manager`. Это основной компонент оператора, базирующийся на следующих контейнерах:
  - `ac-operator-ac-manager` – контейнер, реализующий основную логику оператора;
  - `kube-rbac-proxy` – сопровождающий (sidecar) контейнер для проксирования метрик, доступ к которому обеспечивает ролевая система управления (RBAC, Role-Based Access Control).

  Остальные объекты предназначены для обслуживания этого ресурса.
- **ServiceAccount** – сервисная учетная запись для идентификации менеджера при его взаимодействии с Kubernetes.
- **Roles & RoleBindings** – роли типа `ClusterRole` и привязка их к сервисной учетной записи (тип `RoleBinding`), обеспечивающие необходимые привилегии для менеджера в Kubernetes.
- **ConfigMap** – настройки менеджера (`ac-operator-ac-manager-config`).
- **Metrics Service** – сервис для экспорта метрик, доступный по порту TCP 8443.

В манифесте оператора определены четыре специальных ресурса приложения (CRD) в группе `ac.astra-automation.ru`:

- **AutomationController** (`acs.ac.astra-automation.ru/v1beta1`) – основной ресурс для развертывания управляющего узла Automation Controller.
- **AutomationControllerBackup** (`acbackups.ac.astra-automation.ru/v1beta1`) – ресурс для создания резервных копий контроллера.
- **AutomationControllerRestore** (`acrestores.ac.astra-automation.ru/v1beta1`) – ресурс для восстановления контроллера из резервной копии.
- **AutomationControllerMeshIngress** (`acmeshingresses.ac.astra-automation.ru/v1alpha1`) – ресурс для настройки сетевого доступа через сервис Mesh.

> **Примечание**
>
> На текущий момент этот ресурс не поддерживается. Связь с сетью Mesh осуществляется через `task` pod в Automation Controller.

Каждое описание CRD определяет комплексную архитектуру, состоящую из множества объектов Kubernetes.

Компоненты приложения

При создании приложения AutomationController необходимо в среде Kubernetes исполнить манифест, требующий создания ресурсов, описанных в CRD оператора. В манифесте указывают значения параметров, которые необходимы для реализации требуемой топологии. Если они не указаны, используются значения по умолчанию. Оператор автоматически создает требуемые компоненты приложения, которые в представленной ранее схеме заданы обобщенно без подробностей реализации. Более подробно каждый компонент Astra Automation представлен в *описании топологии*.

Основные компоненты (**Core Components**) приложения:

- **Web Deployment** – объекты веб-интерфейса Automation Controller;
- **Task Deployment** – под для выполнения задач;
- **PostgreSQL StatefulSet** – объекты для доступа к базе данных PostgreSQL;
- **Redis Deployment** – объекты обеспечения работы Redis;
- **Execution Environments (EE)** – среда исполнения заданий автоматизации.

Средства хранения данных (Storage):

- **PostgreSQL PVC** – запрос на предоставление постоянного хранилища для базы данных;
- **Projects PVC (optional)** – запрос на предоставление постоянного хранилища для проектов автоматизации;
- **Backup PVC** – запрос на предоставление постоянного хранилища для резервных копий.

Сетевые компоненты (Networking):

- **Services** – сервисы различного типа для доступа к компонентам контроллера:
  - внутри кластера Kubernetes с помощью приватных адресов (**ClusterIP**);
  - извне через IP-адрес рабочего узла кластера и порт TCP, выделенный для конкретного сервиса (**NodePort**);
  - извне через выделенный публичный IP-адрес (**LoadBalancer**);
- **Ingress/Route** – внешний доступ к объединенному интерфейсу;

Настройки (Configuration) в виде подключаемых томов:

- **ConfigMaps** – настройка приложения в виде параметров типа ключ-данные;
- **Secrets** – секреты для паролей, ключей и сертификатов;
- **ServiceAccounts** – сервисные учетные записи для компонентов.

Дополнительные компоненты (**Optional Components**):

- **Metrics Utility CronJobs** – запланированные задания для сбора метрик по расписанию;
- **Init Containers** – служебные контейнеры для подготовки к запуску основных контейнеров;
- **Syslog-ng Container** - контейнер для централизованного ведения журналов.

ГЛАВА 4

Ansible Core

Ansible – это инструмент для автоматизации процессов управления инфраструктурой и конфигурацией. Он позволяет автоматизировать развертывание приложений, настройку серверов, управление сервисами и многим другим. Ansible использует простой синтаксис, в основном с использованием формата YAML, хотя применяются и другие форматы, например, INI и JSON. В большинстве сценариев управления серверами и другими компьютерами Ansible подключается к ним по протоколу SSH, загружает и выполняет модули Python для реализации заданных задач автоматизации. В таком сценарии необходимо создать ключи SSH, позволяющие управлять такими узлами. Установка агентов на управляемые узлы не требуется. В других случаях можно использовать расширения, которые позволяют Ansible управлять узлами с помощью других протоколов.

Ansible Core представляет собой ядро этого инструмента и является основой всей платформы. Это минимальная и обязательная часть Ansible, которая обеспечивает основные функции:

- обработку инвентарных списков;
- управление подключениями;
- выполнение базовых модулей управления на удаленных узлах.

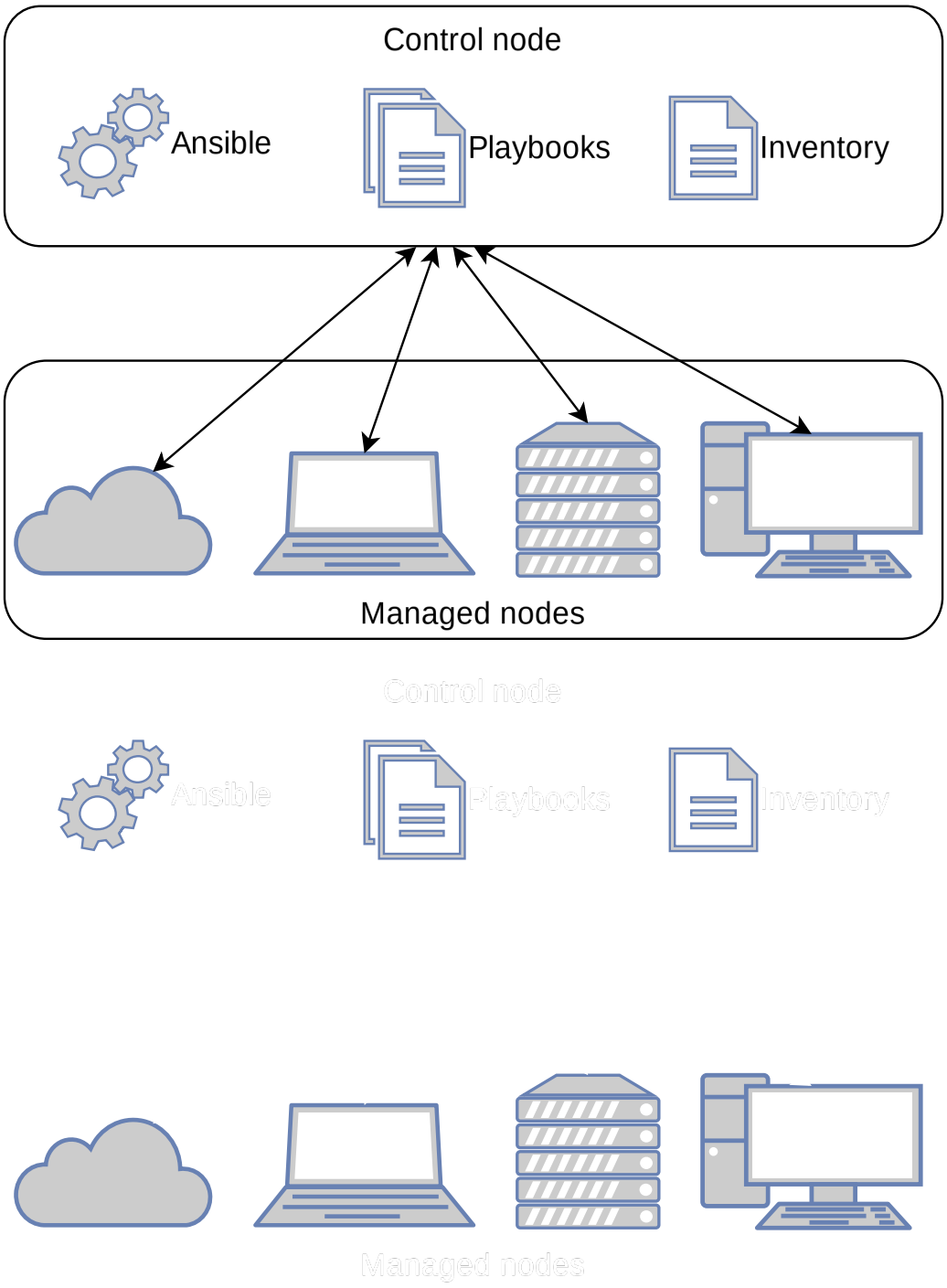
Ansible Core включает в себя следующие компоненты:

- утилиты командной строки:
  - `ansible` – выполняет одиночные команды на управляемых узлах;
  - `ansible-playbook` – запускает сценарии автоматизации;
  - `ansible-galaxy` – управляет коллекциями и ролями;
  - `ansible-inventory` – работает с инвентарем;
  - `ansible-config` – управляет конфигурацией Ansible;
  - `ansible-vault` – используется для шифрования и защиты данных.
- встроенную коллекцию `ansible.builtin`, в которую входят основные модули, например, `file`, `service`, `user` и другие.

Базовая функциональность Ansible обеспечивается большим количеством встроенных модулей. При этом возможности Ansible можно расширять с помощью коллекций. Подробную информацию см. в документе *Коллекции Ansible*.

Структура управления

При использовании Ansible управляющий узел подключается к управляемым узлам и выполняет по отношению к ним команды управления.



Управляющий узел

Управляющим узлом (control node) Ansible из состава Astra Automation является рабочая станция под управлением Astra Linux, на которой установлены компоненты Ansible и которую используют для выполнения различных задач по автоматизации управляемых узлов.

Управляемые узлы

Управляемым узлом (managed node) называется узел, к которому можно обращаться как к самостоятельной единице с целью автоматизации процессов на нем с помощью Ansible. Это могут быть любые типы компьютеров, сетевое оборудование, средства хранения, технологическое оборудование и тому подобное.

Факты

Факты (facts) – это набор сведений об управляемом узле, например, архитектура и версия установленной ОС, версия BIOS, наличие в системе определенных устройств и тому подобное.

Настройки

Во время работы Ansible использует настройки, переданные в аргументах командной строки, заданные в переменных окружения или указанные в файле настроек `ansible.cfg`. Поиск файла настроек `ansible.cfg` выполняется в следующем порядке:

1. Каталог, указанный в значении переменной окружения `ANSIBLE_CONFIG`.
2. Текущий каталог.
3. Домашний каталог активного пользователя.
4. Каталог `/etc/ansible/`.

Поиск прекращается как только файл `ansible.cfg` будет найден в любом из указанных расположений.

4.1 Подготовка управляемых узлов

Чтобы узлами можно было управлять с помощью Ansible, необходимо соответствующим образом их настроить.

4.1.1 ОС Linux

Для подключения к управляемым узлам под управлением ОС Linux рекомендуется использовать протокол SSH. Он позволяет авторизоваться на узле различными способами, но чаще всего используются следующие два:

- пароль;
- ключи шифрования (далее – ключи).

Рекомендуемым способом авторизации является использование ключей. Для этого необходимо создать пару ключей – приватный и публичный. Приватный ключ необходимо оставить на управляющем узле, а публичный разместить на управляемых узлах.

Генерация ключей

Для создания пары ключей выполните следующие действия:

1. Перейдите в каталог `~/.ssh/`:

   ```
   cd ~/.ssh/
   ```

2. Выполните команду:

   ```
   ssh-keygen -C "<comment>" -f ./<filename> -N "<password>"
   ```

   где:

   - <comment> – комментарий к ключу.
   - <filename> – название файла для сохранения приватного ключа. Публичный ключ хранится в файле с тем же названием, к которому добавляется расширение `.pub`.
   - <password> – пароль для защиты приватного ключа. Если в качестве пароля используется пустая строка `""`, приватный ключ создается без защиты.

   Подробности о команде `ssh-keygen` доступны во встроенной справке:

   ```
   man ssh-keygen
   ```

Пример

Для создания пары ключей можно использовать следующую команду:

```
ssh-keygen -C "Astra Automation" -f ~/.ssh/astra-automation
```

Утилита ssh-keygen запросит пароль для защиты приватного ключа, после чего создаст в каталоге ~/.ssh/ два файла:

- astra-automation – приватный ключ;
- astra-automation.pub – публичный ключ.

3. Установите на приватный ключ права доступа 600:

```
chmod 600 ./<filename>
```

где <filename> – название файла, в котором сохранен приватный ключ.

Примечание

Права доступа 600 означают, что только пользователь-владелец файла имеет права на чтение и запись.

Настройка целевых узлов

Чтобы разрешить подключение к узлу по SSH с использованием ключей, выполните следующие действия:

1. Установите пакеты сервера OpenSSH:

```
sudo apt update && sudo apt-install openssh-server --yes
```

2. Включите автоматический запуск сервера OpenSSH при загрузке системы:

```
sudo systemctl enable ssh.service
```

3. В каталоге /etc/ssh/sshd_config.d/ создайте файл ssh_auth_keys со следующим содержимым:

```
PubkeyAuthentication  yes
AuthorizedKeysFile   .ssh/authorized_keys  .ssh/authorized_keys2
```

4. Перезапустите сервер OpenSSH:

```
sudo systemctl restart ssh
```

5. Выберите учетную запись, которую будете использовать для авторизации по протоколу SSH при выполнении сценариев автоматизации. В домашнем каталоге этой учетной записи создайте подкаталог .ssh/, а в нем – файл authorized_keys. Владельцем домашнего каталога и всех файлов внутри него должна быть выбранная учетная запись.

6. Установите на файл ~/.ssh/authorized_keys права доступа 600:

```
chmod 600 ~/.ssh/authorized_keys
```

7. Добавьте в файл ~/.ssh/authorized_keys содержимое соответствующего публичного ключа, например (часть содержимого ключа опущена с целью сокращения):

```
ssh-rsa AAAAB3NzaC1yc2EAd...4bwy3tY2/ administrator@node1.example.com
```

Совет

Если узел разрешает подключение с использованием протокола SSH с авторизацией по паролю, публичную часть ключей можно разместить с помощью утилиты `ssh-copy-id`:

```
ssh-copy-id -i <filename> <user>@<host>
```

где:

- <filename> – путь к файлу публичного ключа;
- <user> – название учетной записи пользователя узла;
- <host> – IP-адрес или FQDN узла.

4.1.2 ОС Windows

Для подключения к управляемым узлам с установленной ОС Windows используется протокол WinRM.

Чтобы разрешить такое подключение, выполните следующие действия на этом узле с помощью базовых команд Windows и PowerShell:

1. Откройте командную строку и создайте отдельного пользователя для управления узлом с помощью Ansible:

```
net user ansibleadmin password123! /add
```

2. Добавьте созданного пользователя в группу Администраторы:

```
net localgroup Администраторы ansibleadmin /add
```

3. Запустите PowerShell с привилегиями администратора и включите утилиту WinRM:

```
Enable-PSRemoting -Force
```

4. В настройках брандмауэра разрешите входящие соединения через порт 5985 по протоколу TCP:

```
netsh advfirewall firewall add rule name="Allow WinRM" dir=in action=allow␣
↪protocol=TCP localport=5985
```

5. Отключите фильтрацию токенов UAC:

```
New-ItemProperty -Path "HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\
↪System" `
                 -Name "LocalAccountTokenFilterPolicy" `
                 -Value 1 `
                 -PropertyType DWORD `
                 -Force
```

6. Перезагрузите службу WinRM:

```
Restart-Service WinRM
```

7. Проверьте конфигурацию:

```
winrm get winrm/config/service/auth
```

   Если значение параметра `Negotiate` равно `true`, настройки выполнены корректно.

4.2 Утилиты

Важно

Эта часть документации находится в стадии разработки.

4.3 Инвентаризация

Терминология:

- *Инвентарь* – совокупность управляемых узлов (managed nodes).
- *Описание инвентаря* (inventory) или инвентарный список – файлы или другие источники данных, определяющие перечень и параметры узлов, управляемых через сценарии автоматизации (playbooks) или отдельные задачи.
- *Инвентаризация* – процесс создания описания инвентаря.

4.3.1 Описание инвентаря

Описание инвентаря может быть создано следующими способами:

- статическая инвентаризация – создание и хранение описания инвентаря в файлах;
- динамическая инвентаризация – создание описания инвентаря в процессе выполнения задания.

Независимо от способа, перед выполнением требуемых сценариев автоматизации Ansible проводит *окончательную инвентаризацию* – обрабатывает исходное представление инвентаря для создания описания, используемого далее в этих сценариях. При запуске утилиты `ansible-playbook` можно задавать несколько источников, используя как статическую, так и динамическую инвентаризацию:

```
ansible-playbook -i <inventory-1> -i <inventory-2> ... <playbook>.yml
```

Ключевые правила:

- Каждый источник представляется значением аргумента `-i`.
- При возникновении конфликта между источниками последний из этих источников имеет более высокий приоритет.

4.3.2 Расширения из Ansible Core

Для инвентаризации Ansible использует (загружает) следующие расширения из коллекции `ansible.builtin`:

- `ansible.builtin.host_list` разбирает строку с перечнем узлов, включая диапазоны названий.
- `ansible.builtin.ini` использует файлы формата INI как источники описания инвентаря.
- `ansible.builtin.yaml` использует файлы формата YAML как источники описания инвентаря.
- `ansible.builtin.toml` использует файлы формата TOML как источники описания инвентаря.
- `ansible.builtin.script` выполняет внешний скрипт, возвращающий описание инвентаря в формате JSON.
- `ansible.builtin.constructed` формирует группы и переменные на основе шаблонов Jinja2.
- `ansible.builtin.generator` создает группы и узлы по шаблонам.

- `ansible.builtin.auto` автоматически определяет нужное расширение Ansible по ключу `plugin` в файлах YAML и загружает его.

Статическая инвентаризация

Описание инвентаря статическим способом осуществляется в помощью текстовых файлов в формате YAML (предпочтительный) или INI.

Основы инвентаризации

В самом простом случае описание инвентаря может содержать для каждого управляемого узла только его IP-адрес:

INI

```
[all]
192.168.56.11
192.168.56.12
192.168.56.13
```

YAML

```
---
all:
  hosts:
    192.168.56.11:
    192.168.56.12:
    192.168.56.13:
```

Здесь `all` – название группы, которая используется для классификации узлов.

Также допускается использовать доменные имена, например:

INI

```
[all]
dc01.example.com
dc02.example.com
dc03.example.com
```

YAML

```
---
all:
  hosts:
    dc01.example.com:
    dc02.example.com:
    dc03.example.com:
```

В этом случае разрешение доменного имени каждого узла в IP-адрес происходит путем обращения к системной службе разрешения имен. В Astra Linux настройки этой службы хранятся в файле `/etc/nsswitch.conf`. По умолчанию в нем указан следующий порядок разрешения имен:

1. Проверка записей, хранящихся в файле `/etc/hosts`.
2. Обращение к серверу DNS, указанному в настройках сети.

Чтобы сделать описание инвентаря не зависящим от службы разрешения имен, в описании каждого узла укажите в поле `ansible_host` его IP-адрес, например:

INI

```
[all]
dc01.example.com   ansible_host=192.168.56.11
dc02.example.com   ansible_host=192.168.56.12
dc03.example.com   ansible_host=192.168.56.13
```

YAML

```
---
all:
hosts:
  dc01.example.com:
    ansible_host: 192.168.56.11
  dc02.example.com:
    ansible_host: 192.168.56.12
  dc03.example.com:
    ansible_host: 192.168.56.13
```

Группы по умолчанию

Даже если вы не определяете группы в описании инвентаря, Ansible создает две группы по умолчанию:

- `all` – все узлы;
- `ungrouped` – все узлы, не входящие ни в одну из групп, кроме `all`.

Каждый узел всегда будет входить как минимум в две группы (`all` и `ungrouped` или `all` и какую-либо другую группу). Например, в приведенном выше примере инвентарного списка узел `dc01.example.com` входит в группу `all` и группу `ungrouped`. Хотя группы `all` и `ungrouped` всегда присутствуют, они могут быть неявными и не отображаться в списках групп, таких как `group_names`.

Узлы в нескольких группах

Вы можете включать узлы в несколько групп для более логичного размещения, например:

- по типу – сервер базы данных или веб-сервер;
- по расположению – восток или запад;
- по этапу разработки – разработка или эксплуатация.

INI

```
[webservers]
web01.example.com
web02.example.com

[dbservers]
db01.example.com
db02.example.com

[east]
web01.example.com
db02.example.com

[west]
web02.example.com
db01.example.com

[devel]
web01.example.com
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
db01.example.com

[prod]
web02.example.com
db02.example.com
```

YAML

```
---
webservers:
  hosts:
    web01.example.com:
    web02.example.com:
dbservers:
  hosts:
    db01.example.com:
    db02.example.com:
east:
  hosts:
    web01.example.com:
    db02.example.com:
west:
  hosts:
    web02.example.com:
    db01.example.com:
devel:
  hosts:
    web01.example.com:
    db01.example.com:
prod:
  hosts:
    web02.example.com:
    db02.example.com:
```

Родительские и дочерние группы

В Ansible можно создавать родительские и дочерние группы для организации иерархии и более удобной настройки. Группы могут быть вложенными, то есть одна группа может включать в себя другие группы.

Дочерние группы указываются после `children:` или в группе формата INI – `[<parent_group>:children]`.

Дочерние группы обладают следующими свойствами:

- любой узел, являющийся членом дочерней группы, автоматически становится членом родительской группы;
- группы могут иметь несколько родительских и дочерних групп, но не циклические связи.

Пример с родительскими и дочерними группами:

INI

```
[webservers]
web01.example.com
web02.example.com

[dbservers]
db01.example.com
db02.example.com
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
[east]
web01.example.com
db02.example.com

[west]
web02.example.com
db01.example.com

[regional_servers:children]
east
west
```

YAML

```
---
webservers:
  hosts:
    web01.example.com:
    web02.example.com:
dbservers:
  hosts:
    db01.example.com:
    db02.example.com:
east:
  hosts:
    web01.example.com:
    db02.example.com:
west:
  hosts:
    web02.example.com:
    db01.example.com:
regional_servers:
  children:
    east:
    west:
```

Диапазон узлов

При использовании большого количества узлов с похожими наименованиями, их можно добавить в виде диапазона, а не перечислять каждый отдельно:

INI

```
[dbservers]
db[01:10].example.com
```

YAML

```
---
dbservers:
  hosts:
    db[01:10].example.com:
```

При определении числового диапазона можно указать разницу между порядковыми номерами:

INI

```
[dbservers]
db[01:10:2].example.com
```

YAML

```
---
dbservers:
  hosts:
    db[01:10:2].example.com:
```

Также можно использовать алфавитный диапазон:

INI

```
[dbservers]
db-[a:f].example.com
```

YAML

```
---
dbservers:
  hosts:
    db-[a:f].example.com:
```

Использование переменных при заполнении описания инвентаря

Переменные в описании инвентаря позволяют управлять конфигурацией узлов и групп, задавая параметры, такие как учетные данные, пути к исполняемым файлам, настройки сети и прочее.

Переменные поведения

Переменные поведения – это встроенные переменные, которые определяют способ взаимодействия Ansible с узлами. Они определяют параметры подключения, выполнения и другие аспекты работы.

Наиболее часто используемые переменные поведения:

- `ansible_host` – IP-адрес или доменнное имя узла;
- `ansible_user` – название учетной записи пользователя;
- `ansible_port` – порт подключения;
- `ansible_become` – использование повышенных привилегий;
- `ansible_become_user` – название учетной записи, используемой для повышения привилегий;
- `ansible_connection` – метод подключения;
- `ansible_ssh_private_key_file` – путь к приватному ключу SSH;
- `ansible_python_interpreter` – путь к интерпретатору Python на удаленном узле.

Описание всех возможных переменных поведения см. в *справочнике*.

Назначение переменных в описании инвентаря

Переменные в описании инвентаря указываются после `vars:` или в группе формата INI – `[<group_name>:vars]`, например:

INI

```
[webservers]
web01 ansible_host=192.168.56.11 ansible_user=admin ansible_port=2222
web02 ansible_host=192.168.56.12

[dbservers]
db01 ansible_host=192.168.56.13 ansible_user=root
db02 ansible_host=192.168.56.14 ansible_user=root

[dbservers:vars]
ansible_port=3306
```

YAML

```
---
webservers:
  hosts:
    web01:
      ansible_host: 192.168.56.11
      ansible_user: admin
      ansible_port: 2222
    web02:
      ansible_host: 192.168.56.12
dbservers:
  hosts:
    db01:
      ansible_host: 192.168.56.13
      ansible_user: root
     db02:
       ansible_host: 192.168.56.14
       ansible_user: root
     vars:
      ansible_port: 3306
```

Использование каталогов group_vars и host_vars

Каталоги group_vars и host_vars должны быть расположены в каталоге с описанием инвентаря.

Рекомендуемая структура каталогов:

```
inventory/
├── group_vars
│   ├── group_vars1.yml
│   ├── group_vars2.yml
│   │   ...
│   └── group_varsN.yml
├── host_vars
│   ├── hosts_vars1.yml
│   ├── hosts_vars2.yml
│   │   ...
│   └── hosts_varsM.yml
└── inventory.yml
```

Здесь:

- inventory.yml – файл с описанием инвентаря;
- group_vars/ – каталог с файлами, содержащими переменные для разных групп узлов;
- host_vars/ – каталог с файлами, содержащими переменные для конкретных узлов.

Ansible автоматически загружает файлы из каталогов group_vars и host_vars, если они находятся в том же каталоге, что и файл описания инвентаря, или путь к нему указан при

запуске с помощью команды:

```
ansible-playbook -i inventory.yml playbook.yml
```

Применение нескольких источников инвентаризации

Ansible позволяет указывать несколько файлов и источников инвентаризации, объединяя их при выполнении команд. Это удобно, если инфраструктура описана в разных файлах.

Вы можете объединить несколько источников инвентаризации в один каталог.

Пример структуры каталогов с использованием нескольких файлов с описанием инвентаря:

```
inventory/
├── group_vars
│   ├── group_vars1.yml
│   ├── group_vars2.yml
│   │   ...
│   └── group_varsN.yml
├── host_vars
│   ├── hosts_vars1.yml
│   ├── hosts_vars2.yml
│   │   ...
│   └── hosts_varsM.yml
├── inventory1.yml
└── inventory2.yml
```

Чтобы использовать несколько файлов, укажите их с помощью аргумента `-i` при вызове утилиты `ansible-playbook`:

```
ansible-playbook -i inventory/hosts1.yml -i inventory/hosts2.yml playbook.yml
```

Также можно указать каталог с файлами:

```
ansible-playbook -i inventory/ playbook.yml
```

В этом случае утилита `ansible-playbook` будет использовать все файлы из каталога `inventory/`.

Просмотр фактов об узле

Для просмотра всех доступных фактов об узле выполните команду:

```
ansible <pattern> -m ansible.builtin.setup -i inventory.yml
```

Здесь <pattern> – шаблон для выбора из инвентаря тех узлов, сведения о которых вы хотите получить.

Пример списка фактов об управляемом узле

```
{
  "ansible_facts": {
    "all_ipv4_addresses": ["10.0.2.15", "192.168.56.101"],
    "all_ipv6_addresses": [
      "fe80::a00:27ff:fe0d:82ad",
      "fe80::a00:27ff:feac:2f5c"
    ],
    "ansible_local": {},
    "apparmor": {
      "status": "disabled"
    },
    "architecture": "x86_64",
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
    "bios_date": "12/01/2006",
    "bios_vendor": "innotek GmbH",
    "bios_version": "VirtualBox",
    "board_asset_tag": "NA",
    "board_name": "VirtualBox",
    "board_serial": "NA",
    "board_vendor": "Oracle Corporation",
    "board_version": "1.2",
    "chassis_asset_tag": "NA",
    "chassis_serial": "NA",
    "chassis_vendor": "Oracle Corporation",
    "chassis_version": "NA",
    "cmdline": {
      "BOOT_IMAGE": "/boot/vmlinuz-6.1.50-1-generic",
      "net.ifnames": "0",
      "parsec.max_ilev": "63",
      "quiet": true,
      "ro": true,
      "root": "UUID=17007b23-97d0-46fd-90b6-2c359be9c2b0"
    },
    "date_time": {
      "date": "2024-01-29",
      "day": "29",
      "epoch": "1706532112",
      "epoch_int": "1706532112",
      "hour": "15",
      "iso8601": "2024-01-29T12:41:52Z",
      "iso8601_basic": "20240129T154152867152",
      "iso8601_basic_short": "20240129T154152",
      "iso8601_micro": "2024-01-29T12:41:52.867152Z",
      "minute": "41",
      "month": "01",
      "second": "52",
      "time": "15:41:52",
      "tz": "MSK",
      "tz_dst": "MSK",
      "tz_offset": "+0300",
      "weekday": "Понедельник",
      "weekday_number": "1",
      "weeknumber": "05",
      "year": "2024"
    },
    "default_ipv4": {
      "address": "10.0.2.15",
      "alias": "eth0",
      "broadcast": "10.0.2.255",
      "gateway": "10.0.2.2",
      "interface": "eth0",
      "macaddress": "08:00:27:0d:82:ad",
      "mtu": 1500,
      "netmask": "255.255.255.0",
      "network": "10.0.2.0",
      "prefix": "24",
      "type": "ether"
    },
    "default_ipv6": {},
    "device_links": {
      "ids": {
        "sda": ["ata-VBOX_HARDDISK_VBcf71a1d6-26264a12"],
        "sda1": ["ata-VBOX_HARDDISK_VBcf71a1d6-26264a12-part1"]
      },
      "labels": {},
      "masters": {},
      "uuids": {
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
        "sda1": ["17007b23-97d0-46fd-90b6-2c359be9c2b0"]
      }
    },
    "devices": {
      "loop0": {
        "holders": [],
        "host": "",
        "links": {
          "ids": [],
          "labels": [],
          "masters": [],
          "uuids": []
        },
        "model": null,
        "partitions": {},
        "removable": "0",
        "rotational": "1",
        "sas_address": null,
        "sas_device_handle": null,
        "scheduler_mode": "none",
        "sectors": "0",
        "sectorsize": "512",
        "size": "0.00 Bytes",
        "support_discard": "0",
        "vendor": null,
        "virtual": 1
      },
      "loop1": {
        "holders": [],
        "host": "",
        "links": {
          "ids": [],
          "labels": [],
          "masters": [],
          "uuids": []
        },
        "model": null,
        "partitions": {},
        "removable": "0",
        "rotational": "1",
        "sas_address": null,
        "sas_device_handle": null,
        "scheduler_mode": "none",
        "sectors": "0",
        "sectorsize": "512",
        "size": "0.00 Bytes",
        "support_discard": "0",
        "vendor": null,
        "virtual": 1
      },
      "loop2": {
        "holders": [],
        "host": "",
        "links": {
          "ids": [],
          "labels": [],
          "masters": [],
          "uuids": []
        },
        "model": null,
        "partitions": {},
        "removable": "0",
        "rotational": "1",
        "sas_address": null,
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
        "sas_device_handle": null,
        "scheduler_mode": "none",
        "sectors": "0",
        "sectorsize": "512",
        "size": "0.00 Bytes",
        "support_discard": "0",
        "vendor": null,
        "virtual": 1
      },
      "loop3": {
        "holders": [],
        "host": "",
        "links": {
          "ids": [],
          "labels": [],
          "masters": [],
          "uuids": []
        },
        "model": null,
        "partitions": {},
        "removable": "0",
        "rotational": "1",
        "sas_address": null,
        "sas_device_handle": null,
        "scheduler_mode": "none",
        "sectors": "0",
        "sectorsize": "512",
        "size": "0.00 Bytes",
        "support_discard": "0",
        "vendor": null,
        "virtual": 1
      },
      "loop4": {
        "holders": [],
        "host": "",
        "links": {
          "ids": [],
          "labels": [],
          "masters": [],
          "uuids": []
        },
        "model": null,
        "partitions": {},
        "removable": "0",
        "rotational": "1",
        "sas_address": null,
        "sas_device_handle": null,
        "scheduler_mode": "none",
        "sectors": "0",
        "sectorsize": "512",
        "size": "0.00 Bytes",
        "support_discard": "0",
        "vendor": null,
        "virtual": 1
      },
      "loop5": {
        "holders": [],
        "host": "",
        "links": {
          "ids": [],
          "labels": [],
          "masters": [],
          "uuids": []
        },
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
        "model": null,
        "partitions": {},
        "removable": "0",
        "rotational": "1",
        "sas_address": null,
        "sas_device_handle": null,
        "scheduler_mode": "none",
        "sectors": "0",
        "sectorsize": "512",
        "size": "0.00 Bytes",
        "support_discard": "0",
        "vendor": null,
        "virtual": 1
      },
      "loop6": {
        "holders": [],
        "host": "",
        "links": {
          "ids": [],
          "labels": [],
          "masters": [],
          "uuids": []
        },
        "model": null,
        "partitions": {},
        "removable": "0",
        "rotational": "1",
        "sas_address": null,
        "sas_device_handle": null,
        "scheduler_mode": "none",
        "sectors": "0",
        "sectorsize": "512",
        "size": "0.00 Bytes",
        "support_discard": "0",
        "vendor": null,
        "virtual": 1
      },
      "loop7": {
        "holders": [],
        "host": "",
        "links": {
          "ids": [],
          "labels": [],
          "masters": [],
          "uuids": []
        },
        "model": null,
        "partitions": {},
        "removable": "0",
        "rotational": "1",
        "sas_address": null,
        "sas_device_handle": null,
        "scheduler_mode": "none",
        "sectors": "0",
        "sectorsize": "512",
        "size": "0.00 Bytes",
        "support_discard": "0",
        "vendor": null,
        "virtual": 1
      },
      "sda": {
        "holders": [],
        "host": "SATA controller: Intel Corporation 82801HM/HEM (ICH8M/ICH8M-E) SATA␣
↪Controller [AHCI mode] (rev 02)",
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
        "links": {
          "ids": ["ata-VBOX_HARDDISK_VBcf71a1d6-26264a12"],
          "labels": [],
          "masters": [],
          "uuids": []
        },
        "model": "VBOX HARDDISK",
        "partitions": {
          "sda1": {
            "holders": [],
            "links": {
              "ids": ["ata-VBOX_HARDDISK_VBcf71a1d6-26264a12-part1"],
              "labels": [],
              "masters": [],
              "uuids": ["17007b23-97d0-46fd-90b6-2c359be9c2b0"]
            },
            "sectors": "61435904",
            "sectorsize": 512,
            "size": "29.29 GB",
            "start": "2048",
            "uuid": "17007b23-97d0-46fd-90b6-2c359be9c2b0"
          }
        },
        "removable": "0",
        "rotational": "1",
        "sas_address": null,
        "sas_device_handle": null,
        "scheduler_mode": "mq-deadline",
        "sectors": "61440000",
        "sectorsize": "512",
        "size": "29.30 GB",
        "support_discard": "0",
        "vendor": "ATA",
        "virtual": 1
      }
    },
    "distribution": "Astra Linux",
    "distribution_file_parsed": true,
    "distribution_file_path": "/etc/os-release",
    "distribution_file_variety": "NA",
    "distribution_major_version": "1",
    "distribution_release": "1.7_x86-64",
    "distribution_version": "1.7_x86-64",
    "dns": {
      "domain": "astralinux.ru",
      "nameservers": ["10.0.2.3"],
      "search": ["astralinux.ru"]
    },
    "domain": "",
    "effective_group_id": 1000,
    "effective_user_id": 1000,
    "env": {
      "HOME": "/home/vagrant",
      "LANG": "ru_RU.UTF-8",
      "LC_CTYPE": "C.UTF-8",
      "LOGNAME": "vagrant",
      "PATH": "/usr/local/bin:/usr/bin:/bin:/usr/games",
      "PWD": "/home/vagrant",
      "SHELL": "/bin/bash",
      "SHLVL": "1",
      "SSH_CLIENT": "192.168.56.11 37284 22",
      "SSH_CONNECTION": "192.168.56.11 37284 192.168.56.101 22",
      "SSH_TTY": "/dev/pts/0",
      "TERM": "xterm",
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
    "USER": "vagrant",
    "XDG_RUNTIME_DIR": "/run/user/1000",
    "XDG_SESSION_CLASS": "user",
    "XDG_SESSION_ID": "31",
    "XDG_SESSION_TYPE": "tty",
    "_": "/usr/bin/python3"
  },
  "eth0": {
    "active": true,
    "device": "eth0",
    "ipv4": {
      "address": "10.0.2.15",
      "broadcast": "10.0.2.255",
      "netmask": "255.255.255.0",
      "network": "10.0.2.0",
      "prefix": "24"
    },
    "ipv6": [
      {
        "address": "fe80::a00:27ff:fe0d:82ad",
        "prefix": "64",
        "scope": "link"
      }
    ],
    "macaddress": "08:00:27:0d:82:ad",
    "module": "e1000",
    "mtu": 1500,
    "pciid": "0000:00:03.0",
    "promisc": false,
    "speed": 1000,
    "type": "ether"
  },
  "eth1": {
    "active": true,
    "device": "eth1",
    "ipv4": {
      "address": "192.168.56.101",
      "broadcast": "192.168.56.255",
      "netmask": "255.255.255.0",
      "network": "192.168.56.0",
      "prefix": "24"
    },
    "ipv6": [
      {
        "address": "fe80::a00:27ff:feac:2f5c",
        "prefix": "64",
        "scope": "link"
      }
    ],
    "macaddress": "08:00:27:ac:2f:5c",
    "module": "e1000",
    "mtu": 1500,
    "pciid": "0000:00:08.0",
    "promisc": false,
    "speed": 1000,
    "type": "ether"
  },
  "fibre_channel_wwn": [],
  "fips": false,
  "form_factor": "Other",
  "fqdn": "node-1",
  "gather_subset": ["all"],
  "hostname": "node-1",
  "hostnqn": "",
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
    "interfaces": ["eth0", "eth1", "lo"],
    "is_chroot": false,
    "iscsi_iqn": "",
    "kernel": "6.1.50-1-generic",
    "kernel_version": "#astra2+ci6 SMP PREEMPT_DYNAMIC Fri Oct  6 14:38:42 UTC 2023",
    "lo": {
      "active": true,
      "device": "lo",
      "ipv4": {
        "address": "127.0.0.1",
        "broadcast": "",
        "netmask": "255.0.0.0",
        "network": "127.0.0.0",
        "prefix": "8"
      },
      "ipv6": [
        {
          "address": "::1",
          "prefix": "128",
          "scope": "host"
        }
      ],
      "mtu": 65536,
      "promisc": false,
      "type": "loopback"
    },
    "lsb": {
      "codename": "1.7_x86-64",
      "description": "Astra Linux 1.7 x86-64",
      "id": "AstraLinux",
      "major_release": "1",
      "release": "1.7_x86-64"
    },
    "machine": "x86_64",
    "machine_id": "d383adfed45648cdbd75a02273b61314",
    "memfree_mb": 1639,
    "memory_mb": {
      "nocache": {
        "free": 1805,
        "used": 160
      },
      "real": {
        "free": 1639,
        "total": 1965,
        "used": 326
      },
      "swap": {
        "cached": 0,
        "free": 0,
        "total": 0,
        "used": 0
      }
    },
    "memtotal_mb": 1965,
    "module_setup": true,
    "mounts": [
      {
        "block_available": 6732778,
        "block_size": 4096,
        "block_total": 7514846,
        "block_used": 782068,
        "device": "/dev/sda1",
        "fstype": "ext4",
        "inode_available": 1884545,
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
        "inode_total": 1921360,
        "inode_used": 36815,
        "mount": "/",
        "options": "rw,relatime,errors=remount-ro",
        "size_available": 27577458688,
        "size_total": 30780809216,
        "uuid": "17007b23-97d0-46fd-90b6-2c359be9c2b0"
      }
    ],
    "nodename": "node-1",
    "os_family": "Astra Linux",
    "pkg_mgr": "apt",
    "proc_cmdline": {
      "BOOT_IMAGE": "/boot/vmlinuz-6.1.50-1-generic",
      "net.ifnames": "0",
      "parsec.max_ilev": "63",
      "quiet": true,
      "ro": true,
      "root": "UUID=17007b23-97d0-46fd-90b6-2c359be9c2b0"
    },
    "processor": [
      "0",
      "GenuineIntel",
      "Intel(R) Core(TM) i7-8700K CPU @ 3.70GHz",
      "1",
      "GenuineIntel",
      "Intel(R) Core(TM) i7-8700K CPU @ 3.70GHz"
    ],
    "processor_cores": 2,
    "processor_count": 1,
    "processor_nproc": 2,
    "processor_threads_per_core": 1,
    "processor_vcpus": 2,
    "product_name": "VirtualBox",
    "product_serial": "NA",
    "product_uuid": "NA",
    "product_version": "1.2",
    "python": {
      "executable": "/usr/bin/python3",
      "has_sslcontext": true,
      "type": "cpython",
      "version": {
        "major": 3,
        "micro": 3,
        "minor": 7,
        "releaselevel": "final",
        "serial": 0
      },
      "version_info": [3, 7, 3, "final", 0]
    },
    "python_version": "3.7.3",
    "real_group_id": 1000,
    "real_user_id": 1000,
    "selinux": {
      "status": "disabled"
    },
    "selinux_python_present": true,
    "service_mgr": "systemd",
    "ssh_host_key_ecdsa_public":
↪"AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBJe9D5fI+xDzdcOfpQ/
↪JzxooISS7wskk7RzuX1Qe2eB+yIitXmNBkzRUTvTGsokhQCGPzl8rRlPEOcWCkwA37O4=",
    "ssh_host_key_ecdsa_public_keytype": "ecdsa-sha2-nistp256",
    "ssh_host_key_ed25519_public":
↪"AAAAC3NzaC1lZDI1NTE5AAAAILjVQqxasGBvwxxIFoAV+b37omFihnDLe/AW1XT0raVF",
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
    "ssh_host_key_ed25519_public_keytype": "ssh-ed25519",
    "ssh_host_key_rsa_public":
↪"AAAAB3NzaC1yc2EAAAADAQABAAABgQC5SLc9hewzm7M9prRWaR+v5ds0Qjj41Bzb2rZz0ITwHrEAIcdSXXvlRkq2wQgGxdcVL
↪xEvX2iwJffJ2yyILr+yXlBQehYbqCJhZt7zkF7KyLwC/QESIRG416U5KQuviMStXCbLBOoTo1BmNHNW8jX2NE4/
↪8hY/M94/naC86wJ805/l4Dc2+FKXTua8/
↪yq53Abc8I9RBP7qMPOx7nAcbCmiKxJd1IED2Bj9avwhsYKCdWqBlKMJ3EUccrVYM20Jhp5j/
↪EqTRYWtkpda2XSkFkXTYgU0nmyboav0JDd18PmnxcG3/Jzzm5m/
↪EDM3vulRqWoqIYTCmAh5F32tleJllhBsMHsGOmYWgXUF0zEZob1kkn5/
↪mUVMHxN3lLdxkjY7WRuTXjaplW71fgV6Mq3mJIaHHVuBnpUkwFwWLZdPkLEQyma0lrK5O1W2Uenxmed5Y72vMq2Mvj+X5P3Tng
↪",
    "ssh_host_key_rsa_public_keytype": "ssh-rsa",
    "swapfree_mb": 0,
    "swaptotal_mb": 0,
    "system": "Linux",
    "system_capabilities": [""],
    "system_capabilities_enforced": "True",
    "system_vendor": "innotek GmbH",
    "uptime_seconds": 58762,
    "user_dir": "/home/vagrant",
    "user_gecos": "vagrant,,,",
    "user_gid": 1000,
    "user_id": "vagrant",
    "user_shell": "/bin/bash",
    "user_uid": 1000,
    "userspace_architecture": "x86_64",
    "userspace_bits": "64",
    "virtualization_role": "guest",
    "virtualization_tech_guest": ["virtualbox"],
    "virtualization_tech_host": [],
    "virtualization_type": "virtualbox"
  }
}
```


Динамическая инвентаризация

Динамическая инвентаризация предназначена для автоматического формирования описания инвентаря на основе внешних источников данных (файлы, API, базы данных, облачные сервисы, CMDB (Configuration management database[8]) и другие). Это позволяет управлять инфраструктурой, которая часто изменяется, без необходимости вручную обновлять статические файлы инвентаризации.

Способы динамической инвентаризации

Как отмечалось ранее, при любом способе Ansible формирует динамически в начале выполнения задания внутреннее описание инвентаря. Однако, к *динамической инвентаризации* следует относить процессы, в которых пользователь активно влияет на формирование описания с помощью следующих средств:

- собственные утилиты, формирующие описание в формате JSON, соответствующем требованиям Ansible;
- дополнительные расширения Ansible для инвентаризации, разработанные собственными средствами или предоставляемые другими разработчиками, например расширения для популярных облачных платформ Amazon EC2, Google Cloud, OpenStack и других;
- директивы внутри сценариев автоматизации, изменяющие описание инвентаря, например с помощью модулей `ansible.builtin.add_host` и `ansible.builtin.group_by`.

Описание инвентаря в формате JSON должно иметь структуру, как в следующем примере:

[8] https://en.wikipedia.org/wiki/Configuration_management_database

```
{
  "group1": {
    "hosts": ["host1", "host2"],
    "vars": {
      "group_var1": "value1"
    },
    "children": ["child_group1"]
  },
  "group2": {
    "hosts": ["host3"]
  },
  "_meta": {
    "hostvars": {
      "host1": {
        "host_var1": "valueA"
      },
      "host2": {},
      "host3": {
        "host_var2": "valueB"
      }
    }
  }
}
```

Секции, описывающие группы, узлы и переменные, имеют то же значение, что и в файлах со статическим описанием. Секция `_meta` с метаданными описана далее.

Этапы инвентаризации

Основные этапы динамической инвентаризации представлены следующими шагами:

1. Ansible получает источники описания инвентаря, указанные аргументами `-i`.
2. Ansible обрабатывает каждый источник с помощью соответствующего расширения Ansible или путем прямого запуска исполняемого файла. Результатом обработки каждого источника является описание инвентаря в формате JSON.
3. Ansible анализирует полученные данные и формирует *группы*, *переменные* и *метаданные* для каждого сценария.
4. Ansible использует полученное описание инвентаря для выполнения сценариев автоматизации.

Метаданные

Метаданные в динамической инвентаризации Ansible предназначены для хранения информации об узлах (`hostvars`), которая необходима для выполнения сценариев. Метаданные позволяют избежать дополнительных запросов к источнику данных при обращении к переменным узлов во время выполнения сценариев.

Метаданные выполняют следующие функции:

- упрощают доступ к переменным узлов;
- повышают производительность путем предварительной загрузки всех переменных;
- обеспечивают целостность и актуальность информации о каждом узле.

Структура метаданных

В описании инвентаря в формате JSON, возвращаемом динамическим источником, метаданные размещаются в специальном разделе `_meta`. Этот раздел содержит вложенный элемент `hostvars`, который представляет собой словарь, где ключ — название узла, а значение — словарь переменных для этого узла.

Пример структуры метаданных:

```
{
  "_meta": {
    "hostvars": {
      "web01.example.com": {
        "ansible_host": "192.168.56.11",
        "ansible_user": "admin"
      },
      "db01.example.com": {
        "ansible_host": "192.168.56.21",
        "ansible_user": "root"
      }
    }
  }
}
```

Элементы структуры метаданных

Формально структура имеет следующий общий вид:

```
{
  "_meta": {
    "hostvars": {
      "<host_name>": {
        "<variable>": "<value>"
      }
    }
  }
}
```

Ключи словарей указывают на следующие данные:

- `hostvars` – словарь переменных для всех узлов инвентаря;
- <host_name> – название конкретного узла, например `web01.example.com`;
- <variable> – переменная, определяющая параметр подключения или поведения для данного узла, например `ansible_host` или `ansible_user`;
- <value> – значение переменной.

Примечание

Словарь `_meta` и вложенный словарь `hostvars` позволяют Ansible получать все необходимые переменные для каждого узла единовременно, что ускоряет выполнение сценариев и снижает нагрузку на внешний источник данных.

Разработка собственных способов

Собственные способы должны формировать описание в формате JSON, представленном ранее. К ним относятся следующие:

- разработка собственного расширения Ansible на языке Python – расширение инвентаризации (Ansible inventory plugin);
- разработка собственного (также называемого «внешнего») исполняемого скрипта или бинарного файла динамической инвентаризации, который можно использовать как и другие файлы описания инвентаря.

Первый из представленных способов является предпочтительным.

Разработка расширения Ansible для динамической инвентаризации

Все расширения Ansible, включая рассматриваемый тип, разрабатываются на языке Python.

Рекомендации

Следующие рекомендации являются основными при разработке расширения:

- Реализуйте базовый класс `BaseInventoryPlugin` и определите требуемые методы `verify_file` и `parse`.
- При необходимости реализуйте классы `Cacheable` для кеширования данных и `Constructable` для динамического формирования групп узлов.
- Документируйте все параметры расширения и поддерживаемые форматы источников.
- Используйте стандартные структуры данных Ansible для описания групп, узлов и переменных.

Структура расширения

Основные функции инвентаризации реализуются через класс `BaseInventoryPlugin`. Основные методы класса:

- `verify_file(path)` проверяет, может ли расширение обработать указанный источник.
- `parse(inventory, loader, path, cache=True)` обрабатывает источники данных и формирует структуру инвентаря.

Внутри метода `parse` необходимо реализовать логику получения данных, создания групп, добавления узлов и переменных. В нем же, при необходимости, можно реализовать кеширование и динамическое формирование групп через шаблоны Jinja2.

Пример базовой структуры расширения без подробностей:

```
from ansible.plugins.inventory import BaseInventoryPlugin

class InventoryModule(BaseInventoryPlugin):
    NAME = 'myplugin'

    def verify_file(self, path):
        # Validate source files
        return path.endswith('.myplugin.yml')

    def parse(self, inventory, loader, path, cache=True):
        # Main parsing and inventory creation
        config = self._read_config_data(path)
        # Adding groups, nodes, and variables
        # self.inventory.add_host(...)
        # self.inventory.set_variable(...)
```

Разработка утилиты динамической инвентаризации

В качестве альтернативы расширениям можно использовать внешнюю утилиту (исполняемый скрипт или бинарный файл), которая возвращает описание инвентаря в формате JSON. Утилита должна принимать два аргумента:

- `--list`– для вывода полной структуры инвентаря;
- `--host <host_name>` – для вывода переменных по конкретному узлу.

Следующая команда выполняет простую проверку скрипта:

```
ansible-inventory -i ./my_inventory.py --list
```

Примечание

Для повышения производительности рекомендуется, чтобы утилита возвращала секцию `_meta` с переменными всех узлов (`hostvars`) при вызове с параметром `--list`. В противном случае Ansible будет вызывать утилиту для каждого узла отдельно, используя аргумент `--host`.

Для Ansible ссылку на исполняемый файл (утилиту) следует задавать как и на файлы со статической инвентаризацией, то есть используя аргумент `-i`. Если скрипт обрабатывает исходное статическое описание инвентаря, заданного в других файлах, то в командной строке его необходимо указывать после указания на эти файлы.

Дополнительные требования:

- Файл должен быть обязательно отмечен как исполняемый, то есть иметь атрибут x, назначаемый командой вида:

```
chmod +x inventory.py
```

- В файле следует указать строку shebang[9], например для Python это будет `#!/usr/bin/env python3` или `#!/usr/bin/python3`.

Примеры

Следующие примеры демонстрируют создание и применение динамической инвентаризации.

Использование расширения constructed

Ansible использует расширение `ansible.builtin.constructed` для динамического описания инвентаря встроенными средствами.

Для примера рассмотрим два файла с исходным статическим описанием инвентаря:

```
# inventory-staging.ym
---
all:
  hosts:
    web01.staging.example.com:
      environment: staging
      public_ip_address: 192.168.56.11
      role: web
    db01.staging.example.com:
      environment: staging
      private_ip_address: 10.0.0.21
      role: db
```

```
# inventory-production.yml
---
all:
  hosts:
    web01.prod.example.com:
      environment: production
      public_ip_address: 192.168.57.11
      role: web
    db01.prod.example.com:
      environment: production
      private_ip_address: 10.0.1.21
      role: db
```

В файле `inventory-constructed.yml` используется расширение `constructed` для объединения узлов в группы по заданным признакам:

[9] https://en.wikipedia.org/wiki/Shebang_(Unix)

```
# inventory-constructed.yml
---
plugin: ansible.builtin.constructed
compose:
  ansible_host: public_ip_address | default(private_ip_address)
groups:
  staging: environment == 'staging'
  production: environment == 'production'
  webservers: role == 'web'
  dbservers: role == 'db'
```

Следующая команда проверяет результат динамической инвентаризации:

```
ansible-inventory -i inventory-staging.yml -i inventory-production.yml -i inventory-
↪constructed.yml --list
```

Примечание

Файл, в котором используется расширение `constructed`, должен быть последним в списке файлов инвентаря.

Полученное описание инвентаря имеет следующий вид:

Динамическое описание

```
{
  "_meta": {
    "hostvars": {
      "db01.prod.example.com": {
        "ansible_host": "10.0.1.21",
        "environment": "production",
        "private_ip_address": "10.0.1.21",
        "role": "db"
      },
      "db01.staging.example.com": {
        "ansible_host": "10.0.0.21",
        "environment": "staging",
        "private_ip_address": "10.0.0.21",
        "role": "db"
      },
      "web01.prod.example.com": {
        "ansible_host": "192.168.57.11",
        "environment": "production",
        "public_ip_address": "192.168.57.11",
        "role": "web"
      },
      "web01.staging.example.com": {
        "ansible_host": "192.168.56.11",
        "environment": "staging",
        "public_ip_address": "192.168.56.11",
        "role": "web"
      }
    }
  },
  "all": {
    "children": [
      "ungrouped",
      "staging",
      "webservers",
      "dbservers",
      "production"
    ]
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  },
  "dbservers": {
    "hosts": ["db01.staging.example.com", "db01.prod.example.com"]
  },
  "production": {
    "hosts": ["web01.prod.example.com", "db01.prod.example.com"]
  },
  "staging": {
    "hosts": ["web01.staging.example.com", "db01.staging.example.com"]
  },
  "webservers": {
    "hosts": ["web01.staging.example.com", "web01.prod.example.com"]
  }
}
```

Внешний скрипт для объединения описаний инвентаря

Приведенный далее скрипт формирует описание инвентаря в формате JSON в соответствии с требованиями Ansible. В скрипте определены методы для требуемых параметров:

- `--list` – вывод полного инвентаря;
- `--host <host_name>` – вывод переменных для конкретного узла.

Скрипт Python для динамического описания

```
#!/usr/bin/env python3

import argparse
import json

# Пример хранилища узлов
store = {
    "web01.example.com": {"ansible_host": "192.168.56.11", "ansible_user": "admin"},
    "db01.example.com": {"ansible_host": "192.168.56.21", "ansible_user": "root"},
}


def get_host_vars(host):
    return store.get(host, {})


def get_inventory():
    return {
        "_meta": {"hostvars": {host: vars for host, vars in store.items()}},
        "all": {"hosts": list(store.keys())},
    }


if __name__ == "__main__":
    parser = argparse.ArgumentParser()
    parser.add_argument("--list", action="store_true", help="Показать весь инвентарь")
    parser.add_argument("--host", help="Показать переменные для указанного узла")
    args = parser.parse_args()

    if args.list:
        print(json.dumps(get_inventory(), indent=2))
    elif args.host:
        print(json.dumps(get_host_vars(args.host), indent=2))
    else:
        parser.print_help()
```

Подготовка и проверка скрипта:

1. Сделайте файл исполняемым:

```
chmod +x inventory-script.py
```

2. Проверьте формирование списка узлов:

```
ansible-inventory -i ./inventory-script.py --list
```

3. Проверьте получение переменных конкретного узла:

```
./inventory-script.py --host web01.example.com
```

Внешний скрипт для выборки описания инвентаря из базы данных

Следующий пример показывает как извлечь исходные данные об инвентаре из базы данных MySQL и сформировать описание в формате JSON:

Скрипт Python для динамического описания инвентаря, используя базу данных

```
#!/usr/bin/env python3

import argparse
import json
import mysql.connector
from mysql.connector import Error


class MySQLInventory:
    def __init__(self):
        self.connection = None
        self.inventory = {"_meta": {"hostvars": {}}}

    def connect_to_database(self):
        """Подключение к MySQL"""
        try:
            self.connection = mysql.connector.connect(
                host="localhost",
                database="inventory",
                user="ansible_user",
                password="your_password",
            )
        except Error as e:
            print(f"Ошибка подключения к MySQL: {e}")
            return False
        return True

    def get_inventory_from_db(self):
        """Получить инвентарь из базы данных"""
        if not self.connection:
            return self.inventory

        try:
            cursor = self.connection.cursor(dictionary=True)

            # Получаем хосты
            cursor.execute(
                """
                SELECT hostname, ip_address, ansible_user,
                       environment, role, ssh_port
                FROM servers
                WHERE active = 1
            """
            )

            servers = cursor.fetchall()
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
            # Создаем группы
            groups = {}

            for server in servers:
                hostname = server["hostname"]

                # Добавляем переменные хоста
                self.inventory["_meta"]["hostvars"][hostname] = {
                    "ansible_host": server["ip_address"],
                    "ansible_user": server["ansible_user"],
                    "ansible_port": server["ssh_port"] or 22,
                    "environment": server["environment"],
                    "role": server["role"],
                }

                # Группируем по окружениям
                env_group = f"env_{server['environment']}"
                if env_group not in groups:
                    groups[env_group] = {"hosts": [], "vars": {}}
                groups[env_group]["hosts"].append(hostname)

                # Группируем по ролям
                role_group = f"role_{server['role']}"
                if role_group not in groups:
                    groups[role_group] = {"hosts": [], "vars": {}}
                groups[role_group]["hosts"].append(hostname)

            # Добавляем группы в инвентарь
            self.inventory.update(groups)

        except Error as e:
            print(f"Ошибка выполнения запроса: {e}")
        finally:
            if self.connection.is_connected():
                cursor.close()
                self.connection.close()

        return self.inventory

    def get_host_vars(self, hostname):
        """Получить переменные для конкретного хоста"""
        inventory = self.get_inventory_from_db()
        return inventory["_meta"]["hostvars"].get(hostname, {})


def main():
    parser = argparse.ArgumentParser()
    parser.add_argument("--list", action="store_true")
    parser.add_argument("--host", action="store")

    args = parser.parse_args()

    mysql_inventory = MySQLInventory()

    if args.list:
        if mysql_inventory.connect_to_database():
            inventory = mysql_inventory.get_inventory_from_db()
            print(json.dumps(inventory, indent=2))
        else:
            print(json.dumps({"_meta": {"hostvars": {}}}))
    elif args.host:
        if mysql_inventory.connect_to_database():
            host_vars = mysql_inventory.get_host_vars(args.host)
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
            print(json.dumps(host_vars, indent=2))
        else:
            print(json.dumps({}))


if __name__ == "__main__":
    main()
```

Использование расширения ansible.builtin.script

Еще одним способом интеграции исполняемого файла динамической инвентаризации является использование расширения ansible.builtin.script. Для этого необходимо создать промежуточный файл в формате YAML (inventory-plugin.yml в примере) и в нем с помощью этого расширения указать путь к исполняемому файлу.

Список 1: inventory-plugin.yml

```
---
plugin: ansible.builtin.script
path: ./inventory-script.py
```

Проверка динамической инвентаризации:

```
ansible-inventory -i inventory-script.py -i inventory-plugin.yml --list
```

Динамическая инвентаризация в сценариях

С помощью определенных модулей, например ansible.builtin.add_host и ansible.builtin.group_by, можно изменять описание инвентаря в процессе выполнения сценариев, как в приведенном здесь примере.

Первый сценарий в playbook-dyn-inv.yml вносит изменения в описание инвентаря путем добавления узла и формирования новых групп:

```
---
- name: Add and group hosts dynamically
  hosts: localhost
  gather_facts: false
  tasks:
    - name: Add a new host to the inventory
      ansible.builtin.add_host:
        name: newhost.example.com
        groups: dynamic_group
        ansible_host: "192.168.56.100"
        os_family: Debian

    - name: Group hosts by OS family
      ansible.builtin.group_by:
        key: "os_family_{{ hostvars['inventory_hostname'].os_family | default('unknown')␣
↪}}"

- name: Operate on dynamically grouped hosts
  hosts: os_family_Debian
  gather_facts: false
  tasks:
    - name: Show message
      debug:
        msg: This host is in the Debian OS family group.
```

Изменения, внесенные в одном сценарии (play) сохраняются в описании инвентаря, хранящемся в оперативной памяти (in-memory inventory). Таким образом, они доступны для всех загруженных сценариев, исполняемых после упомянутого.

4.4 Набор сценариев

Набор сценариев (playbook) состоит из сценариев (play). Сценарии используются для настройки управляемых узлов. Инвентарный список для связывания с набором сценариев может быть задан несколькими способами:

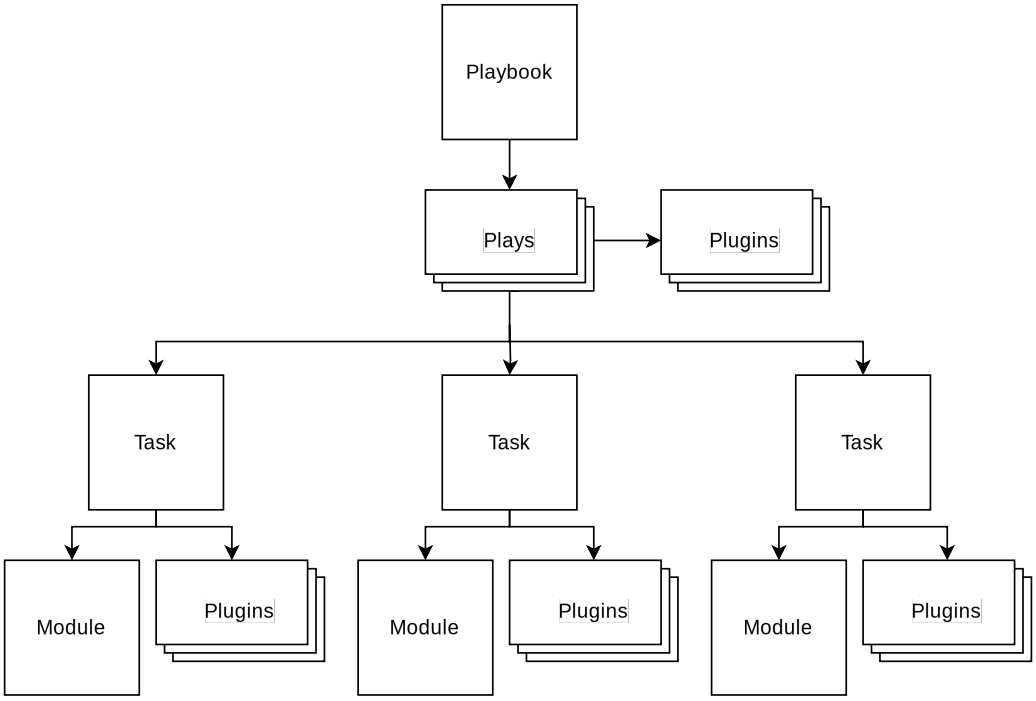
- переменная окружения `ANSIBLE_INVENTORY`;
- аргумент `-i` (`--inventory`, `--inventory-file`) утилиты `ansible-playbook`;
- переменная `inventory` в конфигурационном файле `ansible.cfg`.

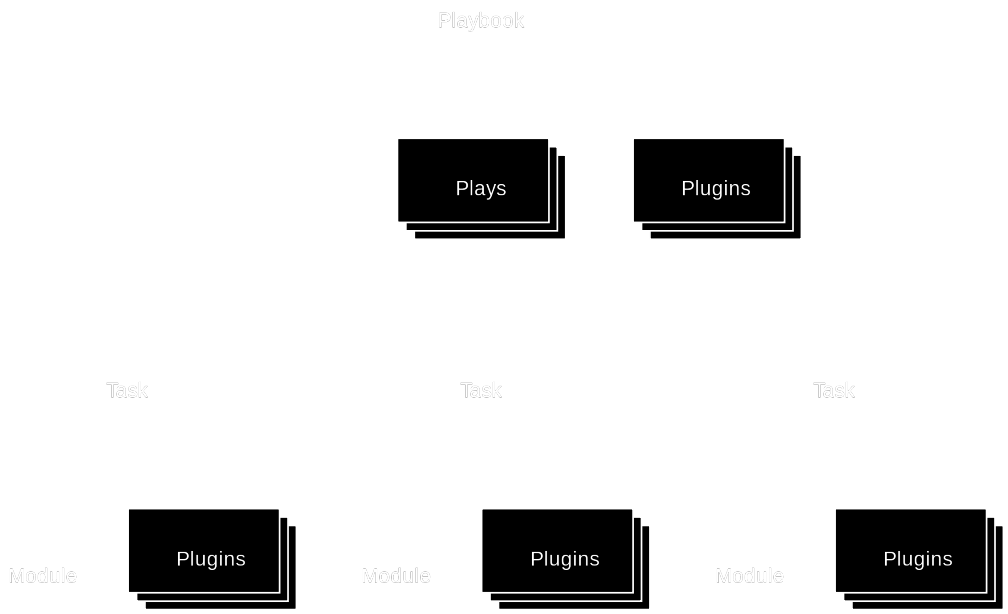
Для выполнения набора сценариев используйте утилиту `ansible-navigator` из состава CDK или `ansible-playbook` из Ansible Core.

В набор сценариев входят следующие компоненты:

- сценарий (play);
- задача (task);
- модуль (module);
- расширение (plugin).

Иерархия исполняемых компонентов в наборе сценариев представлена на схеме:





4.4.1 Сценарий

Сценарий (play) состоит из множества задач, выполняемых для указанных узлов и групп узлов из определенного инвентарного списка.

4.4.2 Задача

Задача (task) использует один из модулей Ansible для выполнения отдельной операции по управлению узлами.

4.4.3 Модуль

Модуль (module) реализует требуемую функциональность. Например, существуют модули для копирования файлов, управления пакетами и службами и так далее. Код модулей выполняется на целевых узлах, которые в большинстве случаев и являются управляемыми.

4.4.4 Расширение

Расширение (plugin) расширяет функциональность Ansible. В отличие от модулей, код расширений выполняется на управляющем узле.

4.5 Модуль

Модуль – это исполняемый файл, который можно использовать из командной строки или в задаче сценария. Ansible выполняет каждый модуль на целевых узлах сценария и собирает возвращаемые значения.

4.5.1 Основные особенности модулей

Модули имеют следующие особенности:

- возвращают данные в формате JSON;
- могут быть написаны на любом языке программирования, но чаще всего используется Python;
- подавляющее большинство модулей удовлетворяют принципу идемпотентности (idempotence), то есть выполняют изменения только при необходимости, чтобы избежать ненужных действий.

Некоторые модули, например, *ansible.builtin.shell* и *ansible.builtin.command*, неидемпотентны.

4.5.2 Запуск модулей через командную строку

Для запуска модулей используйте утилиту `ansible` со следующими аргументами:

- `-m` – название модуля;
- `-a` – параметры вызова модуля.

Например, для запуска службы `httpd` и проверки ее статуса на узлах из группы `webservers` выполните команду:

```
ansible webservers -m ansible.builtin.service -a "name=httpd state=started"
```

4.5.3 Использование модулей в сценариях

В сценариях модули вызываются внутри блока `tasks`. Для этого можно использовать несколько способов:

- Простой синтаксис:

```
- name: Reboot the servers
  ansible.builtin.command: /sbin/reboot -t now
```

- Синтаксис YAML (complex args):

```
- name: Restart webserver
  ansible.builtin.service:
    name: httpd
    state: restarted
```

Синтаксис YAML подходит для сложных задач, когда передается много аргументов.

4.5.4 Просмотр документации по модулям

Для просмотра документации о модуле используйте утилиту `ansible-doc`.

Например, для просмотра информации о модуле *ansible.builtin.ping* используйте команду:

```
ansible-doc ansible.builtin.ping
```

Чтобы получить список всех доступных модулей, используйте команду:

```
ansible-doc -l
```

4.5.5 Исключение модулей

Вы можете исключить определенные модули Ansible из использования, добавив их в список запрета (reject list). Для этого создайте файл конфигурации в формате YAML. По умолчанию он должен быть расположен по пути `/etc/ansible/plugin_filters.yml`.

Предупреждение

Не добавляйте в список запрета модуль `ansible.builtin.stat`! Он необходим для работы Ansible.

Пример файла со списком запретов:

```
filter_version: '1.0'
module_rejectlist:
  - docker
  - easy_install
```

Здесь:

- `filter_version` – версия конфигурационного файла. Этот параметр позволяет обновлять формат конфигурации в будущем, сохраняя обратную совместимость.
- `module_rejectlist` – список модулей для блокировки.

Если необходимо использовать другой путь к конфигурационному файлу со списком запретов, укажите его с одним из следующих способов:

- Переменная окружения `PLUGIN_FILTERS_CFG`.
- Конфигурационный файл `ansible.cfg`:

```
[defaults]
plugin_filters_cfg = /path/to/reject/list
```

4.6 Коллекции Ansible

Коллекцией (collection) называется распространяемый как единое целое набор компонентов, расширяющих возможности Ansible.

В состав коллекции могут входить следующие компоненты:

- роли (roles);
- расширения (plugins);
- модули (modules);
- наборы сценариев (playbooks).

4.6.1 Структура коллекции

Пример команды для создания структуры каталогов коллекции:

```
ansible-galaxy collection init my_space.my_collection
```

Эта команда создаст в текущем каталоге подкаталог `my_space.my_collection/` и разместит в нем файлы коллекции `my_space.my_collection`.

В результате выполнения команды будет создана структура файлов и каталогов:

```
my_space/my_collection/
├── docs/
├── galaxy.yml
├── meta/
│   └── runtime.yml
├── plugins/
│   └── README.md
├── README.md
└── roles/
```

В составе коллекции могут быть следующие файлы и каталоги:

- `meta/`

  Каталог с файлами, содержащими служебную информацию о коллекции, используемую утилитой `ansible-galaxy`.

- `plugins/`

  Опциональный каталог, содержащий расширения, необходимые для работы коллекции.

- `roles/`

  Каталог с подкаталогами ролей. Подробности приведены в *описании ролей*.

- `tests/`

  Каталог с файлами автоматических тестов для коллекции.

- `CHANGELOG.md`

  Файл с историей изменений между версиями.

- `LICENSE`

  Файл лицензии, устанавливающей правила использования и распространения коллекции.

- `README.md`

  Описание коллекции, включающее в себя:

  - предназначение коллекции;
  - требования к окружению;
  - список поддерживаемых операционных систем;
  - порядок описания коллекции в файле зависимостей Ansible и способ установки;
  - список ролей;
  - порядок запуска автоматических тестов;
  - тип лицензии, под которой распространяется код коллекции;
  - информация об авторских правах.

- `galaxy.yml`

  Сведения о коллекции, используемые утилитой `ansible-galaxy`, включая номер ее актуальной версии.

Список 2: Пример заполнения `galaxy.yml`

```
---
namespace: astra
name: ald_pro
version: 1.0.0
description: Collection for ALD Pro deployment
readme: README.md
authors:
  - LLC "RusBITech-Astra"
dependencies:
  "freeipa.ansible_freeipa": "1.10.0"
  "ansible.utils": "3.0.0"
tags:
  - astra
  - ald_pro
repository: https://hub.astra-automation.ru/aa-gca/ARFA/ald_pro
documentation: https://hub.astra-automation.ru/aa-gca/ARFA/ald_pro/-/blob/master/
↪README.md
```

Здесь:

- `astra` – название пространства имен;
- `ald_pro` – название коллекции;
- `version` – номер версии коллекции;
- `description` – краткое описание коллекции;
- `readme` – путь к файлу README, содержащему подробное описание коллекции;
- `authors` – список авторов коллекции;
- `dependencies` – зависимости, необходимые для использования коллекции;
- `tags` – список тегов, по которым можно найти коллекцию среди множества других;

  - `repository` – ссылка на репозиторий с исходным кодом коллекции;
  - `documentation` – ссылка на документацию коллекции.

- `requirements_ansible.yml`

  Зависимости Ansible, необходимые для использования коллекции.

Особенности коллекций, доступных на портале Automation Hub, приведены в *описании Automation Hub*.

4.6.2 Роль

Роли используют для группирования задач по развертыванию и настройке сложных систем.

Пример создания структуры каталогов для роли:

```
ansible-galaxy init httpd
```

Здесь `httpd` – название каталога, в котором будет создана структура роли.

В результате выполнения команды будет создана следующая структура файлов и каталогов:

```
httpd/
├── defaults/
│   └── main.yml
├── files/
├── handlers/
│   └── main.yml
├── meta/
│   └── main.yml
├── README.md
├── tasks/
│   └── main.yml
├── templates/
├── tests/
│   ├── inventory
│   └── test.yml
└── vars/
    └── main.yml
```

В составе роли могут быть следующие файлы и каталоги:

- `defaults/`

  Каталог с файлами, содержащими значения по умолчанию для переменных роли.

- `files/`

  Каталог с вспомогательными файлами, используемыми ролью, например, шаблоны HTML-страниц, изображения и так далее. При выполнении роли файлы из этого каталога копируются на управляемые узлы.

- `handlers/`

  Каталог с файлами обработчиков, выполняемых при использовании роли.

- `meta/`

  Каталог с файлами, содержащими служебную информацию о роли, используемую утилитой `ansible-galaxy`.

- `tasks/`

  Каталог с файлами, выполняющимися при использовании роли.

- `templates/`

  Каталог с шаблонами формата Jinja 2.

- `vars/`

  Каталог с файлами, содержащими список переменных роли. Каждая роль содержит список переменных, позволяющих управлять настройками соответствующего ПО.

- `LICENSE`

  Файл лицензии, под которой распространяется код роли.

- `README.md`

  Описание роли, включающее в себя:

  - Предназначение роли.
  - Требования к окружению.
  - Список переменных роли.

    Переменные роли делятся на обязательные и опциональные. Для опциональных переменных, значения которых не заданы, будут использованы значения по умолчанию.

    Предупреждение

    Значения обязательных переменных должны быть явно заданы в наборе сценариев или используемом им файле с переменными.

  - Примеры использования.
  - Порядок запуска автоматических тестов.
  - Тип лицензии, под которой распространяется код роли.
  - Информация об авторских правах.
  - Список поддерживаемых операционных систем.

Сценарий может включать список подключаемых ролей как в следующем примере:

```
- hosts: server.example.com
  become: true
  gather_facts: true
  roles:
    - astra.postgresql.postgresql
    - astra.rubackup.rubackup_server
```

4.6.3 Загрузка коллекции

По умолчанию утилита `ansible-galaxy` использует https://galaxy.ansible.com/ в качестве реестра коллекций.

Чтобы загрузить коллекцию, используйте следующую команду:

```
ansible-galaxy collection download my_namespace.my_collection
```

Здесь:

- `my_namespace` – название пространства имен;
- `my_collection` – название коллекции.

В результате выполнения этой команды утилита `ansible-galaxy` загрузит указанную коллекцию и ее зависимости и создаст файл `requirements.yml`, который можно использовать для установки этой коллекции на узлах без доступа к серверу Galaxy.

По умолчанию все коллекции загружаются в каталог `collections/`, который создается в текущем каталоге. Чтобы загрузить коллекции в другой каталог, укажите путь к нему в значении аргумента `-p`.

Если необходимо загрузить определенную версию коллекции, выполните следующую команду:

```
ansible-galaxy collection download my_namespace.my_collection:1.0.0
```

Здесь `1.0.0` – версия коллекции.

Если необходимо установить несколько коллекций, воспользуйтесь одним из следующих способов:

- перечислите необходимые коллекции в командной строке, например:

```
ansible-galaxy collection download my_namespace1.my_collection1:1.0.0 my_namespace2.
↪my_collection2
```

- укажите необходимые коллекции файле `requirements.yml`, например:

```
---
collections:
  - name: my_namespace1.my_collection1
  - name: my_namespace2.my_collection2
    version: ">=1.2.0"
```

4.6.4 Установка

Чтобы установить коллекцию, используйте следующую команду:

```
ansible-galaxy collection install my_namespace.my_collection
```

По умолчанию все коллекции устанавливаются в домашнем каталоге пользователя, конкретно в `~/.ansible/collections/ansible_collections/`.

Локальная установка

Если вы загрузили архив коллекции, для ее установки используйте следующую команду:

```
ansible-galaxy collection install my_namespace-my_collection-1.0.0.tar.gz -p ./
↪collections
```

Здесь `-p ./collections` – путь к каталогу для установки коллекции.

Если коллекция разрабатывается локально, чтобы установить ее, используйте следующую команду:

```
ansible-galaxy collection install /path/to/collection -p ./collections
```

Здесь `/path/to/collection` – путь к каталогу, который содержит файлы разрабатываемой коллекции.

В результате выполнения команды Ansible соберет коллекцию на основе файлов `MANIFEST.json` или `galaxy.yml`.

Если в одном каталоге находятся несколько коллекций, они могут быть установлены одной командой, например:

```
ansible-galaxy collection install /path/to/directory -p ./collections
```

Здесь `/path/to/directory` – путь к каталогу, коллекции из которого необходимо установить.

Установка конкретной версии коллекции

По умолчанию утилита `ansible-galaxy` устанавливает последнюю доступную версию коллекции.

Чтобы установить определенную версию коллекции, укажите ее после названия коллекции с использованием идентификатора диапазона ==. Например, чтобы установить версию 1.0.1 коллекции my_namespace.my_collection, используйте следующую команду:

```
ansible-galaxy collection install my_namespace.my_collection:==1.0.1
```

Если вы хотите установить версию в определенном диапазоне, используйте несколько идентификаторов версий, разделенных запятой. Например, чтобы установить версию, которая больше или равна 1.0.0 и меньше 2.0.0, используйте команду:

```
ansible-galaxy collection install 'my_namespace.my_collection:>=1.0.0,<2.0.0'
```

Примечание

По умолчанию утилита ansible-galaxy игнорирует предварительные версии, например, бета-версии. Если необходимо установить такую версию, укажите ее с помощью идентификатора диапазона ==, например:

```
ansible-galaxy collection install my_namespace.my_collection:==1.0.0-beta.1
```

Доступные идентификаторы диапазона:

- * – самая последняя версия (по умолчанию);
- != – не равна указанной версии;
- == – равна указанной версии;
- >= – больше или равна указанной версии;
- > – больше указанной версии;
- <= – меньше или равна указанной версии;
- < – меньше указанной версии.

Установка с использованием файла requirements.yml

Файл requirements.yml позволяет устанавливать несколько коллекций.

Для каждой коллекции можно указать следующую информацию:

- name – название;
- version – версия (можно указать диапазон версий);
- source – URL источника, из которого будет загружена коллекция;
- type – тип источника:
  - dir – локальный каталог;
  - file – локальный файл в формате tar.gz;
  - galaxy – реестр коллекций;
  - git – репозиторий Git;
  - subdirs – подкаталог внутри каталога, указанного в поле source;
  - url – URL-адрес.

Например, для установки коллекции из репозитория Git, добавьте в requirements.yml следующие данные:

```
collections:
  - name: https://github.com/organization/repo_name.git
    type: git
```

В файле requirements.yml можно указать роли, используя ключ roles, например:

```
roles:
  - name: geerlingguy.java
    version: "1.9.6"

collections:
  - name: geerlingguy.php_roles
    version: ">=0.9.3"
    source: https://galaxy.ansible.com
```

Чтобы установить роли и коллекции, указанные в файле `requirements.yml`, используйте следующую команду:

```
ansible-galaxy install -r requirements.yml
```

Если необходимо установить только коллекции, используйте следующую команду:

```
ansible-galaxy collection install -r requirements.yml
```

Если необходимо установить только роли, используйте следующую команду:

```
ansible-galaxy role install -r requirements.yml
```

Важно

Если в аргументах указан путь к целевому каталогу для установки, `ansible-galaxy` установит только роли и пропустит коллекции.

Установка из репозитория Git

Установка коллекций из репозитория Git подходит для тестирования и работы с неофициальными версиями.

Чтобы установить коллекцию из репозитория Git с помощью командной строки, вместо названия коллекции используйте URI репозитория. Перед URI необходимо добавить префикс `git+`, если вы не используете SSH-аутентификацию. Вы можете указать ветку, коммит или тег с помощью синтаксиса, где нужное значение отделяется запятой от URI репозитория.

Например, если необходимо установить коллекцию из репозитория, используя последнюю версию в ветке `devel`, используйте следующую команду:

```
ansible-galaxy collection install git+https://github.com/organization/repo_name.git,devel
```

Если необходимо установить коллекцию из приватного репозитория GitHub, используйте следующую команду:

```
ansible-galaxy collection install git@github.com:organization/repo_name.git
```

Если необходимо установить коллекцию из локального репозитория Git, используйте следующую команду:

```
ansible-galaxy collection install git+file:///home/user/path/to/repo_name.git
```

Важно

Встраивание учетных данных в URI небезопасно. Используйте надежные варианты аутентификации, чтобы ваши учетные данные не были раскрыты в журналах или где-либо еще.

При установке коллекции из репозитория Ansible использует метаданные коллекции, содержащиеся в файлах `galaxy.yml` или `MANIFEST.json`, чтобы корректно собрать коллекцию.

По умолчанию Ansible ищет файлы метаданных в двух местах репозитория:

- в корневом каталоге;
- в каждом каталоге на один уровень ниже от корня.

Если в корневом каталоге репозитория существует файл `galaxy.yml` или `MANIFEST.json`, Ansible использует метаданные коллекции из этого файла для установки отдельной коллекции.

Пример структуры репозитория с одной коллекцией:

```
├── galaxy.yml
├── plugins/
│   ├── lookup/
│   ├── modules/
│   └── module_utils/
└── README.md
```

Если в одном или нескольких каталогах по пути к репозиторию (на один уровень ниже) существует файл `galaxy.yml` или `MANIFEST.json`, Ansible устанавливает каждый каталог с файлом метаданных как коллекцию.

Пример репозитория с двумя коллекциями:

```
├── collection1/
│   ├── docs/
│   ├── galaxy.yml
│   └── plugins/
│       ├── inventory/
│       └── modules/
└── collection2/
    ├── docs/
    ├── galaxy.yml
    ├── plugins/
    │   ├── filter/
    │   └── modules/
    └── roles/
```

Если структура репозитория отличается, или вам нужна только определенная коллекция, можно указать путь к каталогу с метаданными через символ # в URI репозитория. Например, чтобы установить только `collection2` из примера выше, используйте команду:

```
ansible-galaxy collection install git+https://github.com/organization/repo_name.git#/
↪collection2
```

4.6.5 Получение списка коллекций

Чтобы получить список установленных коллекций, выполните следующую команду:

```
ansible-galaxy collection list
```

Запустите команду с аргументом `-vvv`, чтобы отобразить более подробную информацию о коллекциях.

Чтобы вывод команды содержал данные только об определенной коллекции, укажите ее полное название, например:

```
ansible-galaxy collection list my_namespace.my_collection
```

Чтобы искать коллекции, размещенные вне каталога по умолчанию, используйте аргумент `-p`. Если необходимо указать несколько путей, разделите их с помощью `:`, например:

```
ansible-galaxy collection list -p '/opt/ansible/collections:/etc/ansible/collections'
```

4.6.6 Проверка коллекции

После установки вы можете проверить, соответствует ли содержимое установленной коллекции содержимому коллекции на сервере. Эта функция предполагает, что коллекция установлена по одному из настроенных путей и существует на одном из настроенных серверов Galaxy.

Чтобы выполнить проверку, используйте следующую команду:

```
ansible-galaxy collection verify my_namespace.my_collection
```

Если команда проверки коллекции выполнена успешно, то в выводе не будет никаких сообщений. Если коллекция была изменена, измененные файлы будут перечислены под названием коллекции.

Пример вывода:

```
Collection my_namespace.my_collection contains modified content in the following files:
my_namespace.my_collection
   plugins/inventory/my_inventory.py
   plugins/modules/my_module.py
```

Используйте аргумент `-vvv` для отображения дополнительной информации, такой как версия и путь к установленной коллекции, URL-адрес удаленной коллекции, используемой для проверки, и результат успешной проверки.

Если у вас установлена не самая последняя версия коллекции, вам следует указать конкретную версию для проверки. Если версия не указана, установленная коллекция проверяется на соответствие последней версии, доступной на сервере.

Также можно указать коллекции для проверки в файле `requirements.yml`. Зависимости в этом файле не включаются в процесс проверки и должны быть проверены отдельно с помощью команды:

```
ansible-galaxy collection verify -r requirements.yml
```

Проверка файлов формата `tar.gz` не поддерживается. Если `requirements.yml` содержит пути к таким файлам или URL для установки, можно использовать аргумент `--ignore-errors`, чтобы гарантировать обработку всех коллекций.

4.6.7 Удаление коллекции

Если вам больше не нужна коллекция, удалите каталог с ней из своей файловой системы:

```
rm -rf ~/.ansible/collections/ansible_collections/my_namespace/my_collection
```

4.6.8 Использование коллекций

После установки вы можете ссылаться на содержимое коллекции по полному квалифицированному имени (*FQCN*). FQCN состоит из следующих компонентов:

- namespace – пространство имен;
- collection name – название коллекции;
- resource name – название ресурса, например, модуля, роли, набора сценариев или любого другого контента внутри коллекции.

Пример FQCN:

```
astra.ald_pro.controller
```

Использование ключевого слова

Ключевое слово `collections` позволяет определить список коллекций, в которых роль или набор сценариев должны искать неуказанные названия модулей и действий. Таким образом, вы можете использовать ключевое слово, а затем ссылаться на модули и расширения действий по их сокращенным названиям в рамках этой роли или сценария.

Важно

Если в наборе сценариев используется как ключевое слово `collections`, так и одна или несколько ролей, то роли не используют коллекции, определенные в наборе сценариев. Это значит, что внутри роли нельзя ссылаться на коллекцию, заданную в наборе сценариев, и ее необходимо указывать явно. Это одна из причин, по которой рекомендуется всегда использовать FQCN.

Использование в ролях

В рамках роли можно управлять тем, какие коллекции будет искать Ansible для выполнения задач внутри роли. Для этого в файле `meta/main.yml` используйте слово `collections`. Ansible будет использовать список коллекций, определенный внутри роли, даже если в наборе сценариев, который вызывает роль, определены другие коллекции в отдельной записи с ключевым словом `collections`. Роли, определенные внутри коллекции, всегда неявно ищут сначала собственную коллекцию, поэтому вам не нужно использовать ключевое слово `collections` для доступа к модулям, действиям или другим ролям, содержащимся в той же коллекции.

Пример заполнения файла роли:

```
collections:
  - my_namespace.first_collection
  - my_namespace.second_collection
  - other_namespace.other_collection
```

Использование в наборах сценариев

В наборе сценариев можно управлять коллекциями, которые Ansible использует для поиска модулей и расширений действий. Однако любые роли, которые вы вызываете в наборе сценариев, определяют собственный порядок поиска коллекций, они не наследуют настройки вызывающего набора сценариев. Это верно даже в том случае, если роль не определяет собственное ключевое слово для коллекций.

Пример набора сценариев:

```
- name: Run a play using the collections keyword
  hosts: all
  collections:
    - my_namespace.my_collection

  tasks:

    - name: Import a role
      ansible.builtin.import_role:
        name: role1

    - name: Run a module not specifying FQCN
      my_module:
        option1: value

    - name: Run a debug task
      ansible.builtin.debug:
        msg: '{{ lookup("my_namespace.my_collection.lookup1", "param1")| my_namespace.my_
↪collection.filter1 }}'
```

Ключевое слово `collections` создает упорядоченный путь поиска для расширений без пространства имен и ссылок на роли. Оно не устанавливает содержимое и не изменяет поведение Ansible в отношении загрузки расширений или ролей. Обратите внимание, что для недействующих или модульных расширений, например, для поисковиков, фильтров и тестов, все равно требуется указывать FQCN.

При использовании ключевого слова `collections` не обязательно добавлять `ansible.builtin` в список поиска. Если его не включить, по умолчанию доступно следующее содержимое:

- стандартные модули и расширения Ansible, доступные через `ansible-base` или `ansible-core`;
- поддержка старых путей расширений третьих сторон.

Примечание

Предпочтительнее использовать FQCN модуля или расширения вместо использования ключевого слова `collections`.

Использование набора сценариев из коллекции

Вы можете распространять наборы сценариев в своей коллекции и вызывать их следующими способами:

- из командной строки:

```
ansible-playbook my_namespace.my_collection.playbook1.yml -i ./myinventory
```

- из другого набора сценариев:

```
- name: Import a playbook
  ansible.builtin.import_playbook: my_namespace.my_collection.playbookX
```

При создании наборов сценариев внутри коллекций рекомендуется использовать универсальные настройки для `hosts`:

- Глобальное выполнение на всех узлах. При необходимости можно ограничить выполнение с помощью аргумента `--limit` или пользовательского инвентаря:

```
- hosts: all
```

- Ограничение на управляющий узел (localhost), используется для задач, которые выполняются на управляющем узле:

```
- hosts: localhost
```

- Гибкое указание целевых узлов через переменную:

```
- hosts: '{{ target | default("webservers") }}'
```

  Здесь `target` – переменная, которую можно задать через параметр `-e 'target=host1,host2'`. Если переменная не передана, используется группа узлов `webservers` из инвентаря.

Примечание

- Названия наборов сценариев, как и другие ресурсы коллекции, имеют ограниченный набор допустимых символов. Названия могут содержать только строчные алфавитно-цифровые символы, а также _ и должны начинаться с буквы. Символ - не допустим в названиях наборов сценариев. Наборы сценариев с названиями, содержащими недопустимые символы, не могут быть адресованы – это ограничение импортера Python, который используется для загрузки ресурсов коллекции.

- Все расширения должны находиться в каталогах, специфичных для коллекции.

4.7 Справочные данные

4.7.1 Переменные поведения

Переменные поведения можно разделить на следующие типы:

- тип подключения;
- общие для всех подключений;
- настройки протокола SSH;
- повышение привилегий;
- параметры окружений управляемых узлов.

Тип подключения

Тип подключения к узлам платформы задается с помощью настройки `ansible_connection`:

- `ssh` (по умолчанию);
- `winrm` – подключение по протоколу WinRM к узлу с операционной системой Windows;
- `local` – если узел является локальной машиной;
- `paramiko` – подключение с использованием протокола Paramiko.

Пример заполнения:

INI

```
[webservers]
web01 ansible_connection=local
```

YAML

```
---
webservers:
  hosts:
    web01:
      ansible_connection: local
```

Общие для всех подключений

Для всех типов подключения возможно использование следующих настроек:

ansible_host

IP-адрес или доменное имя узла.

Пример заполнения:

INI

```
[webservers]
web01 ansible_host=192.168.56.11
```

YAML

```
---
webservers:
  hosts:
    web01:
      ansible_host: 192.168.56.11
```

ansible_port

Порт TCP для подключения, если он отличается от значения по умолчанию. Значение для SSH – 22.

Пример заполнения:

INI

```
[webservers]
web01 ansible_port=222
```

YAML

```
---
webservers:
  hosts:
    web01:
      ansible_port: 222
```

ansible_user

Название учетной записи пользователя.

Пример заполнения:

INI

```
[webservers]
web01 ansible_user=admin
```

YAML

```
---
webservers:
  hosts:
    web01:
      ansible_user: admin
```

ansible_password

Пароль, используемый для аутентификации.

Пример заполнения:

INI

```
[webservers]
web01 ansible_password=p@ssw0rd!
```

YAML

```
---
webservers:
  hosts:
    web01:
       ansible_password: p@ssw0rd!
```

Настройки протокола SSH

Для настройки протокола SSH используются следующие переменные:

ansible_ssh_private_key_file

Путь к файлу приватного ключа.

Пример заполнения:

INI

```
[webservers]
web01 ansible_ssh_private_key_file=/path/to/private_key
```

YAML

```
---
webservers:
  hosts:
    web01:
       ansible_ssh_private_key_file: /path/to/private_key
```

ansible_ssh_common_args

Дополнительные аргументы вызова утилит `ssh`, `scp` и `sftp`.

Пример заполнения:

INI

```
[webservers]
web01 ansible_ssh_common_args='-o StrictHostKeyChecking=no'
```

YAML

```
---
webservers:
  hosts:
    web01:
      ansible_ssh_common_args: '-o StrictHostKeyChecking=no'
```

ansible_sftp_extra_args

Дополнительные аргументы вызова утилиты `sftp`.

Пример заполнения:

INI

```
[webservers]
web01 ansible_sftp_extra_args='-C'
```

YAML

```
---
webservers:
  hosts:
    web01:
      ansible_sftp_extra_args: '-C'
```

ansible_scp_extra_args

Дополнительные аргументы вызова утилиты `scp`.

Пример заполнения:

INI

```
[webservers]
web01 ansible_scp_extra_args='-C'
```

YAML

```
---
webservers:
  hosts:
    web01:
      ansible_scp_extra_args: '-C'
```

ansible_ssh_extra_args

Дополнительные аргументы вызова утилиты ssh.

Пример заполнения:

INI

```
[webservers]
web01 ansible_ssh_extra_args='-v'
```

YAML

```
---
webservers:
  hosts:
    web01:
      ansible_ssh_extra_args: '-v'
```

ansible_ssh_pipelining

Сокращение количества операций подключения к управляемым узлам за счет выполнения некоторых модулей без копирования файлов:

- `true` – включено;
- `false` – выключено (по умолчанию).

Пример заполнения:

INI

```
[webservers]
web01  ansible_ssh_pipelining=true
```

YAML

```
---
webservers:
  hosts:
    web01:
       ansible_ssh_pipelining: true
```

ansible_ssh_executable

Путь к исполняемому файлу утилиты ssh, который следует использовать вместо системного.

Пример заполнения:

INI

```
[webservers]
web01 ansible_ssh_executable=/usr/bin/ssh
```

YAML

```
---
webservers:
  hosts:
    web01:
      ansible_ssh_executable: /usr/bin/ssh
```

Повышение привилегий

Для выполнения некоторых операций требуются повышенные привилегии. Способы их получения задаются с помощью следующих настроек:

ansible_become

Принудительное повышение привилегий.

Пример заполнения:

INI

```
[webservers]
web01  ansible_become=true
```

YAML

```
---
webservers:
  hosts:
    web01:
       ansible_become: true
```

ansible_become_method

Метод повышения привилегий:

- sudo – использование утилиты sudo (по умолчанию);
- su – использование утилиты su;
- doas – аналог утилиты sudo, используемый в некоторых системах (например, OpenBSD);
- dzdo – метод, который используется в средах, где установлен Centrify DirectAuthorize.

Пример заполнения:

INI

```
[webservers]
web01 ansible_become_method=su
```

YAML

```
---
webservers:
  hosts:
    web01:
      ansible_become_method: su
```

ansible_become_user

Название учетной записи, используемой для повышения привилегий.

Пример заполнения:

INI

```
[webservers]
web01 ansible_become_user=root
```

YAML

```
---
webservers:
  hosts:
    web01:
      ansible_become_user: root
```

ansible_become_password

Пароль учетной записи, используемой для повышения привилегий.

Пример заполнения:

INI

```
[webservers]
web01 ansible_become_password=escalation_password
```

YAML

```
---
webservers:
  hosts:
    web01:
      ansible_become_password: escalation_password
```

ansible_become_exe

Путь к исполняемому файлу, используемому для повышения привилегий.

Пример заполнения:

INI

```
[webservers]
web01 ansible_become_exe=/usr/bin/su
```

YAML

```
---
webservers:
  hosts:
    web01:
      ansible_become_exe: /usr/bin/su
```

ansible_become_flags

Флаги, используемые для выбранного метода повышения привилегий.

Пример заполнения:

INI

```
[webservers]
web01 ansible_become_flags='-p'
```

YAML

```
---
webservers:
  hosts:
    web01:
      ansible_become_flags: '-p'
```

Параметры окружений удаленных узлов

Для задания параметров окружений на удаленных узлах используются следующие настройки:

ansible_shell_type

Тип оболочки на управляемом узле:

- csh;
- fish;
- sh (по умолчанию).

Пример заполнения:

INI

```
[webservers]
web01  ansible_shell_type=csh
```

YAML

```
---
webservers:
  hosts:
    web01:
       ansible_shell_type: csh
```

ansible_python_interpreter

Путь к интерпретатору Python на управляемом узле.

Пример заполнения:

INI

```
[webservers]
web01 ansible_python_interpreter=/usr/bin/python3
```

YAML

```
---
webservers:
  hosts:
    web01:
      ansible_python_interpreter: /usr/bin/python3
```

ansible_shell_executable

Путь к исполняемому файлу оболочки на управляемом узле.

Значение по умолчанию: `/bin/sh`.

Пример заполнения:

INI

```
[webservers]
web01 ansible_shell_executable=/bin/csh
```

YAML

```
---
webservers:
  hosts:
    web01:
      ansible_shell_executable: /bin/csh
```

4.7.2 Модули

Описание модулей коллекции `ansible.builtin`.

add_host

Модуль `add_host` позволяет динамически добавлять узлы и группы в описание инвентаря во время выполнения сценариев. Это может быть полезно, если нужно добавить только что созданный узел и использовать его в последующих задачах.

Особенности работы

Особенности использования модуля `ansible.builtin.add_host`:

- Модуль позволяет динамически добавлять новые узлы и группы в инвентарный список во время выполнения набора сценариев. Однако, эти изменения действуют только во время выполнения конкретного набора сценариев. После его выполнения все добавленные узлы исчезнут из инвентарного списка.
- При добавлении узла обычно задают также его переменные, которые потом можно использовать в последующих сценариях того же набора. Например, при создании нового виртуального сервера в облаке, `add_host` может передать его IP-адрес, порт и название учетной записи для управления им.
- Модуль можно использовать на всех платформах, включая Windows.
- Модуль не обходит ограничение фильтра, задаваемого аргументом `--limit`. Новые узлы, добавленные с помощью `add_host`, не будут использоваться, если они не удовлетворяют условиям фильтрации.

  Например, рассмотрим запуск набора сценариев с помощью следующей команды:

  ```
  ansible-playbook playbook.yml --limit "webservers"
  ```

  Если узел, добавленный модулем `add_host`, не входит в группу `webservers`, то Ansible не будет выполнять для него задачи, хотя он добавлен в описание инвентаря.

Параметры

Модуль принимает следующие параметры:

- `name` (обязательный параметр) – FQDN или IP-адрес узла. Можно указать порт в формате `ip:port`, например, `192.168.1.100:2222`.
- `groups` – список групп, в которые будет добавлен узел.

- Дополнительные переменные – любые другие параметры, необходимые для работы узла, например, ansible_user, ansible_ssh_private_key_file, ansible_port, ansible_connection.

Атрибуты

Атрибуты определяют функции Ansible, которые может использовать модуль.

Атрибут	Описание
action	Модуль имеет соответствующее расширение действий (action plugin). Это означает, что часть работы может быть выполнена на управляющем узле.
async	Модуль не поддерживает асинхронное выполнение.
become	Модуль не поддерживает повышение привилегий.
bypass_h	Модуль добавляет узлы в инвентарный список не поочередно для каждого узла, а как глобальную задачу, которая выполняется для всех узлов сразу, игнорируя обычные ограничения и циклы.
bypass_t	Этот модуль не игнорирует ключевые слова loop и with_.
check_mo	Модуль работает в режиме проверки (check_mode) не в полном объеме. Режим проверки позволяет узнать, что бы произошло, если бы узел был добавлен в инвентарный список.
connecti	Модуль не использует соединение с удаленными узлами.
core	Только часть функциональности модуля может быть изменена с помощью расширений.
delegati	Модуль не поддерживает delegate_to, то есть не может выполнятся на другом узле.
diff_mod	Модуль не поддерживает режим сравнения (diff_mode).
ignore_c	Модуль игнорирует ключевое слово when:.
platform	Модуль поддерживает все платформы.
tags	Модуль поддерживает ключевое слово tags.
until	Модуль поддерживает механизм повторных попыток.

Примеры

Для изучения возможностей модуля ознакомьтесь с приведенными ниже примерами.

Добавление узла в группу и присваивание значения переменной

Следующий пример добавляет узел в группу just_created и присваивает переменной ansible_user значение admin:

```
- name: Add host to group 'just_created' with variable ansible_user=admin
  ansible.builtin.add_host:
    name: '192.168.1.101'
    groups: just_created
    ansible_user: admin
```

Здесь:

- '192.168.1.101' – IP-адрес узла, который будет добавлен в инвентарный список;
- just_created – группа, в которую добавляется узел.

Добавление узла в несколько групп

Следующий пример добавляет узел в несколько групп:

```
- name: Add host to multiple groups
  ansible.builtin.add_host:
    hostname: '192.168.1.102'
    groups:
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
      - group1
      - group2
```

Добавление узла с нестандартным портом

Следующий пример добавит узел с указанным IP-адресом и нестандартным портом:

```
- name: Add a host with a non-standard port local to your machines
  ansible.builtin.add_host:
    name: '192.168.1.103:2222'
```

Здесь `'192.168.1.103:2222'` – IP-адрес узла с портом. Это полезно, если для подключения к узлу по SSH используется порт, отличающийся от 22.

Подключение через промежуточный узел

Следующий пример показывает настройку подключения к узлу `192.168.1.103` через промежуточный узел `192.168.1.104`:

```
- name: Add a host alias that we reach through a tunnel
  ansible.builtin.add_host:
    hostname: '192.168.1.103'
    ansible_host: '192.168.1.104'
    ansible_port: '2222'
```

Здесь:

- `'192.168.1.103'` – IP-адрес управляемого узла;
- `'192.168.1.104'` – IP-адрес промежуточного узла;
- `'2222'` – порт для подключения.

Добавление всех узлов в группу

Следующий пример добавит в группу done все узлы, которые участвуют в выполнении сценария:

```
- name: Add all hosts running this playbook to the done group
  ansible.builtin.add_host:
    name: '{{ item }}'
    groups: done
  loop: "{{ ansible_play_hosts }}"
```

Здесь

- `"{{ ansible_play_hosts }}"` – перебирает все узлы, которые задействованы в текущем наборе сценариев;
- `'{{ item }}'` – использует каждый узел в списке и добавляет его в группу done.

apt

Важно

Эта часть документации находится в стадии разработки.

apt_key

Важно

Эта часть документации находится в стадии разработки.

apt_repository

Важно

Эта часть документации находится в стадии разработки.

assemble

Важно

Эта часть документации находится в стадии разработки.

assert

Важно

Эта часть документации находится в стадии разработки.

async_status

Важно

Эта часть документации находится в стадии разработки.

blockinfile

Важно

Эта часть документации находится в стадии разработки.

command

Важно

Эта часть документации находится в стадии разработки.

copy

Важно

Эта часть документации находится в стадии разработки.

cron

Важно

Эта часть документации находится в стадии разработки.

deb822_repository

Важно

Эта часть документации находится в стадии разработки.

debconf

Важно

Эта часть документации находится в стадии разработки.

debug

Важно

Эта часть документации находится в стадии разработки.

dnf

Важно

Эта часть документации находится в стадии разработки.

dnf5

Важно

Эта часть документации находится в стадии разработки.

dpkg_selections

Важно

Эта часть документации находится в стадии разработки.

expect

Важно

Эта часть документации находится в стадии разработки.

fail

Важно

Эта часть документации находится в стадии разработки.

fetch

Важно

Эта часть документации находится в стадии разработки.

file

Важно

Эта часть документации находится в стадии разработки.

find

Важно

Эта часть документации находится в стадии разработки.

gather_facts

Важно

Эта часть документации находится в стадии разработки.

get_url

Важно

Эта часть документации находится в стадии разработки.

getent

Важно

Эта часть документации находится в стадии разработки.

git

Важно

Эта часть документации находится в стадии разработки.

group

Важно

Эта часть документации находится в стадии разработки.

group_by

Важно

Эта часть документации находится в стадии разработки.

hostname

Важно

Эта часть документации находится в стадии разработки.

import_playbook

Важно

Эта часть документации находится в стадии разработки.

import_role

Важно

Эта часть документации находится в стадии разработки.

import_tasks

Важно

Эта часть документации находится в стадии разработки.

include_role

Важно

Эта часть документации находится в стадии разработки.

include_tasks

Важно

Эта часть документации находится в стадии разработки.

include_vars

Важно

Эта часть документации находится в стадии разработки.

iptables

Важно

Эта часть документации находится в стадии разработки.

known_hosts

Важно

Эта часть документации находится в стадии разработки.

lineinfile

Важно

Эта часть документации находится в стадии разработки.

meta

Важно

Эта часть документации находится в стадии разработки.

mount_facts

Важно

Эта часть документации находится в стадии разработки.

package

Важно

Эта часть документации находится в стадии разработки.

package_facts

Важно

Эта часть документации находится в стадии разработки.

pause

Важно

Эта часть документации находится в стадии разработки.

ping

Важно

Эта часть документации находится в стадии разработки.

pip

Важно

Эта часть документации находится в стадии разработки.

raw

Важно

Эта часть документации находится в стадии разработки.

reboot

Важно

Эта часть документации находится в стадии разработки.

replace

Важно

Эта часть документации находится в стадии разработки.

rpm_key

Важно

Эта часть документации находится в стадии разработки.

script

Важно

Эта часть документации находится в стадии разработки.

service

Важно

Эта часть документации находится в стадии разработки.

service_facts

Важно

Эта часть документации находится в стадии разработки.

set_fact

Важно

Эта часть документации находится в стадии разработки.

set_stats

Важно

Эта часть документации находится в стадии разработки.

setup

Важно

Эта часть документации находится в стадии разработки.

shell

Важно

Эта часть документации находится в стадии разработки.

slurp

Важно

Эта часть документации находится в стадии разработки.

stat

Важно

Эта часть документации находится в стадии разработки.

subversion

Важно

Эта часть документации находится в стадии разработки.

systemd_service

Важно

Эта часть документации находится в стадии разработки.

sysvinit

Важно

Эта часть документации находится в стадии разработки.

tempfile

Важно

Эта часть документации находится в стадии разработки.

template

Важно

Эта часть документации находится в стадии разработки.

unarchive

Важно

Эта часть документации находится в стадии разработки.

uri

Важно

Эта часть документации находится в стадии разработки.

user

Важно

Эта часть документации находится в стадии разработки.

validate_argument_spec

Важно

Эта часть документации находится в стадии разработки.

wait_for

Важно

Эта часть документации находится в стадии разработки.

wait_for_connection

Важно

Эта часть документации находится в стадии разработки.

yum_repository

Важно

Эта часть документации находится в стадии разработки.

ГЛАВА 5

Day 0: Планирование и подготовка

На этапе проектирования платформы необходимо выполнить ряд подготовительных действий.

5.1 Последовательность

В процессе подготовки к развертыванию платформы вам предстоит выполнить типовые шаги:

1. Выбрать одну из моделей развертывания платформы:
   - на виртуальных машинах и физических серверах;
   - в кластере Kubernetes.

   Рабочее окружение может быть на ваших серверах в центре обработки данных или в окружении облачного провайдера.
2. Выбрать одну из рекомендуемых топологий:
   - базовую;
   - масштаба предприятия.
3. Подготовить необходимые аппаратные и программные ресурсы:
   - создание и настройка сетевой инфраструктуры;
   - ВМ или физические серверы с необходимыми вычислительными ресурсами;
   - предварительно установленное программное обеспечение.
4. Подготовить служебные файлы:
   - описание инвентаря;
   - ключи и сертификаты TLS;
   - файлы с зашифрованными паролями.

5.2 Общие требования

Существуют общие требования, не зависящие от конкретной реализации.

5.2.1 Лицензия

После развертывания платформы необходимо активировать лицензию на подписку, приобретенную через Личный кабинет[10].

5.2.2 Рабочие места администратора и пользователей

Для работы с платформой через веб-интерфейс необходимо наличие на рабочих станциях (компьютерах) администратора и пользователей одного из следующих веб-браузеров:

Браузер	Версия, не ниже
Google Chrome	109
Mozilla Firefox	102
Яндекс Браузер	23.7.2.767

5.3 Модели развертывания

В последующих инструкциях рассмотрим следующие варианты развертывания платформы, из которых вы сможете выбрать наиболее подходящий для вас:

Способ	Инфраструктура	Описание	Протестированные топологии
Deb-пакеты	*ВМ или физические серверы*	Развертывание из deb-пакетов в операционной системе Astra Linux	- Базовая - Уровень предприятия
Operator	*Кластер Kubernetes*	Развертывание в среде Kubernetes	- Базовая - Уровень предприятия

5.3.1 На базе виртуальных машин

Фаза планирования и подготовки развертывания Astra Automation состоит из нескольких стадий:

Вы прошли стадию выбора модели развертывания, выбрав развертывание на виртуальных машиных или физических серверах. Остальные стадии впереди. Для перехода сразу на некоторую стадию выберите соответствующий блок диаграммы.

Развертывание платформы на виртуальных машинах в облачном пространстве или в собственном центре обработки данных является традиционным способом. Если вам больше подходит использование физических серверов вместо ВМ, то приведенные далее инструкции не меняются по сути.

Топология

На этом этапе необходимо выбрать подходящую топологию Astra Automation.

Согласно архитектуре, платформа содержит определенный набор компонентов. Конкретная топология платформы должна учитывать множество требований, предъявляемых к процессам и средствам автоматизации. При разработке собственной топологии рекомендуется брать в основу один из рассматриваемых здесь вариантов, прошедших необходимое тестирование. Все описанные топологии содержат полный набор компонентов плат-

[10] https://lk.astra.ru/

формы, но различаются отказоустойчивостью, сложностью настройки и количеством узлов.

- *Базовая топология* обеспечивает функциональную полноту, так как содержит все необходимые компоненты платформы. Не обеспечивает отказоустойчивости, однако, может быть удобна для начальной стадии развития процессов автоматизации в информационной инфраструктуре.
- *Топология уровня предприятия* обеспечивает высокое быстродействие и отказоустойчивость за счет увеличения затрат на ресурсы. Рекомендуется для промышленного применения на предприятиях с высокими требованиями к надежности сложной распределенной информационной инфраструктуры.

Характеристики ВМ

Характеристики виртуальных машин, на которых тестировались приведенные топологии:

- Узлы платформы и развернутой ее средствами СУБД:

Параметр	Значение
Количество ядер CPU	8
Количество RAM, ГБ	16
Дисковое пространство, ГБ	60
Тип дискового накопителя	Сетевой SSD
IOPS	3000
Тип ОС	Astra Linux Special Edition 1.7.7, 1.7.8, 1.8.2, 1.8.3
Версия ядра ОС	`6.1.141-1-generic`
Режим защищенности ОС	Базовый («Орел») Усиленный («Воронеж») Максимальный («Смоленск»)

Важно

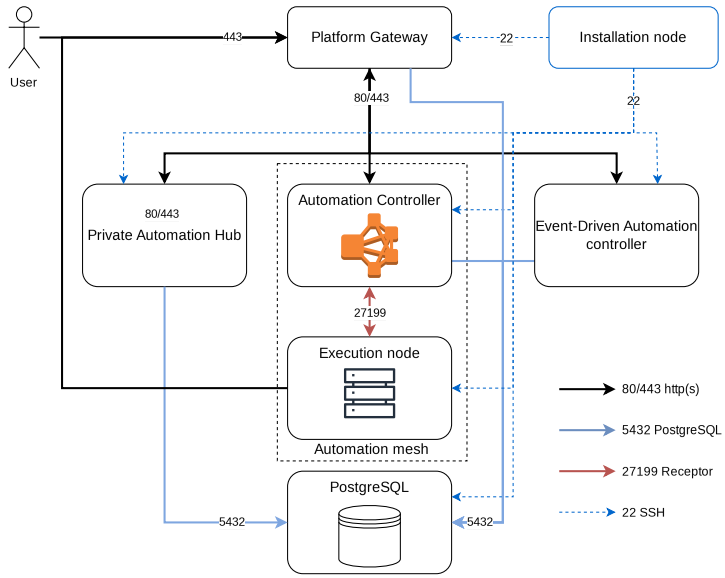
Операционная система должна быть развернута без пакетов графического интерфейса. Подробности см. в *общих требованиях к узлам*.

- Узел СУБД во внешнем окружении:

Параметр	Значение
Количество ядер CPU	8
Количество RAM, ГБ	32
Дисковое пространство, ГБ	100
Лимит на количество одновременно открытых подключений	300

Базовая топология

Топология платформы, использующей для запуска заданий автоматизации отдельный узел, показана на схеме:



При настройке сетевых фильтров надо учитывать следующие связи:

Источник	Назначение	Служба	Порт TCP
User workstation	Platform Gateway	HTTPS	443
Platform Gateway	- Automation Controller - Private Automation Hub - Event-Driven Automation	HTTP/HTTPS	80/443
- Automation Controller - Event-Driven Automation	Platform Gateway	HTTP/HTTPS	80/443
Event-Driven Automation	Platform Gateway	Redis	6379
- Platform Gateway - Automation Controller - Private Automation Hub - Event-Driven Automation	Database	PostgreSQL	5432
Automation Controller	Исполняющий узел	Receptor (двунаправленнс соединение)	27199
Installation node	All nodes	SSH	22
Installation node	Database	PostgreSQL	5432
Installation node	Управляющий и исполняющий узлы	Receptor	27199
Installation node	- Platform Gateway - Automation Controller - Private Automation Hub - Event-Driven Automation - Реестры инфраструктурного кода	HTTP/HTTPS	80/443

Примечание

Устанавливающий узел (installation node) дополнительно к соединению SSH, необходимому для управления узлами в процессе развертывания платформы, также контролирует доступность определенных портов TCP в процессе ранней проверки описания инвентаря.

Пример описания инвентаря для базовой топологии:

```
[automationgateway]
gw1.example.com

[automationcontroller]
ctrl.example.com
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
[automationcontroller:vars]
node_type=control
peers='execution_nodes'

[execution_nodes]
exec.example.com

[automationhub]
hub.example.com

[automationedacontroller]
eda.example.com

[database]
db.example.com

[all:vars]
# General
ansible_ssh_private_key_file='/path/to/private/ssh/key'
ansible_user='admin'
ansible_python_interpreter=/usr/bin/python3

# Redis
redis_mode=standalone

# Platform Gateway
automationgateway_admin_password='gateAAp@ssW0rd'
automationgateway_pg_host='192.168.56.101'
automationgateway_pg_port=5432
automationgateway_pg_database='automationgateway'
automationgateway_pg_username='automationgateway'
automationgateway_pg_password='gateDBpaS$12345'

# Automation Controller
admin_email='admin@example.com'
admin_password='ctRlp@ssW0rd'
pg_host=192.168.56.101
pg_port=5432
pg_database='awx'
pg_username='automationcontroller'
pg_password='ctrlDBpaS$123456'

# Private Automation Hub
automationhub_admin_password='hU8p4ssWd1234'
automationhub_main_url='https://hub.example.com'
automationhub_pg_host=192.168.56.101
automationhub_pg_port=5432
automationhub_pg_database='automationhub'
automationhub_pg_username='automationhub'
automationhub_pg_password='hUbPa55I2345b'

# Контроллер Event-Driven Automation
automationedacontroller_admin_password='edapass'
automationedacontroller_pg_host=192.168.56.101
automationedacontroller_pg_port=5432
automationedacontroller_pg_database='automationedacontroller'
automationedacontroller_pg_username='automationedacontroller'
automationedacontroller_pg_password='edapassword'
```

Особенности настройки:

- Под каждый компонент отводится отдельная секция, в которой описаны параметры идентификации и доступа.

- Если DNS не настроена для узлов платформы, то выберите один из альтернативных подходов:
  - Используйте названия узлов, конвертируемые в IP-адреса через файл `/etc/hosts`, который надо скопировать на все узлы платформы и на установочный узел.

    Пример заполнения секции `[automationgateway]` файла `inventory`:

    ```
    [automationgateway]
    gw1
    ```

  - Вместо названий подставьте IP-адреса узлов.

    Пример заполнения секции `[automationgateway]` файла `inventory`:

    ```
    [automationgateway]
    gw1   ansible_host=10.177.92.12
    ```

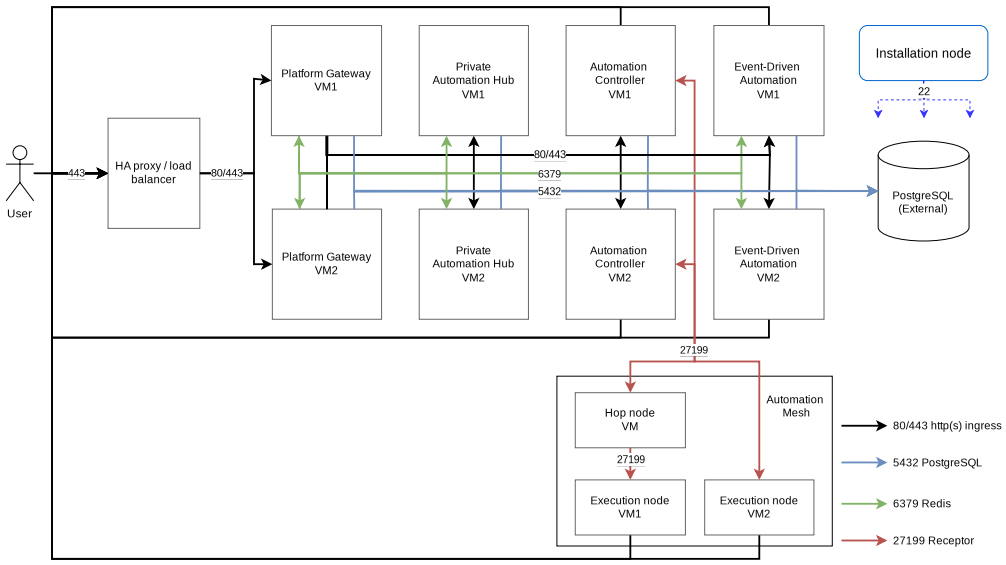
- СУБД развертывается на узле, указанном в секции `[database]`, средствами платформы.
- Каждый компонент платформы имеет доступ к собственной базе данных. Параметры баз данных (они же параметры доступа) описаны в секции `[all:vars]`.
- Секция `[execution_nodes]` содержит описание узлов плоскости исполнения. В приведенном файле она содержит один исполняющий узел. Это создает сеть Automation Mesh для связи управляющего узла с исполняющим.
- Тип узла Automation Controller отмечен явно как управляющий (`node_type=control`).
- Для установления соединения между отмеченными узлами по сети Mesh в настройках управляющего узла используется параметр `peers='execution_nodes'`, который приводит к установлению такого соединения между ним и всеми узлами, приведенными в секции `[execution_nodes]`.
- Сервис Redis устанавливается на каждом узле индивидуально согласно параметру `redis_mode=standalone`.
- Для установочного узла в секции `[all:vars]` описаны параметры подключения к узлам для развертывания платформы:
  - `ansible_ssh_private_key_file` – путь к файлу, содержащему приватный ключ SSH;
  - `ansible_user` – учетная запись, имеющая административные привилегии на каждом узле, устанавливаемые через утилиту `sudo`.

Предупреждение

Во всех компонентах Astra Automation автоматически создается учетная запись `admin` с административными привилегиями. В текущей версии продукта изменение ее названия с помощью параметров `automationgateway_admin_user` (Platform Gateway) , `admin_username` (Automation Controller), `automationhub_admin_user` (Private Automation Hub) и `automationedacontroller_admin_username` (контроллер Event-Driven Automation) приведет к развертыванию неработоспособной системы.

Топология уровня предприятия

Одним из важнейших требований, предъявляемых к платформе со стороны предприятий, является отказоустойчивость. Топология платформы, развернутой в отказоустойчивой конфигурации, показана на схеме:



Такая топология обеспечивает отказоустойчивость и повышенное быстродействие платформы путем горизонтального масштабирования ее состава:

- Каждый компонент платформы реализован с помощью двух узлов.
- Внутри платформы балансировка нагрузки между узлами одного компонента выполняется шлюзом платформы.
- Подключение к узлам шлюза выполняется через балансировщик нагрузки HAProxy[11].
- Двунаправленная связь между управляющими, промежуточными и исполняющими узлами выполняется через порт 27199 сети Automation mesh. Подключение управляющих узлов к исполняющим может выполняться как напрямую, так и через промежуточный узел.
- Устойчивость сервиса Redis обеспечена созданием отдельного кластера для него с использованием узлов платформы:
  - Platform Gateway;
  - контроллер Event-Driven Automation;
  - Private Automation Hub.

  Особенности использования кластера Redis:
  - Кластер для обеспечения устойчивости должен содержать не менее шести узлов.
  - Три (или другое нечетное количество) узла кластера являются основными (primary), которые выбирают одного из них в качестве лидера.
  - Для каждого основного узла создается вспомогательный (secondary) для замены основного в случае потери связи с последним.
  - Данные распределяются по трем основным узлам (с репликацией на соответствующих вспомогательных узлах) по типу шардирования баз данных.

Примечание

Клиентами кластера являются узлы Platform Gateway и EDA controller. Остальные узлы используют локальный Redis, установленный на каждом из них.

Параметры сетевого взаимодействия компонентов платформы:

[11] https://docs.haproxy.org/

Источник	Назначение	Служба	Порт TCP
User workstation	Load balancer	HTTPS	443
Load balancer	Platform Gateway	HTTP/HTTPS	80/443
Platform Gateway	- Automation Controller - Private Automation Hub - Event-Driven Automation	HTTP/HTTPS	80/443
- Automation Controller - Event-Driven Automation	Load balancer	HTTP/HTTPS	80/443
Platform Gateway	- Platform Gateway - Private Automation Hub - Event-Driven Automation	Redis cluster	6379
Event-Driven Automation	- Platform Gateway - Private Automation Hub - Event-Driven Automation	Redis cluster	6379
- Platform Gateway - Automation Controller - Private Automation Hub - Event-Driven Automation	Database	PostgreSQL	5432
Automation Controller	Execution_node	Receptor (двунаправленное соединение)	27199
Automation Controller	Hop node	Receptor (двунаправленное соединение)	27199
Hop node	Execution_node	Receptor (двунаправленное соединение)	27199
Installation node	All nodes	SSH	22
Installation node	Database	PostgreSQL	5432
Installation node	- Control node - Execution_node - Hop node	Receptor	27199
Installation node	- Platform Gateway	HTTP/HTTPS	80/443

Примечание

Устанавливающий узел (installation node) дополнительно к соединению SSH, необходимому для управления узлами в процессе развертывания платформы, также контролирует доступность определенных портов TCP в процессе ранней проверки описания инвентаря.

Пример файла инвентаря для топологии уровня предприятия:

```
# Platform Gateway
[automationgateway]
gw1.example.com
gw2.example.com

# Automation Controller
[automationcontroller]
ctrl1.example.com
ctrl2.example.com

[automationcontroller:vars]
node_type='control'
peers='instance_group_local'

# Execution plane
[execution_nodes]
hop.example.com
exec1.example.com
exec2.example.com

[instance_group_local]
exec2.example.com

[hop]
hop.example.com

[hop:vars]
node_type='hop'
peers='automationcontroller'

[instance_group_remote]
exec1.example.com

[instance_group_remote:vars]
peers='hop'

# Private Automation Hub
[automationhub]
hub1.example.com
hub2.example.com

# EDA controller
[automationedacontroller]
eda1.example.com
eda2.example.com

# Redis cluster
[redis]
hub1.example.com
hub2.example.com
gw1.example.com
gw2.example.com
eda1.example.com
eda2.example.com
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
# Database management system
[database]
# Используется внешний кластер PostgreSQL

# All variables
[all:vars]
# General
ansible_ssh_private_key_file='/path/to/private/ssh/key'
ansible_user='admin'
ansible_python_interpreter=/usr/bin/python3

# Gateway
automationgateway_admin_password='gateAAp@ssW0rd'
automationgateway_pg_host='pg-ha.example.com'
automationgateway_pg_port=5432
automationgateway_pg_database='automationgateway'
automationgateway_pg_username='automationgateway'
automationgateway_pg_password='gateDBpaS$12345'

# Automation Controller
admin_email='admin@example.com'
admin_password='ctRlp@ssW0rd'
pg_host='pg-ha.example.com'
pg_port=5432
pg_database='awx'
pg_username='automationcontroller'
pg_password='ctrlDBpaS$123456'

# Private Automation Hub
automationhub_admin_password='hU8pAssWd1234'
automationhub_pg_host='pg-ha.example.com'
automationhub_pg_port=5432
automationhub_pg_database='automationhub'
automationhub_pg_username='automationhub'
automationhub_pg_password='hUbPa55I2345b'

# Контроллер Event-Driven Automation
automationedacontroller_admin_password='edapass'
automationedacontroller_pg_host='pg-ha.example.com'
automationedacontroller_pg_port=5432
automationedacontroller_pg_database='automationedacontroller'
automationedacontroller_pg_username='automationedacontroller'
automationedacontroller_pg_password='edapassword'
```

Особенности настройки:

- Под каждый компонент отводится отдельная секция, в которой описаны параметры идентификации и доступа.
- Если DNS не настроена для узлов платформы, то выберите один из альтернативных подходов:
  - каждому узлу дайте название (не должно быть подобным доменному имени, то есть без точек), конвертируемое в IP-адрес через файл `/etc/hosts`, который надо скопировать на все узлы платформы и на установочный узел;
  - просто подставьте IP-адреса узлов.
- Секция `[database]` оставлена пустой, поскольку развертывание сервера PostgreSQL средствами платформы в этом случае не производится. В приведенном примере отказоустойчивый внешний сервис PostgreSQL доступен по доменному имени `pg-ha.example.com`.
- Кластер Redis развернут на шести узлах для распределения нагрузки и обеспечения его устойчивости в случае отказа узлов.

Примечание

При наличии более одного узла в группе `automationcontroller` учитывайте *принцип главного узла*.

Предупреждение

Во всех компонентах Astra Automation автоматически создается учетная запись `admin` с административными привилегиями. В текущей версии продукта изменение ее названия с помощью параметров `automationgateway_admin_user` (Platform Gateway) , `admin_username` (Automation Controller), `automationhub_admin_user` (Private Automation Hub) и `automationedacontroller_admin_username` (контроллер Event-Driven Automation) приведет к развертыванию неработоспособной системы.

Системные требования

Вы начинаете стадию подготовки узлов платформы.

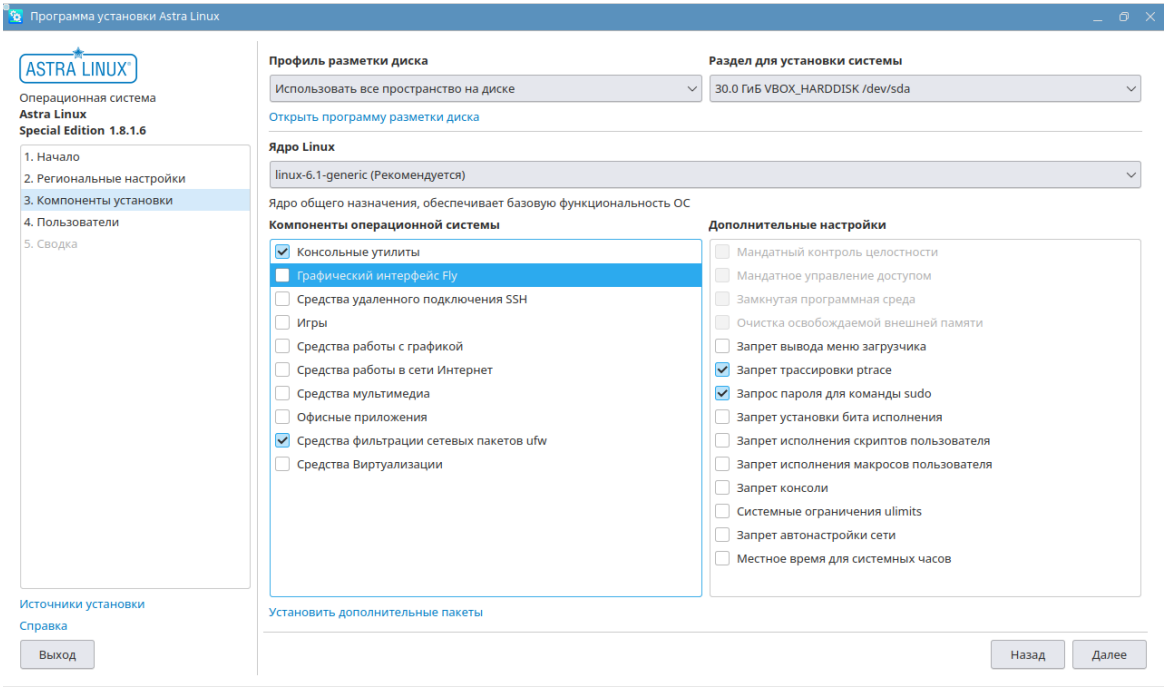
Для развертывания платформы необходимо обеспечить выполнение системных требований, предъявляемых к узлам платформы и используемой СУБД.

Общие требования

Фактический список узлов для развертывания платформы и их технические характеристики определяются требованиями к отказоустойчивости, обеспечению высокой доступности и быстродействию.

Дополнительные требования:

- Все узлы должны иметь постоянные IP-адреса или доменные имена (в зависимости от того, как заполняется файл инвентаря, используемый установщиком).
- Рекомендуется обеспечить передачу данных между всеми узлами плоскости управления Automation Controller с минимальными задержками.
- Узлы, используемые для развертывания Private Automation Hub, должны быть доступны с других узлов, входящих в состав развертываемой конфигурации, через порты TCP `80` и `443` в типовой топологии. В противном случае они не смогут загружать и обновлять из него образы сред исполнения, а запуск заданий будет прерываться с ошибками:
  - Image not known.
  - Error while pulling image.
- При развертывании без доступа к интернету (Offline Bundle) зарезервируйте на каждом узле не менее 10 ГБ свободного пространства для хранения установочных файлов платформы.
- На всех узлах используется одна и та же версия OS (операционной системы).
- На всех узлах установленная OS не должна содержать пакетов графического интерфейса.
  - Если вы устанавливаете OS вручную, на этапе выбора компонентов выключите флаг **Графический интерфейс Fly**.



- Если вы устанавливаете OS в автоматическом режиме, удалите из файла ответов строки, требующие установки графических пакетов.

  Astra Linux Special Edition 1.7 использует для автоматической установки утилиту `debian-installer`. Файл ответов называется `preseed.cfg`. Порядок его создания описан в статье Справочного центра Автоматическая установка Astra Linux Special Edition 1.7[12].

  Astra Linux Special Edition 1.8 использует для автоматической установки утилиту `astra-installer`. Файл ответов называется `astra-installer-preseed.yaml`. Порядок его создания описан в статье Справочного центра Автоматическая установка Astra Linux Special Edition x.8[13]. Там же приведены инструкции по преобразованию файла `preseed.cfg` в `astra-installer-preseed.yaml`.

  Примеры фрагмента файла ответов с исключением пакетов графического интерфейса:

  preseed.cfg

```
tasksel tasksel/first multiselect Base packages, SSH server
tasksel tasksel/astra-feat-setup multiselect
```

  astra-installer-preseed.yaml

```
---
# ...
tasks:
  - Base
  - Fly-ssh
```

- Если для развертывания платформы используются виртуальные машины, создавайте их на основе образов без пакетов поддержки графического интерфейса.

  Такие образы, созданные ПАО Группа Астра, не содержат в названии поле `-gui`. Подробности см. в документации универсальных базовых образов Astra Linux[14].

[12] https://wiki.astralinux.ru/pages/viewpage.action?pageId=68914051
[13] https://wiki.astralinux.ru/pages/viewpage.action?pageId=299469899
[14] https://registry.astralinux.ru/latest/descriptions/

Шлюз платформы

Типовые системные требования к вычислительным ресурсам для установки шлюза платформы (Platform Gateway):

Параметр	Минимальное значение	Рекомендуемое значение
Количество ядер CPU	4	≥ 8
Объем RAM, ГБ	16	≥ 16
Дисковое пространство, ГБ	60	≥ 100
Тип дискового накопителя	SSD	SSD
Производительность, IOPS	3000	≥3000
Тип OS	1.7.7	1.8.3
Версия ядра OS	`6.1.124-1-generic`	`6.1.141-1-generic`
Режим защищенности OS	• Базовый («Орел») • Усиленный («Воронеж») • Максимальный («Смоленск»)	• Базовый («Орел») • Усиленный («Воронеж») • Максимальный («Смоленск»)

Система автоматизации процессов

Система состоит из плоскости управления и плоскости исполнения. Типовые системные требования к узлам этих плоскостей зависят от их типа. Для предлагаемых топологий гибридные узлы не используются.

Управляющие узлы

Типовые системные требования к управляющим узлам:

Параметр	Минимальное значение	Рекомендуемое значение
Количество ядер CPU	4	≥ 8
Объем RAM, ГБ	16	≥ 16
Дисковое пространство, ГБ	60	≥ 100
Тип дискового накопителя	SSD	SSD
Производительность, IOPS	3000	≥3000
Тип OS	1.7.7	1.8.3
Версия ядра OS	`6.1.124-1-generic`	`6.1.141-1-generic`
Режим защищенности OS	• Базовый («Орел») • Усиленный («Воронеж») • Максимальный («Смоленск»)	• Базовый («Орел») • Усиленный («Воронеж») • Максимальный («Смоленск»)

Исполняющие узлы

Типовые системные требования к исполняющим узлам:

Параметр	Минимальное значение	Рекомендуемое значение
Количество ядер CPU	4	≥ 8
Объем RAM, ГБ	16	≥ 16
Дисковое пространство, ГБ	60	≥ 100
Тип OS	1.7.7	1.8.3
Версия ядра OS	`6.1.124-1-generic`	`6.1.141-1-generic`
Режим защищенности OS	• Базовый («Орел») • Усиленный («Воронеж») • Максимальный («Смоленск»)	• Базовый («Орел») • Усиленный («Воронеж») • Максимальный («Смоленск»)

На производительность исполняющих узлов наибольшее влияние оказывают количество ядер CPU и RAM.

Промежуточные узлы

Типовые системные требования к промежуточным узлам:

Параметр	Минимальное значение	Рекомендуемое значение
Количество ядер CPU	4	≥ 8
Объем RAM, ГБ	16	≥ 16
Дисковое пространство, ГБ	60	≥ 100
Тип OS	1.7.7	1.8.3
Версия ядра OS	`6.1.124-1-generic`	`6.1.141-1-generic`
Режим защищенности OS	• Базовый («Орел») • Усиленный («Воронеж») • Максимальный («Смоленск»)	• Базовый («Орел») • Усиленный («Воронеж») • Максимальный («Смоленск»)

На производительность промежуточных узлов наибольшее влияние оказывают пропускная способность сети и низкие задержки при передаче пакетов.

Рекомендации

На управляющих и исполняющих узлах зарезервируйте не менее 20 ГБ дискового пространства каталога `/var/lib/awx/`, используемого для хранения проектов и образов среды исполнения. При большом количестве проектов, хранящихся в каталоге `/var/lib/awx/projects/`, или их значительном объеме увеличьте размер пространства как минимум вдвое. Это позволит избежать ошибок, связанных с недостатком дискового пространства.

Требования к объему RAM зависят от количества одновременно проводимых операций автоматизации управляемых узлов. Оно, в свою очередь, не может превышать максимально допустимое количество ответвленных процессов (forks), которое задается в настройках шаблона задания или в конфигурационном файле `ansible.cfg`.

Методику и пример расчета количества управляющих и исполняющих узлов см. в разделе «Производительность».

Система управления контентом

Приватный реестр хранит контент автоматизации, который может храниться локально или во внешнем хранилище. Поэтому далее представлены требования как к узлам Private Automation Hub, так и к внешнему хранилищу, если оно используется.

Узлы Private Automation Hub

Типовые системные требования к узлам Private Automation Hub:

Параметр	Минимальное значение	Рекомендуемое значение
Количество ядер CPU	4	≥ 8
Объем RAM, ГБ	16	≥ 16
Дисковое пространство, ГБ	60	≥ 120
Тип дискового накопителя	SSD	SSD
Тип OS	1.7.7	1.8.3
Версия ядра OS	`6.1.124-1-generic`	`6.1.141-1-generic`
Режим защищенности OS	• Базовый («Орел») • Усиленный («Воронеж») • Максимальный («Смоленск»)	• Базовый («Орел») • Усиленный («Воронеж») • Максимальный («Смоленск»)

Хранилище контента

В качестве внешнего хранилища можно использовать один из вариантов:

- объектное хранилище S3 – рекомендуемый вариант;
- сервер NFS.

Примечание

Изменение внешнего хранилища после развертывания платформы потребует удаления и повторной загрузки всего содержимого.

Минимальный объем хранилища определяется объемом контента, который загружается из облачного Automation Hub и составляет не более трех ГБ. Рекомендуется зарезервировать не менее 40 ГБ.

Система управления событиями

Типовые системные требования к узлу контроллера Event-Driven Automation:

Параметр	Минимальное значение	Рекомендуемое значение
Количество ядер CPU	4	≥ 8
Объем RAM, ГБ	16	≥ 16
Дисковое пространство, ГБ	60	≥ 80
Тип дискового накопителя	SSD	SSD
Тип OS	1.7.7	1.8.3
Версия ядра OS	`6.1.124-1-generic`	`6.1.141-1-generic`
Режим защищенности OS	• Базовый («Орел») • Усиленный («Воронеж») • Максимальный («Смоленск»)	• Базовый («Орел») • Усиленный («Воронеж») • Максимальный («Смоленск»)

СУБД

Типовые системные требования к узлам СУБД PostgreSQL, в том числе развернутой за пределами рабочего пространства, выделенного для Astra Automation:

Параметр	Минимальное значение	Рекомендуемое значение
Версия PostgreSQL	15	15
Количество ядер CPU	4	≥ 8
Объем RAM, ГБ	16	≥ 16
Дисковое пространство, ГБ	20	≥ 150
Производительность, IOPS	1500	≥ 1500
Тип дискового накопителя	HDD	SSD

Для базы Private Automation Hub необходимо расширение `hstore` для PostgreSQL. При развертывании СУБД *средствами платформы* оно устанавливается и включается автоматически. При использовании внешней СУБД выполните настройки *самостоятельно*.

Astra Automation не поддерживает кластеризацию СУБД. Это значит, что для обеспечения ее высокой доступности необходимо использовать сторонние средства.

Балансировщик нагрузки

Для обеспечения высоких требований по устойчивости платформы рекомендуется использовать более одного узла для каждого компонента. Доступ к платформе в таком случае следует предоставлять через балансировщик нагрузки.

Подготовка узлов

На этом этапе необходимо подготовить узлы для развертывания Astra Automation с учетом выбранной топологии.

Принципиально различной выглядит подготовка следующих узлов:

- узел установщика (*установочный узел*) – вспомогательный узел, на котором будет установлен и использован пакет развертывания Astra Automation;
- узлы платформы для непосредственной установки компонентов Astra Automation.

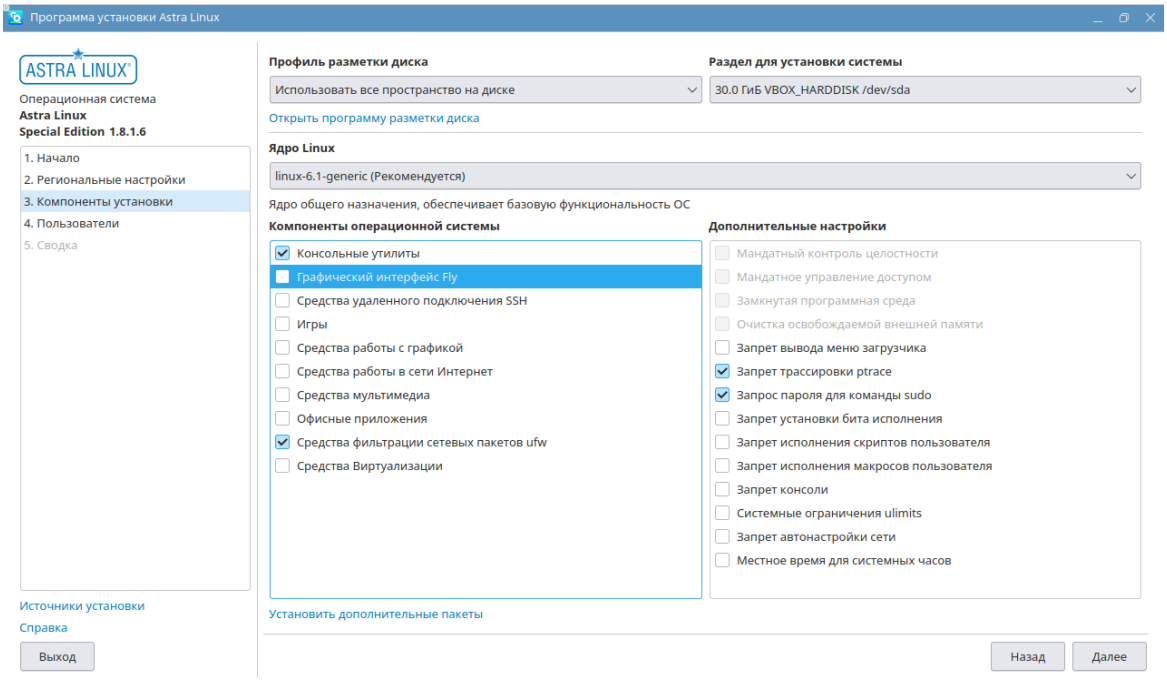
Общие требования

Общие требования к узлам:

- Все узлы должны иметь постоянные IP-адреса или доменные имена. От этого зависит способ идентификации узлов в файле инвентаря.
- Рекомендуется обеспечить передачу данных между всеми узлами плоскости управления Automation Controller с минимальными задержками.
- Узлы, используемые для развертывания Private Automation Hub, должны быть доступны с других узлов, входящих в состав развертываемой платформы. В противном случае некоторые узлы не смогут использовать содержимое, хранящееся в Private Automation Hub.
- При развертывании без доступа к интернету (с помощью Offline Bundle) зарезервируйте на каждом узле не менее 10 ГБ свободного места для хранения установочных файлов платформы.
- На всех узлах используется одна и та же версия ОС (операционной системы).
- Установленная на узлах ОС не должна содержать пакеты графического интерфейса.
  - Если для развертывания платформы используются виртуальные машины, создавайте их на основе образов без пакетов поддержки графического интерфейса.

Такие образы, созданные ПАО Группа Астра, не содержат в названии поле `-gui`. Подробности см. в документации универсальных базовых образов Astra Linux[15].

- Если вы устанавливаете ОС вручную, на этапе выбора компонентов выключите флаг **Графический интерфейс Fly**.



- Если вы устанавливаете ОС в автоматическом режиме, удалите из файла ответов строки, требующие установки графических пакетов.

  Astra Linux Special Edition 1.7 использует для автоматической установки утилиту `debian-installer`. Файл ответов называется `preseed.cfg`. Порядок его создания описан в статье Справочного центра Автоматическая установка Astra Linux Special Edition 1.7[16].

  Astra Linux Special Edition 1.8 использует для автоматической установки утилиту `astra-installer`. Файл ответов называется `astra-installer-preseed.yaml`. Порядок его создания описан в статье Справочного центра Автоматическая установка Astra Linux Special Edition x.8[17]. Там же приведены инструкции по преобразованию файла `preseed.cfg` в `astra-installer-preseed.yaml`.

  Примеры фрагмента файла ответов с исключением пакетов графического интерфейса:

  preseed.cfg

```
tasksel tasksel/first multiselect Base packages, SSH server
tasksel tasksel/astra-feat-setup multiselect
```

  astra-installer-preseed.yaml

```
---
# ...
tasks:
  - Base
  - Fly-ssh
```

[15] https://registry.astralinux.ru/latest/about
[16] https://wiki.astralinux.ru/pages/viewpage.action?pageId=68914051
[17] https://wiki.astralinux.ru/pages/viewpage.action?pageId=299469899

Установочный узел

Установочный узел должен содержать пакет для развертывания Astra Automation. Кроме того, для выполнения в дальнейшем операций резервного копирования потребуется утилита rsync.

Установка утилиты rsync

Установите утилиту rsync:

```
sudo apt install -y rsync
```

Установка пакета для развертывания Astra Automation

Установите пакет для развертывания Astra Automation:

1. Если в файле /etc/apt/sources.list не содержатся ссылки на основной (main) и расширенный (extended) репозитории Astra Linux Special Edition, добавьте их.

   Важно

   Версии используемых репозиториев должны соответствовать версии ОС, установленной на узлах платформы.

   Например, для Astra Linux Special Edition 1.8.3.UU1 ссылки на репозитории имеют следующий вид:

   ```
   deb https://download.astralinux.ru/astra/frozen/1.8_x86-64/1.8.3/uu/1/extended-
   ↪repository/ 1.8_x86-64 main non-free non-free-firmware contrib
   deb https://download.astralinux.ru/astra/frozen/1.8_x86-64/1.8.3/uu/1/main-
   ↪repository/ 1.8_x86-64 main contrib non-free non-free-firmware
   ```

   Подробности о сетевых репозиториях Astra Linux Special Edition 1.8 см. в статье справочного центра Интернет-репозитории Astra Linux Special Edition x.8[18].

2. В каталоге /etc/apt/sources.list.d/ создайте файл astra-automation.list со ссылкой на репозиторий Astra Automation:

   ```
   deb https://dl.astralinux.ru/aa/aa-debs-for-alse-1.8 <version> main
   ```

   Вместо <version> подставьте версию устанавливаемой платформы, например, 2.0.

   Доступные версии продукта опубликованы в таблице История обновлений[19].

3. Обновите список доступных пакетов:

   ```
   sudo apt update
   ```

4. Установите пакет astra-automation-setup:

   ```
   sudo apt install astra-automation-setup --yes
   ```

   По окончании установки выводится сообщение следующего вида:

   ```
   ----------------------------------
   Welcome to Astra Automation Setup!
   ----------------------------------

   Follow next steps to install Astra Automation:
   1. Go to /opt/rbta/aa/astra-automation-setup
   ```

(продолжается на следующей странице)

[18] https://wiki.astralinux.ru/pages/viewpage.action?pageId=302043111
[19] https://docs.astra-automation.ru/latest/releases/#releases-history

(продолжение с предыдущей страницы)

```
2. Edit inventory file
3. Run aa-setup
4. Enjoy!

For guidance on installing Astra Automation, consult https://docs.astra-automation.
↪ru/
```

5. В каталоге установщика подготовлены шаблоны файла инвентаря для базовой топологии (`inventory-base`) и топологии уровня предприятия (`inventory-enterprise`). В зависимости от выбранной вами топологии скопируйте соответствующий шаблон в файл инвентаря для установщика:

```
cd /opt/rbta/aa/astra-automation-setup && cp inventory-base.ini inventory
```

или

```
cd /opt/rbta/aa/astra-automation-setup && cp inventory-enterprise.ini inventory
```

На *этапе описания инвентаря* вы будете редактировать файл `inventory` так, чтобы он отвечал вашим требованиям.

Узлы платформы

Каждый узел платформы должен быть доступен и управляем по SSH со стороны установочного узла. Для подготовки узлов к развертыванию платформы выполните следующие действия:

1. Обеспечьте сетевую связность всех узлов согласно *выбранной топологии*.
2. (Необязательно) Обеспечьте конвертацию названий узлов в IP-адреса (с помощью `/etc/hosts` или DNS).
3. Настройте доступ к узлам по SSH с использованием ключей.

   Инструкция по настройке приведена *ниже*.
4. Разрешите выполнение команд с использованием `sudo` без ввода пароля:

```
sudo astra-sudo-control disable
```

Подготовка узлов для кластера Private Automation Hub

Существует три варианта хранения контента в Private Automation Hub:

- Локальная файловая система.

  Вариант используется по умолчанию. При этом весь контент сохраняется в локальной файловой системе узла, на котором развернут Private Automation Hub. Этот вариант подходит, когда Private Automation Hub развернут на одном узле. Из-за отсутствия синхронизации между узлами, при отказе одного из них контент может быть утерян.

  Рекомендуется для тестовых установок, стендов разработки и ситуаций, когда отказоустойчивость не критична.

  Для реализации данного варианта достаточно *настроить узел* и добавить его в *описание инвентаря* без дополнительных шагов.

- Облачное хранилище типа S3.

  Поддерживается подключение к внешнему объектному хранилищу типа S3[20]. Контент хранится вне узлов кластера, что позволяет обеспечить отказоустойчивость и масштабируемость без необходимости настройки локального сервера хранения. Вариант подходит для рабочих или облачных сред при наличии доступа к сервисам, поддерживающим S3, например:

[20] https://en.wikipedia.org/wiki/Amazon_S3

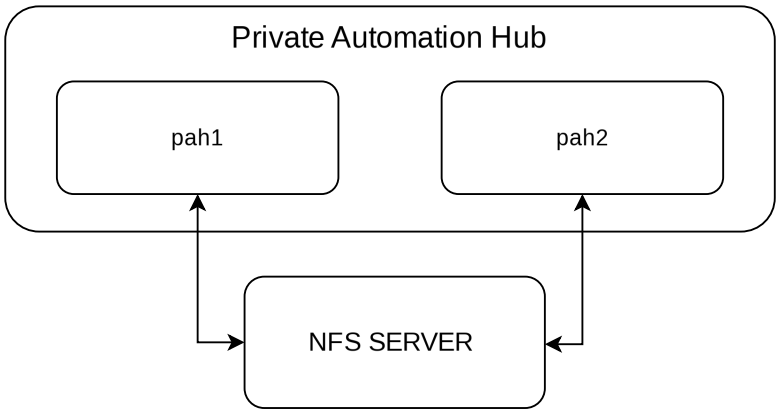
  - Yandex Object Storage[21];
  - Selectel Object Storage[22];
  - VK Cloud Object Storage[23];
  - Amazon S3[24].

- Разделяемое хранилище NFS.

  Предусматривает использование отдельного узла – сервера NFS, который предоставляет единое хранилище для всех узлов кластера Private Automation Hub. Такой подход обеспечивает централизованное хранение данных и возможность отказоустойчивой работы в кластере из нескольких узлов. Контент, размещенный в этом хранилище, доступен сразу всем узлам, что делает возможным масштабирование без дополнительной синхронизации.

Разделяемое хранилище NFS

Настройка разделяемого хранилища может различаться в зависимости от требуемых характеристик. Далее представлен пример настройки разделяемого хранилища NFS для двух узлов Private Automation Hub и одного сервера NFS.



Важно

- На всех используемых узлах в файле `/etc/apt/sources.list` должны содержаться ссылки на основной (main) и расширенный (extended) репозитории Astra Linux Special Edition. Версии используемых репозиториев должны соответствовать версии ОС, установленной на узлах платформы.

  Например, для Astra Linux Special Edition 1.8.1.UU1 ссылки на репозитории имеют следующий вид:

```
deb http://download.astralinux.ru/astra/frozen/1.8_x86-64/1.8.1/uu/1/main-
↪repository     1.8_x86-64 main non-free non-free-firmware contrib
deb http://download.astralinux.ru/astra/frozen/1.8_x86-64/1.8.1/uu/1/extended-
↪repository 1.8_x86-64 main non-free non-free-firmware contrib
```

- Для корректной работы протокола NFS требуется открыть TCP-порты 111 и 2049 на сервере. Разрешите доступ к этим портам только с IP-адресов узлов Private Automation Hub, как показано далее.

Настройте сервер NFS:

1. Обновите список доступных пакетов:

[21] https://yandex.cloud/ru/docs/storage/
[22] https://docs.selectel.ru/object-storage/
[23] https://cloud.vk.com/storage/
[24] https://docs.aws.amazon.com/AmazonS3/latest/userguide/Welcome.html

```
sudo apt update
```

2. Установите пакеты:

```
sudo apt install -y iptables iptables-persistent
```

3. Разрешите доступ к портам NFS (111 и 2049) с узлов Private Automation Hub:

```
sudo iptables -A INPUT -p tcp -s <ip_pah_1> --dport 111 -j ACCEPT
sudo iptables -A INPUT -p tcp -s <ip_pah_2> --dport 111 -j ACCEPT
sudo iptables -A INPUT -p tcp -s <ip_pah_1> --dport 2049 -j ACCEPT
sudo iptables -A INPUT -p tcp -s <ip_pah_2> --dport 2049 -j ACCEPT
```

Здесь <ip_pah_1> и <ip_pah_2> – IP-адреса узлов кластера Private Automation Hub.

4. Сохраните правила:

```
sudo netfilter-persistent save
```

5. Убедитесь, что правила применились:

```
sudo iptables -L -n | grep -wE '111|2049'
```

В выводе должны отображаться добавленные правила для IP-адресов клиентских узлов.

6. Установите сервер NFS:

```
sudo apt install -y nfs-kernel-server
```

7. Создайте каталог, который будет экспортироваться:

```
sudo mkdir -p /mnt/nfs/pulp_media
```

8. Измените режим доступа к созданному каталогу на 755:

```
sudo chmod -R 755 /mnt/nfs/pulp_media
```

9. Добавьте в файл `/etc/exports` следующие строки:

```
/mnt/nfs/pulp_media <ip_pah_1>(sync,rw,no_root_squash,no_subtree_check)
/mnt/nfs/pulp_media <ip_pah_2>(sync,rw,no_root_squash,no_subtree_check)
```

10. Примените настройку:

```
sudo exportfs -rav
```

11. Перезапустите сервер NFS:

```
sudo systemctl restart nfs-kernel-server
```

12. Проверьте успешность экспорта:

```
sudo exportfs -v
```

Пример вывода:

```
/mnt/nfs/pulp_media
                <ip_pah_1>(sync,wdelay,hide,no_subtree_check,sec=sys,rw,secure,no_
↪root_squash,no_all_squash)
/mnt/nfs/pulp_media
                <ip_pah_2>(sync,wdelay,hide,no_subtree_check,sec=sys,rw,secure,no_
↪root_squash,no_all_squash)
```

Настройте узлы Private Automation Hub:

Примечание

Нижеуказанные действия необходимо выполнить на каждом узле Private Automation Hub, который должен иметь доступ к общему хранилищу.

1. Обновите список доступных пакетов:

```
sudo apt update
```

2. Установите клиент NFS:

```
sudo apt install -y nfs-common
```

3. Создайте точку монтирования:

```
sudo mkdir -p /var/lib/pulp/media
```

4. Чтобы общее хранилище монтировалось при загрузке ОС, добавьте в `/etc/fstab` следующие строки:

```
<ip_nfs>:/mnt/nfs/pulp_media /var/lib/pulp/media nfs defaults,_netdev 0 0
```

   Здесь <ip_nfs> – IP-адрес сервера NFS.

5. Обновите настройки служб systemd:

```
sudo systemctl daemon-reload
```

6. Смонтируйте файловую систему:

```
sudo mount -a
```

7. Убедитесь, что хранилище подключено:

```
mount | grep pulp_media
```

   В выводе должна присутствовать строка с путем монтирования `/var/lib/pulp/media`.

Проверьте корректность настройки:

1. На первом узле Private Automation Hub создайте тестовый файл:

```
sudo touch /var/lib/pulp/media/test_hub1.txt
```

2. На втором узле Private Automation Hub убедитесь, что файл доступен:

```
ls -la /var/lib/pulp/media/
```

   В выводе должен отображаться файл `test_hub1.txt`.

3. Удалите файл на втором узле Private Automation Hub:

```
sudo rm /var/lib/pulp/media/test_hub1.txt
```

4. На первом узле Private Automation Hub убедитесь, что файл отсутствует:

```
ls -la /var/lib/pulp/media/
```

   В выводе не должен отображаться файл `test_hub1.txt`.

Настройка SSH для доступа к узлам

Рекомендуемым способом доступа к узлам является подключение по протоколу SSH с использованием ключей. Необходимо создать пару ключей – приватный и публичный. Приватный ключ необходимо оставить на установочном узле, а публичный передать на остальные узлы платформы.

Генерация ключей SSH

Для генерации пар ключей SSH на установочном узле выполните следующие действия:

1. Перейдите в каталог установщика, например:

```
cd /opt/rbta/aa/astra-automation-setup/
```

2. Создайте пару ключей SSH:

```
sudo ssh-keygen -C "<comment>" -f <path/to/file> -N "<password>"
```

   где:

   - <comment> – комментарий к ключу.
   - <path/to/file> – путь к файлу, в котором будет приватный ключ. Этот путь необходимо будет указать в настройках инвентаря для установщика. Публичный ключ будет создан по тому же пути в файле с тем же названием, к которому добавляется расширение `.pub`.
   - <password> – пароль для защиты приватного ключа. Если пароль пустой, приватный ключ создается без защиты.

3. Сделайте файл доступным только для его владельца:

```
sudo chmod 600 <path/to/file>
```

   где <path/to/file> – путь к файлу, в котором сохранен приватный ключ.

Настройка целевых узлов

Чтобы разрешить подключение к узлу по SSH с использованием ключей, выполните следующие действия на каждом узле:

1. В каталоге `/etc/ssh/sshd_config.d/` создайте файл `ssh_auth_keys` со следующим содержимым:

```
PubkeyAuthentication  yes
AuthorizedKeysFile   .ssh/authorized_keys  .ssh/authorized_keys2
```

2. Перезапустите сервер SSH:

```
sudo systemctl restart ssh
```

3. В домашнем каталоге пользователя-администратора создайте подкаталог `.ssh/`, а в нем – файл `authorized_keys`.
4. Сделайте файл доступным только его владельцу:

```
chmod 600 ~/.ssh/authorized_keys
```

5. Добавьте в файл `~/.ssh/authorized_keys` содержимое соответствующего публичного ключа, например (часть содержимого ключа опущена с целью сокращения):

```
ssh-rsa AAAAB3NzaC1yc2EAd...4bwy3tY2/ administrator@node1.example.com
```

Совет

Если узел разрешает подключение по SSH с использованием пароля, публичную часть ключей SSH можно разместить с помощью утилиты `ssh-copy-id`, выполнив следующую команду на установочном узле:

```
ssh-copy-id -i <path/to/file> <user>@<host>
```

где:

- <path/to/file> – путь к файлу, содержащему публичный ключ SSH;
- <user> – название учетной записи администратора узла;
- <host> – IP-адрес или FQDN узла.

Описание инвентаря

Для развертывания платформы с помощью утилиты `aa-setup` необходимо предоставить для нее полное описание всех узлов платформы в виде файла инвентаря.

По умолчанию утилита использует файл `inventory`, который вы уже создали при *подготовке установочного узла*. Теперь необходимо отредактировать этот файл так, чтобы он содержал параметры вашей инфраструктуры. Для этого рассмотрим более подробно параметры инвентаря по каждому компоненту Astra Automation. Начнем с общих настроек (**General** на диаграмме):

Если необходимо сразу перейти на какой-либо шаг, выберите соответствующий блок диаграммы.

Описание инвентаря следует правилам Ansible с использованием формата INI или YAML. В зависимости от используемого формата название файла должно иметь одно из следующих расширений:

- формат INI:
  - `.ini`;
  - `.cfg`;
- формат YAML:
  - `.yml`;
  - `.yaml`.

Примечание

Если расширение не указано, утилита развертывания интерпретирует файл как имеющий формат INI.

Утилита `aa-setup` по умолчанию использует файл инвентаря `inventory` в формате INI, расположенный в одном из следующих каталогов:

- `/opt/rbta/aa/astra-automation-setup/` – при использовании интернет-репозиториев ПАО Группа Астра;
- в корневом каталоге распакованного архива – при использовании оффлайн-установщика.

Чтобы использовать другой файл инвентаря, при запуске утилиты `aa-setup` укажите путь к нему в значении аргумента `--inventory` (`-i`), например:

INI

```
sudo ./aa-setup --inventory /var/aa/setup-settings.ini
```

YAML

```
sudo ./aa-setup --inventory /var/aa/setup-settings.yaml
```

Компоненты платформы

Настройки отдельных компонентов платформы задаются в соответствующих секциях файла инвентаря:

INI	YAML	Описание
[automationgateway]	automationgateway.hosts	Узлы шлюза платформы
[automationcontroller]	automationcontroller.hosts	Узлы плоскости управления (Automation Controller)
[execution_nodes]	execution_nodes.hosts	Узлы плоскости исполнения
[automationhub]	automationhub.hosts	Узлы Private Automation Hub
[automationedacontroll	automationedacontrolle hosts	Узлы контроллера Event-Driven Automation
[database]	database.hosts	Узел СУБД, развертываемой средствами платформы
[<group>:vars]	<group>.vars	Параметры настройки узлов конкретной группы <group>
[all:vars]	all.vars	Параметры настройки узлов и платформы в целом

После заголовка секции указывают параметры узлов платформы.

Внимание

Использование одного узла для развертывания двух и более компонентов платформы не допускается.

Реквизиты доступа к узлам

Для доступа Ansible к узлам платформы необходимо указать адрес узла и реквизиты учетной записи, которые вы подготовили при *настройке установочного узла*:

- `ansible_user` – название учетной записи пользователя, используемой для подключения к узлу (по умолчанию – учетная запись текущего пользователя);
- `ansible_ssh_private_key_file` – путь к файлу приватного ключа SSH.

Если для подключения ко всем узлам используются одни и те же реквизиты, укажите их в глобальных переменных, например:

INI

```
[automationcontroller]
node1.example.com
node2.example.com

[all:vars]
ansible_user='administrator'
ansible_ssh_private_key_file='ssh-keys/ssh_key'
```

YAML

```
---
automationcontroller:
  hosts:
    node1.example.com:
    node2.example.com:
  vars:
    ansible_user: administrator
    ansible_ssh_private_key_file: ssh-keys/ssh_key
```

Если реквизиты для доступа к узлам различаются, укажите их в параметрах соответствующих узлов, например:

INI

```
[automationcontroller]
node1.example.com  ansible_user=alex  ansible_ssh_private_key_file=ssh-keys/node1_key
node2.example.com  ansible_user=john  ansible_ssh_private_key_file=ssh-keys/node2_key

[execution_nodes]
node3.example.com  ansible_user=jack  ansible_ssh_private_key_file=ssh-keys/node3_key
```

YAML

```
---
automationcontroller:
  hosts:
    node1.example.com:
      ansible_user: alex
      ansible_ssh_private_key_file: ssh-keys/node1_key
    node2.example.com:
      ansible_user: john
      ansible_ssh_private_key_file: ssh-keys/node2_key
execution_nodes:
  hosts:
    node3.example.com:
      ansible_user: jack
      ansible_ssh_private_key_file: ssh-keys/node3_key
```

Защита конфиденциальных данных с помощью Ansible Vault

Для защиты указанных в описании инвентаря конфиденциальных данных рекомендуется вынести их в отдельный файл и зашифровать с помощью утилиты `ansible-vault`.

Преимущества такого подхода:

- Конфиденциальные данные не хранятся в открытом виде. Зашифрованный файл можно безопасно хранить и передавать.
- Основной файл инвентаря установщика можно добавить в систему управления версиями. Поскольку он не содержит конфиденциальных данных, это исключает риск их компрометации.
- Конфиденциальные данные обновляются отдельно от основной конфигурации.

Чтобы защитить конфиденциальные данные, выполните следующие действия:

1. Убедитесь, что у вас установлен пакет `ansible` на вашей рабочей станции. Если работаете на подготовленном установочном узле, то этот пакет уже установлен на нем. В противном случае установите его:

   ```
   sudo apt install ansible --yes
   ```

2. В каталоге установщика создайте файл формата YAML и добавьте в него необходимые переменные. В их значениях укажите конфиденциальные данные в открытом виде.

В этом примере файл с конфиденциальными данными называется `secrets.yml`. Пример содержимого:

```
---
ac_username: superadmin
ac_password: p@ssW0rD!
postgresql_ac_username: user3532
postgresql_ac_password: pgPa5Sw0r0
postgresql_pah_username: user9853
postgresql_pah_password: s$949d9fK
```

3. Зашифруйте файл `secrets.yml`:

```
sudo ansible-vault encrypt secrets.yml
```

   По запросу введите пароль для защиты содержимого файла `secrets.yml`.

4. В описании инвентаря вместо конфиденциальных данных укажите названия соответствующих переменных из файла `secrets.yml`. Для доступа к значениям переменных используйте синтаксис шаблонов Jinja.

   Для примера выше:

   INI

```
[all:vars]
admin_username='{{ ac_username }}'
admin_password='{{ ac_password }}'
pg_username='{{ postgresql_ac_username }}'
pg_password='{{ postgresql_ac_password }}'
automationhub_pg_username='{{ postgresql_pah_username }}'
automationhub_pg_password='{{ postgresql_pah_password }}'
```

   YAML

```
---
# ...
all:
  vars:
    admin_username: "{{ ac_username }}"
    admin_password: "{{ ac_password }}"
    # ...
    pg_username: "{{ postgresql_ac_username }}"
    pg_password: "{{ postgresql_ac_password }}"
    # ...
    automationhub_pg_username: "{{ postgresql_pah_username }}"
    automationhub_pg_password: "{{ postgresql_pah_password }}"
```

5. На *этапе развертывания платформы* при запуске утилиты `aa-setup` следует добавить аргументы `--extra-vars` и `--ask-vault-pass`, например:

   INI

```
sudo ./aa-setup --inventory /var/aa/setup-settings.ini -- --extra-vars @secrets.yml␣
↪--ask-vault-pass
```

   YAML

```
sudo ./aa-setup --inventory /var/aa/setup-settings.yml -- --extra-vars @secrets.yml␣
↪--ask-vault-pass
```

Примечание

Символ `@` перед названием файла `secrets.yml` является частью синтаксиса и указывает, что значения переменных нужно загрузить из указанного файла.

6. Введите пароль, которым защищен файл `secrets.yml`.

Шлюз платформы

На этом шаге отредактируйте описание шлюза платформы.

Шлюз платформы предоставляет единый интерфейс для управления всеми компонентами платформы. Настройка его узлов выполняется через описание инвентаря в отдельной секции `[automationgateway]` и различные специальные переменные. Кроме обязательных переменных, приведенных в *описании рекомендуемых топологий*, существует множество других, которые могут понадобиться в зависимости от ваших конкретных требований к платформе. Многие параметры принимают значения по умолчанию, но вы можете изменить эти параметры.

Описание узлов

В зависимости от выбранной топологии секция `[automationgateway]` содержит один или несколько узлов. Пример описания двух узлов:

INI

```
[automationgateway]
gw1.example.com
gw2.example.com
```

YAML

```
---
automationgateway:
  hosts:
    gw1.example.com
    gw2.example.com
```

При наличии более одного узла следует определить URL, по которому будут обращаться компоненты платформы при взаимодействии с другими компонентами внутри платформы. Для этого существует переменная `automationgateway_main_url`. Ее значение может указывать на следующие узлы:

- При использовании внешнего балансировщика нагрузки (HAProxy, NGINX, F5 и другие) URL должен указывать на этот балансировщик.
- Если внешний балансировщик не используется, URL должен указывать на один из узлов шлюза.

Если не использовать переменную `automationgateway_main_url`, то компоненты Astra Automation будут настроены на адрес первого узла в группе `automationgateway`.

Пример настройки:

INI

```
[all:vars]
automationgateway_main_url=https://aa-gateway.example.com
```

YAML

```
---
all:
  vars:
    automationgateway_main_url: https://aa-gateway.example.com
```

Защита входных данных

Для защиты данных по протоколу HTTPS используют сертификаты TLS с соответствующими ключами. Рекомендуется использовать сертификаты, подписанные удостоверяющим центром (публичным или собственным). Пример настройки на использование определенных сертификатов и ключей описано подробно в *инструкции по применению сертификатов TLS*.

Администратор платформы

Учетные данные системного администратора платформы включают:

- название учетной записи («admin» по умолчанию);
- пароль («admin» по умолчанию);

  Примечание

  В те

- почтовый адрес для посылки сообщений.

Укажите требуемые значения в глобальных переменных, например:

INI

```
[all:vars]
automationgateway_admin_username='aaadmin'
automationgateway_admin_password='gateAAp@ssW0rd'
automationgateway_admin_email='admin@example.com'
```

YAML

```
---
all:
  vars:
    automationgateway_admin_username: admin
    automationgateway_admin_password: p@ssw0rd!
    automationgateway_admin_email: admin@example.com
```

Доступ к базе данных

Обеспечьте доступ к базе данных с помощью следующих параметров:

INI

```
[all:vars]
automationgateway_pg_host='192.168.56.101'
automationgateway_pg_port=5432
automationgateway_pg_database='automationgateway'
automationgateway_pg_username='automationgateway'
automationgateway_pg_password='gateDBpaS$12345'
automationgateway_pg_sslmode='prefer'
```

YAML

```yaml
all:
  vars:
    automationgateway_pg_host: 192.168.56.101
    automationgateway_pg_port: 5432
    automationgateway_pg_database: automationgateway
    automationgateway_pg_username: automationgateway
    automationgateway_pg_password: gateDBpaS$12345
    automationgateway_pg_sslmode: prefer
```

Назначение переменных:

- `automationgateway_pg_host` – IP-адрес или FQDN узла с развернутой СУБД;
- `automationgateway_pg_port` – порт TCP для доступа к СУБД;
- `automationgateway_pg_database` – название базы данных;
- `automationgateway_pg_username` – учетная запись для доступа к базе данных;
- `automationgateway_pg_password` – пароль учетной записи базы данных;
- `automationgateway_pg_sslmode` – режим установления защищенного соединения.

Автоматизация процессов

На этом шаге отредактируйте описание узлов плоскостей управления и исполнения.

Система автоматизации процессов является наиболее развитой и вариативной, особенно в топологии уровня предприятия. Поэтому описание ее инвентаря существенно зависит от топологических особенностей. С помощью сети Automation Mesh система объединяет плоскости управления и исполнения.

Общие параметры

Учетные данные системного администратора Automation Controller необходимы для прямого обращения к Automation Controller. Это обеспечивает обратную совместимость с версией 1 платформы.

Предупреждение

В текущей версии платформы допускается только учетная запись «admin», которая назначается по умолчанию.

В глобальных переменных укажите только пароль для учетной записи admin:

INI

```ini
[all:vars]
admin_password='p@ssw0rd!'
```

YAML

```yaml
all:
  vars:
    admin_password: p@ssw0rd!
```

Система из одного управляющего и одного исполняющего узла

Для создания сети Mesh из одного управляющего и одного исполняющего узла опишите соответствующий инвентарь следующим образом:

- В группу `automationcontroller` добавьте запись об управляющем узле.

- Создайте группу execution_nodes и добавьте в нее запись об исполняющем узле.
- Создайте для группы automationcontroller переменные node_type и peers со значениями control и execution_nodes соответственно.

Пример описания системы из одного управляющего и одного исполняющего узлов:

INI

```
# Control plane
[automationcontroller]
ctrl1.example.com

# Control plane variables
[automationcontroller:vars]
node_type='control'
peers='execution_nodes'

# Execution plane
[execution_nodes]
exec1.example.com
```

YAML

```
---
# Control plane
automationcontroller:
  hosts:
    ctrl1.example.com:
  vars:
    node_type: control
    peers: execution_nodes

# Execution plane
execution_nodes:
  hosts:
    exec1.example.com:
```

Переменная peers со значением execution_nodes указывает, что управляющие узлы из группы automationcontroller должны связываться с узлами группы execution_nodes. В этом примере не использованы промежуточные узлы (hop nodes).

Система из управляющих и исполняющих узлов

Отказоустойчивая конфигурация требует наличия в системе по крайней мере двух управляющих и двух исполняющих узлов. Для этого опишите инвентарь следующим образом:

1. Добавьте в группу automationcontroller записи об управляющих узлах.
2. Создайте группу execution_nodes и добавьте в нее записи об исполняющих узлах.
3. Создайте для группы automationcontroller переменные node_type и peers со значениями control и execution_nodes соответственно.

Пример описания системы из двух управляющих и двух исполняющих узлов:

INI

```
# Control plane
[automationcontroller]
ctrl1.example.com
ctrl2.example.com

# Control plane variables
[automationcontroller:vars]
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
node_type='control'
peers='execution_nodes'

# Execution plane
[execution_nodes]
exec-1.example.com
exec-2.example.com
```

YAML

```
---
# Control plane
automationcontroller:
  hosts:
    ctrl1.example.com:
    ctrl2.example.com:
  vars:
    node_type: control
    peers: execution_nodes

# Execution plane
execution_nodes:
  hosts:
    exec-1.example.com:
    exec-2.example.com:
```

Все управляющие узлы здесь имеют прямой доступ ко всем исполняющим узлам.

> **Примечание**
>
> При наличии более одного узла в группе automationcontroller учитывайте *принцип главного узла*.

Кластер с удаленным исполнением

Конфигурация с промежуточными узлами позволяет подключаться через них к удаленным исполняющим узлам.

Пример отказоустойчивого кластера с промежуточным узлом hop1, через который выполняется подключение к исполняющему узлу exec3:

INI

```
# Control plane
[automationcontroller]
ctrl1.example.com
ctrl2.example.com

[automationcontroller:vars]
node_type='control'
peers='instance_group_local'

# Execution plane
[execution_nodes]
exec1.example.com
exec2.example.com
exec3.example.com
hop1.example.com

# Execution nodes with direct connection to control nodes
[instance_group_local]
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
exec1.example.com
exec2.example.com

# Hop node
[hop]
hop1.example.com

[hop:vars]
# Direct connection to control nodes
node_type='hop'
peers='automationcontroller'

# Remote execution node
[instance_group_remote]
exec3.example.com

# Direct connection to hop nodes
[instance_group_remote:vars]
peers='hop'
```

YAML

```
---
# Control plane
automationcontroller:
  hosts:
    ctrl1.example.com:
    ctrl2.example.com:
  vars:
    node_type: control
    peers: instance_group_local

# Execution plane
execution_nodes:
  hosts:
    exec1.example.com:
    exec2.example.com:
    exec3.example.com:
    hop1.example.com:

# Execution nodes with direct connection to control nodes
instance_group_local:
  hosts:
    exec1.example.com:
    exec2.example.com:

# Hop node
hop:
  hosts:
    hop1.example.com:
  vars:
    node_type: hop
    peers: automationcontroller

# Remote execution node
instance_group_remote:
  hosts:
    exec3.example.com:
  vars:
    peers: hop
```

Особенности этой топологии:

- Управляющие узлы должны связываться с исполняющими узлами из группы

instance_group_local напрямую, без промежуточных узлов.

- Исполняющие узлы из группы instance_group_remote (в примере один узел) связываются с промежуточными узлами группы hop (в примере один узел), которые в свою очередь связываются с управляющими узлами в группе automationcontroller.

Главный узел

При первом запуске установщика узел, указанный первым в группе automationcontroller описания инвентаря, становится главным узлом (primary node).

К главному узлу предъявляются следующие требования:

- При развертывании или обновлении узлов Automation Controller главный узел всегда должен быть доступен.
- Запись о главном узле в инвентаре установщика должна размещаться на первой позиции в группе automationcontroller.

Контент автоматизации

На этом шаге отредактируйте описание узлов реестра контента – Private Automation Hub.

Private Automation Hub можно развернуть на одном или нескольких узлах.

Развертывание Private Automation Hub на одном узле

Для развертывания Private Automation Hub на одном узле выполните следующие действия:

1. Создайте группу automationhub и добавьте в нее сведения об узле:

INI

```
[automationhub]
hub.example.com
```

YAML

```
automationhub:
  hosts:
    hub.example.com:
```

2. В глобальных переменных automationhub_admin_user и automationhub_admin_password укажите название учетной записи и пароль администратора:

INI

```
[all:vars]
automationhub_admin_user='<admin_user>'
automationhub_admin_password='<admin_password>'
```

YAML

```
all:
  vars:
    automationhub_admin_user: <admin_user>
    automationhub_admin_password: <admin_password>
```

Фрагмент файла инвентаря для развертывания Private Automation Hub на одном узле:

INI

```
[automationhub]
hub.example.com

[all:vars]
automationhub_admin_user='admin'
automationhub_admin_password='automationhub'
```

YAML

```
automationhub:
  hosts:
    hub.example.com:

all:
  vars:
    automationhub_admin_user: admin
    automationhub_admin_password: automationhub
```

Отказоустойчивая конфигурация

Для развертывания отказоустойчивой структуры Private Automation Hub необходимо выполнение следующих условий:

- Использование внешнего отказоустойчивого сервиса PostgreSQL. Предполагается, что такой сервис уже настроен и доступен по доменному имени, например `database.example.com` через порт 5432.
- Развертывание Private Automation Hub не менее чем на двух узлах.
- Использование внешнего хранилища контента типа S3.

С учетом описанных требований, для развертывания отказоустойчивой конфигурации Private Automation Hub выполните следующие действия:

1. В группе `automationhub` перечислите все узлы, на которых необходимо развернуть Private Automation Hub:

   INI

   ```
   [automationhub]
   hub1.example.com
   hub2.example.com
   ```

   YAML

   ```
   automationhub:
     hosts:
       hub1.example.com:
       hub2.example.com:
   ```

2. В переменных группы `automationhub` укажите параметры подключения к хранилищу S3, который будет использоваться для хранения содержимого:

   INI

   ```
   [automationhub:vars]
   s3_endpoint_url='https://s3-storage.example.com'
   s3_region_name='ru-central1'
   s3_access_key='<access_key>'
   s3_secret_key='<secret_key>'
   s3_bucket_name='private-hub-example-com'
   ```

YAML

```yaml
automationhub:
  vars:
    s3_endpoint_url: https://s3-storage.example.com
    s3_region_name: ru-central1
    s3_access_key: <access_key>
    s3_secret_key: <secret_key>
    s3_bucket_name: private-hub-example-com
```

3. В глобальных переменных укажите следующие данные:
   - параметры подключения к СУБД;
   - название учетной записи и пароль администратора;
   - URL, который пользователи будут использовать для доступа к Private Automation Hub.

INI

```ini
[all:vars]
automationhub_pg_host='database.example.com'
automationhub_pg_port=5432
automationhub_pg_database='automationhub'
automationhub_pg_username='admin'
automationhub_pg_password='automationhub'

automationhub_admin_user='admin'
automationhub_admin_password='automationhub'
```

YAML

```yaml
all:
  vars:
    automationhub_pg_host: database.example.com
    automationhub_pg_port: 5432
    automationhub_pg_database: automationhub
    automationhub_pg_username: admin
    automationhub_pg_password: automationhub

    automationhub_admin_user: admin
    automationhub_admin_password: automationhub
```

Фрагмент файла инвентаря с примером отказоустойчивой конфигурации Private Automation Hub:

INI

```ini
[automationhub]
hub1.example.com
hub2.example.com

[automationhub:vars]
s3_endpoint_url='https://s3-storage.example.com'
s3_region_name='ru-central1'
s3_access_key='<access_key>'
s3_secret_key='<secret_key>'
s3_bucket_name='private-hub-example-com'

[database]

[all:vars]
automationhub_admin_user='admin'
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
automationhub_admin_password='automationhub'
automationhub_pg_host='database.example.com'
automationhub_pg_port=5432
```

YAML

```
automationhub:
  hosts:
    hub1.example.com:
    hub2.example.com:
  vars:
    s3_endpoint_url: https://s3-storage.example.com
    s3_region_name: ru-central1
    s3_access_key: <access_key>
    s3_secret_key: <secret_key>
    s3_bucket_name: private-hub-example-com

database:

all:
  vars:
    automationhub_admin_user: admin
    automationhub_admin_password: automationhub
    automationhub_pg_host: database.example.com
    automationhub_pg_port: 5432
```

Подробное описание настроек и их возможные значения см. в *справочнике*.

Управление событиями

На этом шаге отредактируйте описание узлов обработки событий – контроллер Event-Driven Automation.

Контроллер Event-Driven Automation может состоять из узлов трех типов. Тип узла определяется параметром eda_node_type:

- `api` обрабатывает запросы HTTP REST API;
- `worker` управляет проектами, активациями проектов и их исполнением;
- `hybrid` (default) выполняет функции обоих отмеченных ранее типов.

Для развертывания контроллера Event-Driven Automation выполните следующие действия:

1. Создайте группу `automationedacontroller` и добавьте в нее сведения об узлах:

   INI

   ```
   [automationedacontroller]
   edac1.example.com
   edac2.example.com
   ```

   YAML

   ```
   automationedacontroller:
     hosts:
       edac1.example.com:
       edac2.example.com
   ```

   По умолчанию оба узла будут гибридными. При расширении платформы можно добавить дополнительные узлы, указав их тип. Увеличение количества узлов `api` повысит возможность обрабатывать большее количество одновременно работающих пользователей или программ. Увеличение количества узлов `worker` повысит производительность обработки событий.

2. В глобальных переменных укажите следующие данные:
   - параметры подключения к СУБД;
   - пароль администратора.

INI

```
[all:vars]
automationedacontroller_admin_password='admin_password'

automationedacontroller_pg_host='database.example.com'
automationedacontroller_pg_port=5432
automationedacontroller_pg_database='automationedacontroller'
automationedacontroller_pg_username='automationedacontroller'
automationedacontroller_pg_password='edapassword'
automationedacontroller_pg_sslmode='prefer'
```

YAML

```
---
all:
  vars:
    automationedacontroller_pg_host: database.example.com
    automationedacontroller_pg_port: 5432
    automationedacontroller_pg_database: automationedacontroller
    automationedacontroller_pg_username: automationedacontroller
    automationedacontroller_pg_password: edapassword
    automationedacontroller_pg_sslmode: prefer

    automationedacontroller_admin_username: admin
    automationedacontroller_admin_password: admin_password
```

Подробное описание настроек и их возможные значения см. в *справочнике*.

Сертификаты TLS

На этом шаге компоненты платформы на использование сертификатов TLS.

Сертификаты TLS необходимы для защиты данных, передаваемых по протоколу HTTP. В Astra Automation они настраиваются и применяются через специальные переменные в описании инвентаря, используемом в процессе развертывания.

Классификация

Платформа поддерживает различные варианты сертификатов, каждый из которых имеет особенности настройки:

- Сертификаты, подписанные публичным удостоверяющим центром CA (Certificate Authority):
  - выданы доверенными публичными центрами сертификации (Let’s Encrypt, DigiCert, GlobalSign и др.);
  - им доверяют все браузеры и клиенты HTTP;
  - подходят для публично доступных сервисов, то есть особенно важны для шлюза платформы.
- Сертификаты, подписанные собственным (корпоративным) удостоверяющим центром (Custom/Corporate CA) в организации, которая использует собственную структуры PKI (Public Key Infrastructure):
  - выданы внутренним центром сертификации организации;
  - требуют установки корневого CA-сертификата на клиентские системы;

  - подходят для внутренних корпоративных систем.
- Самоподписанные сертификаты (Self-signed):
  - сертификаты, подписанные собственным закрытым ключом;
  - им не доверяют браузеры по умолчанию (предупреждения безопасности);
  - подходят для тестирования и разработки.

Применение сертификатов от публичного удостоверяющего центра

Пример описания инвентаря для сертификатов от публичного удостоверяющего центра, например Let's Encrypt:

```
[all:vars]
# SSL certificate settings for public CA
# No custom_ca_cert required for public CA certificates

# Platform Gateway SSL certificate
automationgateway_ssl_cert=/etc/letsencrypt/live/gateway.example.com/fullchain.pem
automationgateway_ssl_key=/etc/letsencrypt/live/gateway.example.com/privkey.pem

# Automation Controller SSL certificate
web_server_ssl_cert=/etc/letsencrypt/live/controller.example.com/fullchain.pem
web_server_ssl_key=/etc/letsencrypt/live/controller.example.com/privkey.pem

# Automation Hub SSL certificate
automationhub_ssl_cert=/etc/letsencrypt/live/hub.example.com/fullchain.pem
automationhub_ssl_key=/etc/letsencrypt/live/hub.example.com/privkey.pem

# Event-Driven Automation SSL certificate
automationedacontroller_ssl_cert=/etc/letsencrypt/live/eda.example.com/fullchain.pem
automationedacontroller_ssl_key=/etc/letsencrypt/live/eda.example.com/privkey.pem
```

Особенности применения публичных сертификатов:

- используйте файл, например, `fullchain.pem`, который содержит как серверный сертификат, так и промежуточные сертификаты;
- переменная `custom_ca_cert` не требуется, так как корневые CA уже доверяются системой;
- сертификаты имеют ограниченный срок действия (например, 90 дней);
- требуется автоматизированное обновление сертификатов.

Применение сертификатов собственного удостоверяющего центра

Пример описания инвентаря для корпоративного CA:

```
[all:vars]
# SSL certificate settings for custom/corporate CA

# Custom CA certificate (root or intermediate)
custom_ca_cert=/path/to/corporate-ca-chain.crt

# Platform Gateway SSL certificate
automationgateway_ssl_cert=/path/to/gateway.corp.example.com.crt
automationgateway_ssl_key=/path/to/gateway.corp.example.com.key

# Automation Controller SSL certificate
web_server_ssl_cert=/path/to/controller.corp.example.com.crt
web_server_ssl_key=/path/to/controller.corp.example.com.key

# Automation Hub SSL certificate
automationhub_ssl_cert=/path/to/hub.corp.example.com.crt
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
automationhub_ssl_key=/path/to/hub.corp.example.com.key

# Event-Driven Automation SSL certificate
automationedacontroller_ssl_cert=/path/to/eda.corp.example.com.crt
automationedacontroller_ssl_key=/path/to/eda.corp.example.com.key
```

Особенности корпоративных CA-сертификатов:

- обязательно используйте параметр `custom_ca_cert`, указывающий путь к файлу с сертификатами;
- файл с сертификатами должен содержать всю цепочку от сервера до корневого сертификата CA;
- серверные сертификаты должны быть в формате PEM.

Формат файла цепочки CA:

```
-----BEGIN CERTIFICATE-----
[Intermediate CA Certificate]
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
[Root CA Certificate]
-----END CERTIFICATE-----
```

Применение самоподписанных сертификатов

Пример описания инвентаря для самоподписанных сертификатов с использованием собственного корневого сертификата:

```
[all:vars]
# SSL certificate settings for self-signed certificates

# Custom CA certificate (self-signed root)
custom_ca_cert=/path/to/self-signed-ca.crt

# Platform Gateway SSL certificate
automationgateway_ssl_cert=/path/to/gateway.local.example.com.crt
automationgateway_ssl_key=/path/to/gateway.local.example.com.key

# Automation Controller SSL certificate
web_server_ssl_cert=/path/to/controller.local.example.com.crt
web_server_ssl_key=/path/to/controller.local.example.com.key

# Automation Hub SSL certificate
automationhub_ssl_cert=/path/to/hub.local.example.com.crt
automationhub_ssl_key=/path/to/hub.local.example.com.key

# Event-Driven Automation SSL certificate
automationedacontroller_ssl_cert=/path/to/eda.local.example.com.crt
automationedacontroller_ssl_key=/path/to/eda.local.example.com.key

# Alternative: Regenerate all self-signed certificates automatically
# aap_service_regen_cert=true
```

Особенности самоподписанных сертификатов с использованием собственного корневого сертификата:

- Требуют ручного добавления в доверенные сертификаты на клиентских системах.
- Браузеры будут показывать предупреждения безопасности.
- Не подходят для производственных систем.

Если в дальнейшем потребуется регенерация самоподписанных сертификатов, то используйте переменную `aap_service_regen_cert` при повторном запуске `aa-setup`:

```
[all:vars]
# Automatically regenerate self-signed certificates for all services
aap_service_regen_cert=true
```

Этот вариант требует всего лишь одну строку настройки. Созданные таким образом сертификаты имеют следующие особенности и ограничения в применении:

- простота развертывания: автоматически создаются утилитой установки при указании `aap_service_regen_cert=true` без требования ручной генерации, предварительной подготовки файлов или знания команд OpenSSL;
- комплексное покрытие: генерируются для всех компонентов платформы одновременно (Gateway, Controller, Hub, EDA) с единым корневым CA-сертификатом и автоматическим размещением в соответствующих каталогах;
- гибкость управления: могут быть перегенерированы в любое время повторным запуском утилиты установки с тем же параметром, что обеспечивает простое обновление при необходимости;
- оптимизация для тестирования: используют стандартные параметры шифрования с достаточно длительным сроком действия, автоматически включают необходимые Subject Alternative Names (SAN) и подходят для изолированных тестовых сред;
- надежность конфигурации: обеспечивают консистентность параметров шифрования между всеми компонентами и позволяют избежать ошибок, связанных с неправильными путями к файлам или форматами сертификатов.

Рекомендации по выбору типа сертификатов

В зависимости от области применения и корпоративной политики защиты данных выберите один из следующих вариантов настройки сертификатов TLS:

Производственные системы (публичный доступ):

- используйте сертификаты публичных CA;
- настройте автоматическое обновление сертификатов;
- используйте проверку доступа для автоматизации обновления сертификатов.

Корпоративные системы (внутренняя сеть):

- используйте корпоративный CA при наличии инфраструктуры PKI;
- убедитесь, что корневые сертификаты установлены на всех клиентских системах;
- настройте политики истечения сертификатов.

Тестирование и разработка:

- самоподписанные сертификаты подходят для изолированных тестовых сред;
- используйте `aap_service_regen_cert=true` для упрощения развертывания;
- добавьте исключения в браузеры для тестирования.

Процесс подготовки

На этом этапе выполните следующие подготовительные действия, если не используется режим автоматической регенерации:

1. Подготовьте сертификаты и ключи согласно выбранному типу.
2. Скопируйте файлы на установочный узел.
3. Настройте переменные в файле инвентаря.

Параметры СУБД

На этом шаге отредактируйте описание узла PostgreSQL, если он развертывается средствами платформы, или реквизиты доступа к внешней СУБД.

В составе платформы Astra Automation используются две СУБД:

- PostgreSQL – реляционная;
- Redis – нереляционная.

PostgreSQL

СУБД PostgreSQL необходима для хранения данных компонентов платформы. Возможно использование уже существующего кластера PostgreSQL, например, развернутого в одном из облачных сервисов управляемых баз данных. Если отдельного кластера PostgreSQL нет, можно развернуть одиночный сервер СУБД средствами платформы на одном из узлов, выделенных для Astra Automation.

Примечание

СУБД PostgreSQL используется платформой для хранения данных, но не является ее частью. Настройка отказоустойчивой конфигурации PostgreSQL в этом руководстве не рассматривается. Для получения соответствующих инструкций обращайтесь к документации PostgreSQL[25].

Развертывание средствами платформы

Чтобы средствами платформы развернуть СУБД на отдельном узле, выполните следующие действия:

1. В описании инвентаря создайте группу `database` и добавьте в нее сведения об узле, например:

   INI

   ```
   [database]
   database.example.com
   ```

   YAML

   ```
   database:
     hosts:
       database.example.com:
   ```

2. В глобальных переменных укажите значения параметров подключения к СУБД.

При развертывании СУБД средствами платформы в систему автоматически устанавливается расширение `hstore`. Дополнительные действия не требуются.

Внешняя СУБД

Если у вас уже есть кластер PostgreSQL, выполните следующие действия:

1. Создайте в кластере PostgreSQL пользователей и принадлежащие им базы данных для используемых компонентов платформы.
2. Убедитесь, что в описании инвентаря группа `database` существует, но не содержит узлов.
3. В глобальных переменных укажите значения параметров подключения к СУБД.

[25] https://www.postgresql.org/docs/current/high-availability.html

4. Проверьте, установлено ли расширение `hstore` для базы данных Private Automation Hub:

```
psql -d <pah_database> -c "SELECT * FROM pg_available_extensions WHERE name='hstore
↪';"
```

   Здесь <pah_database> – название базы данных.

Если расширение не установлено, выполните следующие действия:

1. Установите дополнительные пакеты для PostgreSQL:

```
sudo apt install postgresql-contrib
```

2. Создайте расширение `hstore` в базе данных Private Automation Hub:

```
psql -d <pah_database> -c "CREATE EXTENSION hstore;"
```

3. Проверьте доступность расширения с помощью запроса:

```
psql -d <pah_database> -c "SELECT * FROM pg_available_extensions WHERE name='hstore
↪';"
```

   Если расширение установлено и настроено, в терминал выводится таблица следующего вида:

```
name   | default_version | installed_version | comment
-------+-----------------+-------------------+------------------------------------
↪----------------
hstore |      1.7        |       1.7         | data type for storing sets of (key,␣
↪value) pairs
(1 row)
```

Параметры подключения к PostgreSQL

Параметры подключения Platform Gateway, Automation Controller, Private Automation Hub и контроллера Event-Driven Automation к СУБД задаются в глобальных переменных:

INI

```
# ...
[all:vars]
# Platform Gateway
automationgateway_pg_host='database.example.com'
automationgateway_pg_port=5432
automationgateway_pg_database='automationgateway'
automationgateway_pg_username='automationgateway'
automationgateway_pg_password='gwpassword'

# Automation Controller
pg_host='database.example.com'
pg_port=5432
pg_database='awx'
pg_username='automationcontroller'
pg_password='ctrlpassword'

# Automation Hub
automationhub_pg_host='database.example.com'
automationhub_pg_port=5432
automationhub_pg_database='automationhub'
automationhub_pg_username='automationhub'
automationhub_pg_password='hubpgpassword'

# Event-Driven Automation controller
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
automationedacontroller_pg_host='database.example.com'
automationedacontroller_pg_port=5432
automationedacontroller_pg_database='automationedacontroller'
automationedacontroller_pg_username='automationedacontroller'
automationedacontroller_pg_password='edapassword'

# PostgreSQL TLS
postgres_use_ssl='true'
postgres_ssl_cert='/path/to/ssl/cert'
postgres_ssl_key='/path/to/ssl/key'
```

YAML

```
---
# ...
all:
  vars:
    # Platform Gateway
    automationgateway_pg_host: database.example.com
    automationgateway_pg_port: 5432
    automationgateway_pg_database: automationgateway
    automationgateway_pg_username: automationgateway
    automationgateway_pg_password: gwpassword

    # Automation Controller
    pg_host: database.example.com
    pg_port: 5432
    pg_database: awx
    pg_username: automationcontroller
    pg_password: ctrlpassword

    # Automation Hub
    automationhub_pg_host: database.example.com
    automationhub_pg_port: 5432
    automationhub_pg_database: automationhub
    automationhub_pg_password: hubpgpassword

    # Event-Driven Automation controller
    automationedacontroller_pg_host: database.example.com
    automationedacontroller_pg_port: 5432
    automationedacontroller_pg_database: automationedacontroller
    automationedacontroller_pg_username: automationedacontroller
    automationedacontroller_pg_password: edapassword

    # PostgreSQL TLS
    postgres_use_ssl: true
    postgres_ssl_cert: /path/to/ssl/cert
    postgres_ssl_key: /path/to/ssl/key
```

Здесь:

- `pg_host`, `automationhub_pg_host` и `automationedacontroller_pg_host`– IP-адреса или FQDN серверов СУБД для Platform Gateway, Automation Controller, Private Automation Hub и контроллера Event-Driven Automation соответственно.

  Значение по умолчанию: пустая строка (используется локальное подключение).

- `automationgateway_pg_port`, `pg_port`, `automationhub_pg_port` и `automationedacontroller_pg_port` – порты для подключения к СУБД для Platform Gateway, Automation Controller, Private Automation Hub и контроллера Event-Driven Automation соответственно.

  Значение по умолчанию: 5432.

- `automationgateway_pg_database`, `pg_database`, `automationhub_pg_database` и

automationedacontroller_pg_database – названия БД для Platform Gateway, Automation Controller, Private Automation Hub и контроллера Event-Driven Automation соответственно.

- pg_username, automationhub_pg_username и automationedacontroller_pg_username – названия учетных записей пользователей БД Platform Gateway, Automation Controller, Private Automation Hub и контроллера Event-Driven Automation соответственно.
- pg_password, automationhub_pg_password и automationedacontroller_pg_password – пароли пользователей БД Platform Gateway, Automation Controller, Private Automation Hub и контроллера Event-Driven Automation соответственно.

> **Примечание**
>
> При использовании СУБД, созданной установщиком платформы, пароль указывать не требуется.

- postgres_use_ssl – использование SSL при подключении к БД Automation Controller:
  - true – включено;
  - false – выключено.
- postgresql_ssl_cert – полный путь к файлу сертификата, который используется для защиты подключения к серверу СУБД PostgreSQL, развернутому установщиком платформы.
- postgresql_ssl_key – полный путь к файлу ключа сертификата, который используется для защиты подключения к серверу СУБД PostgreSQL, развернутому установщиком платформы.

При использовании SSL по умолчанию используется режим prefer, который можно изменить для каждого компонента отдельно с помощью переменных automationgateway_pg_sslmode, pg_sslmode, automationhub_pg_sslmode и automationedacontroller_pg_sslmode соответственно. Они могут принимать следующие значения:

- prefer – подключение к БД будет защищено с помощью SSL, если использование шифрования поддерживается настройками сервера;
- verify-full – защита подключения с помощью SSL необходима, производится строгая верификация сервера.

Подробности о поддерживаемых параметрах шифрования см. в документации PostgreSQL[26].

Redis

Компоненты Astra Automation используют для кеширования данных сервис Redis, который можно настроить на один из режимов работы в зависимости от выбранной топологии:

- standalone – используют в базовой топологии, когда каждый компонент устанавливается на собственном узле, где также устанавливается служба Redis. Этот режим надо указывать явно:

```
[all: vars]
redis_mode=standalone
```

- Режим кластера (по умолчанию) – используют в топологии уровня предприятия. В этом режиме надо перечислить все узлы кластера (не менее шести) в том составе, который представлен в рекомендуемой топологии (нельзя включать узлы из плоскостей управления и исполнения):

[26] https://www.postgresql.org/docs/current/libpq-ssl.html

```
# Redis cluster
[redis]
hub1.example.com
hub2.example.com
gw1.example.com
gw2.example.com
eda1.example.com
eda2.example.com
```

Пользуются этим кластером только узлы Platform Gateway и EDA. Остальные узлы пользуются собственным сервисом Redis.

Без доступа к интернету

Развертывание платформы в окружении, не имеющем доступа к интернету, имеет ряд особенностей.

Развертывание зеркал репозиториев

Поскольку при установке в окружении без доступа к интернету невозможно использовать репозитории ПАО Группа Астра, следует самостоятельно создать их копии так, чтобы они были доступны для узлов, используемых для развертывания платформы.

Пошаговые инструкции по созданию зеркал приведены в статье справочного центра Создание локальных и сетевых репозиториев[27].

При использовании ОС Astra Linux Special Edition 1.8 локально должны быть развернуты копии основного (main) и расширенного (extended) репозиториев.

Важно

Версии используемых репозиториев должны соответствовать версии ОС, установленной на узлах платформы.

Подготовка узлов

Общие настройки узлов в окружении без доступа к интернету не отличаются от настроек в окружении с доступом к интернету:

1. Обеспечьте сетевую связность всех узлов.
2. (Необязательно) Обеспечьте конвертацию названий узлов в IP-адреса (с помощью `/etc/hosts` или DNS).
3. Обеспечьте доступ к узлам по SSH с использованием ключей согласно *инструкции*.
4. Разрешите выполнение команд с использованием sudo без ввода пароля:

```
sudo astra-sudo-control disable
```

Настройка установочного узла

Для настройки установочного узла выполните следующие действия:

1. Добавьте в файл `/etc/apt/sources.list` ссылки на развернутые зеркала репозиториев, например:

```
deb http://192.168.0.1/astra/frozen/1.8_x86-64/1.8.1uu1/repository-main     1.8_x86-
↪64 main non-free contrib
deb http://192.168.0.1/astra/frozen/1.8_x86-64/1.8.1uu1/repository-extended 1.8_x86-
↪64 main non-free contrib
```

[27] https://wiki.astralinux.ru/pages/viewpage.action?pageId=61575159

2. Обновите список доступных пакетов:

```
sudo apt update
```

3. Загрузите архив установочного пакета Astra Automation на установочный узел.

> **Примечание**
>
> Архив доступен для загрузки в Личном кабинете[28] при наличии действующей лицензии на продукт.

4. Создайте каталог для файлов установщика, например:

```
sudo mkdir -p /opt/rbta/aa/astra-automation-setup
```

5. Распакуйте содержимое архива в созданный каталог, например:

```
sudo tar -xvzf <archive>.tgz -C /opt/rbta/aa/astra-automation-setup
```

Описание инвентаря

При описании инвентаря выполните следующие действия:

1. Опишите параметры платформы согласно инструкциям из раздела *Описание инвентаря*.
2. Добавьте в глобальные переменные описания инвентаря следующие настройки:

INI

```
[all:vars]
bundle_install='true'
bundle_dir='/path/to/aa-setup/files'
```

YAML

```
all:
  vars:
    bundle_install: true
    bundle_dir: /path/to/aa-setup/files
```

Здесь:

- `bundle_dir` – путь к каталогу с файлами установщика платформы.

> **Примечание**
>
> Значение этого параметра необходимо задавать только в том случае, когда путь к каталогу с файлами установщика платформы отличается от `/opt/rbta/aa/astra-automation-setup/`.

> **Примечание**
>
> Подробное описание параметров см. в *справочнике*.

[28] https://lk.astra.ru

Справочные данные

Для развертывания платформы и управления ею в процессе эксплуатации используются специальные утилиты. Здесь приведены справочные данные по их назначению и использованию.

aa-setup

Утилита `aa-setup` используется для развертывания платформы и предоставляет различные возможности по управлению ею.

Назначение

Утилита `aa-setup` выполняет следующие действия:

- развертывает платформу в виде нескольких связанных компонентов;
- обновляет версии и структуру компонентов платформы;
- выполняет резервное копирование и восстановление базы данных;
- генерирует и распространяет секретный ключ;
- проверяет версии Astra Automation, доступные в репозитории;
- удаляет Astra Automation.

Установка и запуск

Для установки утилиты `aa-setup` выполните следующие действия:

1. В каталоге `/etc/apt/sources.list.d/` создайте файл `astra-automation.list` со ссылкой на репозиторий Astra Automation:

   ```
   deb https://dl.astralinux.ru/aa/aa-debs-for-alse-1.8 <version> main
   ```

   Вместо <version> необходимо подставить версию устанавливаемой платформы, например, `2.0`.

   Доступные версии продукта опубликованы в таблице История обновлений[29].

2. Обновите список доступных пакетов:

   ```
   sudo apt update
   ```

3. Выполните команду установки:

   ```
   sudo apt install astra-automation-setup --yes
   ```

Описание аргументов

Запустить утилиту можно в каталоге `/opt/rbta/aa/astra-automation-setup/` со следующими аргументами:

`-h`, `--help`
: Выводит информацию о доступных опциях и соответствующих аргументах.

`-i` <inventory_path>, **`--inventory`** <inventory_path>
: Путь к файлу инвентаризации. Значение по умолчанию: `./inventory`.

`-p` <log_path>, **`--log-path`** <log_path>
: Путь к альтернативному каталогу для сохранения журнала установки. Текущий пользователь должен иметь привилегию на создание файлов и запись в этом каталоге.

  Значение по умолчанию: `./` (текущий каталог).

[29] https://docs.astra-automation.ru/latest/releases/#releases-history

-k, --generate-key

Генерирует и распространяет новый секретный ключ, используемый для выполнения следующих задач:

- шифрование данных пользовательских сессий;
- генерация токенов для паролей;
- защита данных контроллера.

-b, --backup

Сохраняет в виде *резервной копии* данные, необходимые для полного восстановления платформы на узлах, описанных в инвентаре установщика.

-r, --restore

Использует резервную копию данных для восстановления платформы на узлах, описанных в инвентаре установщика.

-d, --debug

Включает режим отладки для более подробного вывода информации о процессе.

-u, --uninstall

Запускает процесс удаления компонентов Astra Automation, перечисленных в файле инвентаризации.

-y, --force-yes

Ключ отключает запрос на подтверждение выполнения операции, подразумевая, что ответ всегда Да (Yes).

--force-postgres-removal

Запускает процесс удаления PostgreSQL.

--plain

Отключает расширенный вариант текстового пользовательского интерфейса.

--repo-url <repo_url>

URL репозитория с пакетами Astra Automation.

Эта настройка может быть полезна в том случае, когда вы хотите использовать собственное зеркало репозитория Astra Automation вместо серверов ПАО Группа Астра, например:

```
sudo ./aa-setup --repo-url https://example.com/aa-debs-for-alse-1.8
```

Предупреждение

По умолчанию `aa-setup` настраивает узлы платформы на использование репозитория Astra Automation для соответствующей версии Astra Linux Special Edition. При использовании опции `--repo-url` на всех узлах платформы используется одна и та же ссылка на репозиторий, не зависящая от версии установленной ОС. Использование репозитория Astra Automation для версии Astra Linux Special Edition, отличающейся от установленной на узле, приведет к нарушению работы платформы либо ошибкам при ее развертывании и обновлении.

--default-job-ee <ee_url>

Указывает на образ контейнера, который будет использован в качестве среды исполнения по умолчанию (Default execution environment). Например, эта команда задает в качестве среды исполнения по умолчанию образ `hub.astralinux.ru/aa-1.2/aa-full-ee:latest`:

```
sudo ./aa-setup --default-job-ee hub.astralinux.ru/aa-1.2/aa-full-ee:latest
```

--component <component_name>

Указывает альтернативный компонент репозитория. Например, если необходимо использовать пакеты, размещенные в компоненте `testing`, воспользуйтесь командой:

```
sudo ./aa-setup --component testing
```

--product-version <product_version>

Указывает определенную версию Astra Automation. По умолчанию устанавливается новейшая версия. Для установки определенной версии <product_version>, например, `2.0` воспользуйтесь командой:

```
sudo ./aa-setup --product-version <product_version>
```

-v, --version

Показывает версию утилиты `aa-setup`.

--upgrade

Обновляет компоненты Astra Automation до новейшей версии. Может быть использован совместно с аргументом `--product-version` для обновления до определенной версии.

--check-releases

Выводит на экран терминала версии Astra Automation, доступные в репозитории.

-i, --inventory

Путь к файлу инвентаря, который утилита `aa-setup` должна использовать вместо инвентаря по умолчанию (`inventory`).

В зависимости от используемого формата файл должен иметь одно из указанных расширений:

- INI:
  - `.ini`;
  - `.cfg`;
- YAML:
  - `.yml`;
  - `.yaml`.

Примечание

Если расширение не указано, файл интерпретируется как имеющий формат INI.

Дополнительные аргументы

Утилита `aa-setup` позволяет передать дополнительные аргументы для команды `ansible-playbook`. Укажите их после `--`, например:

```
sudo ./aa-setup -b -- -vvv --ask-pass
```

Параметры инвентаря

Инвентарь установщика может иметь формат INI или YAML.

Инвентарь в формате INI имеет следующие особенности:

- Для формирования групп узлов используют секции. Название каждой секции обрамляется квадратными скобками `[]`.

```
[automationcontroller]
```

- Параметры отдельных узлов задают в той же строке, что и название узла. Название параметра и его значение указывают в одной строке через знак =. Для разделения нескольких параметров используются символы пробела.

```
[automationcontroller]
node1   ansible_host=192.168.56.1   node_type=control
node2   ansible_host=192.168.56.2   node_type=control
```

- Переменные, общие для всех узлов группы, задают в секции, название которой состоит из названия группы узлов и суффикса `:vars`.

```
[automationcontroller:vars]
ansible_user='administrator'
ansible_ssh_private_key='/path/to/ssh/private/key'
node_type='control'
```

- В секции `[all:vars]` задают *глобальные переменные* – они действуют на все узлы, указанные в инвентаре.

```
[all:vars]
pgdatabase='controllerdb'
```

Инвентарь в формате YAML имеет следующие особенности:

- Для формирования групп узлов использует запись следующего вида:

```
automationcontroller:
  hosts:
```

- Для перечисления узлов используют блоки, а не списки. Название узла указывается в названии блока.

```
automationcontroller:
  hosts:
    node1:
      ansible_host: 192.168.56.1
      ansible_connection: ssh
      node_type: control
    node2:
      ansible_host: 192.168.56.2
      ansible_connection: ssh
      node_type: control
```

- Переменные, общие для всех узлов группы, задают в блоке `<group>.vars`:

```
automationcontroller:
  hosts:
    node1:
      ansible_host: 192.168.56.1
    node2:
      ansible_host: 192.168.56.2
  vars:
    ansible_connection: ssh
    node_type: control
```

- Глобальные переменные задают в блоке `all.vars`:

```
all:
  vars:
    pgdatabase: controllerdb
```

Примечание

С целью сокращения все настройки описываются для инвентаря формата INI.

Переменные компонентов платформы можно задавать на следующих уровнях:

- Для всех узлов.
- Для группы узлов.

  Чтобы задать переменные для всех узлов определенной группы, создайте соответствующую секцию. Название секции формирует по следующему принципу:

  ```
  [<group>:vars]
  ```

  Здесь <group> – название группы узлов.

  Например, чтобы задать переменные для узлов из группы `main` создайте секцию `[main:vars]`:

  ```
  [main]
  node1.example.com   ansible_host=192.168.56.1
  node2.example.com   ansible_host=192.168.56.2
  node3.example.com   ansible_host=192.168.56.3

  [main:vars]
  ansible_user='administrator'
  ansible_ssh_private_key='/path/to/ssh/private/key'
  ```

  Эта запись эквивалентна следующей:

  ```
  [main]
  node1.example.com   ansible_host=192.168.56.1   ansible_user=administrator
  ↪ansible_ssh_private_key=/path/to/ssh/private/key
  node2.example.com   ansible_host=192.168.56.2   ansible_user=administrator
  ↪ansible_ssh_private_key=/path/to/ssh/private/key
  node3.example.com   ansible_host=192.168.56.3   ansible_user=administrator
  ↪ansible_ssh_private_key=/path/to/ssh/private/key
  ```

- Для отдельных узлов в соответствующих строках.

При определении значений настроек используется следующий порядок:

1. глобальные переменные;
2. переменные группы;
3. переменные отдельных узлов.

Более поздние значения перезаписывают более ранние.

Настройка подключения

Тип подключения к узлам платформы задают переменной `ansible_connection`:

- `local` – локальное подключение к узлу. Используют для развертывания компонента платформы на установочном узле (не рекомендуется).
- `ssh` – подключение с использованием протокола SSH (по умолчанию).
- `paramiko` – подключение с использованием протокола Paramiko.

Основные переменные подключения

Для всех типов подключения возможно использование следующих дополнительных настроек:

- `ansible_host` – IP-адрес управляемого узла.

- ansible_port – номер порта, через который следует выполнять подключение к управляемому узлу. При использовании протокола SSH значение по умолчанию – 22.
- ansible_user – название учетной записи, от имени которой следует выполнять подключение к управляемому узлу.
- ansible_password – пароль учетной записи.

Настройка протокола SSH

Для протокола SSH доступны следующие переменные:

- ansible_scp_extra_args – дополнительные аргументы вызова утилиты scp.
- ansible_sftp_extra_args – дополнительные аргументы вызова утилиты sftp.
- ansible_ssh_common_args – дополнительные аргументы вызова утилит ssh, scp и sftp.
- ansible_ssh_executable – путь к исполняемому файлу утилиты ssh, который следует использовать вместо системного.
- ansible_ssh_extra_args – дополнительные аргументы вызова утилиты ssh.
- ansible_ssh_pipelining – сокращение количества операций подключения к управляемым узлам за счет выполнения некоторых модулей без копирования файлов:
  - true – включено;
  - false – выключено (по умолчанию).
- ansible_ssh_private_key_file – путь к файлу приватного ключа SSH.

Повышение привилегий

Для выполнения некоторых операций требуются повышенные привилегии. Способы их получения задают в следующих настройках:

- ansible_become – принудительное использование повышения привилегий;
- ansible_become_exe – путь к исполняемому файлу, используемому для повышения привилегий;
- ansible_become_flags – дополнительные флаги, используемые для выбранного метода повышения привилегий;
- ansible_become_method – метод повышения привилегий;
- ansible_become_user – название учетной записи, используемой для повышения привилегий;
- ansible_become_password – пароль учетной записи, используемой для повышения привилегий.

Настройка окружения управляемых узлов

Параметры выполнения сценариев Ansible на управляемых узлах задают в следующих настройках:

- ansible_shell_type – тип оболочки на управляемом узле:
  - csh;
  - fish;
  - sh (по умолчанию).

Важно

Значение этой переменные нужно изменять только если значением переменные ansible_shell_executable выбрана оболочка, не совместимая с sh.

- `ansible_shell_executable` – путь к командному интерпретатору.

  Важно

  Значение этой переменной нужно изменять только в том случае, когда использование `/bin/sh` по какой-либо причине невозможно.

  Значение по умолчанию: `/bin/sh`.

Группы узлов

Переменные для отдельных компонентов платформы задают в соответствующих секциях:

- Platform Gateway – `[automationgateway]`.
- Automation Controller:
  - `[automationcontroller]` – узлы плоскости управления;
  - `[execution_nodes]` – исполняющие узлы;
  - `[instance_group_<name>]` – группа исполняющих узлов;
  - дополнительные группы для промежуточных и исполняющих узлов.
- Private Automation Hub – `[automationhub]`.
- Контроллер Event-Driven Automation – `[automationedacontroller]`.
- СУБД – `[database]`.

  На указанном узле средствами платформы будет развернута СУБД PostgreSQL, а в ней создана база данных для Automation Controller.

```
[database]
rdbs.example.com    ansible_host=192.168.56.101
```

  Важно

  Узел, используемый для развертывания PostgreSQL, не должен использоваться для других целей, например, запуска заданий или управления контроллером.

  Если для хранения баз данных будет использован внешний PostgreSQL, секцию `[database]` надо создать, но оставить пустой.

Настройка Platform Gateway

Переменные Platform Gateway задают в секциях `[automautomationgateway]`, `[automationgateway:vars]` и `[all:vars]`.

Настройка узлов

В секции `[automationgateway]` задают параметры подключения к узлам Platform Gateway, например:

```
[automationgateway]

gateway1.example.com
gateway2.example.com
```

Учетные данные администратора

Учетные данные администратора Platform Gateway задают следующими глобальными переменными:

- `automationgateway_admin_password` – пароль;
- `automationgateway_admin_username` – название учетной записи.

  Значение по умолчанию – «admin». В текущей версии изменять это значение нельзя.
- `automationgateway_admin_email` – адрес электронной почты.

Сеть

Параметры сети задают следующими глобальными переменными:

- `automationgateway_http_port` – номер порта, на котором Platform Gateway прослушивает входящие HTTP-соединения.

  Значение по умолчанию: `80`.
- `automationgateway_https_port` – номер порта, на котором Platform Gateway прослушивает входящие HTTPS-соединения.

  Значение по умолчанию: 443.
- `automationgateway_ssl_cert` – путь к файлу сертификата TLS, используемого веб-сервером.
- `automationgateway_ssl_key` – путь к файлу ключа сертификата TLS, используемого веб-сервером.
- `automationgateway_ssl_protocol` – протоколы, которые Platform Gateway будет поддерживать при обработке HTTPS-трафика.

  Значение по умолчанию: `TLSv1_2, TLSv1_3`.
- `automationgateway_https_enabled` – включение HTTPS для Platform Gateway.
  - `true` – включено;
  - `false` – выключено.
- `automationgateway_hsts_enabled` – включение HTTP Strict Transport Security (HSTS) для Platform Gateway.
  - `true` – включено;
  - `false` – выключено.
- `automationgateway_hsts_max_age` – максимальная длительность (в секундах), в течение которой HSTS принудительно применяется для Platform Gateway.
- `automationgateway_main_url` – URL, к которому подключаются клиенты и который используют внутренние компоненты платформы для взаимодействия между собой.

  Используют при развертывании кластера, когда требуется использовать URL балансировщика нагрузки вместо серверного адреса компонента. Значение URL должно начинаться с `https://` или `http://`.

СУБД

Параметры подключения Platform Gateway к СУБД задают следующими глобальными переменными:

- `automationgateway_pg_database` – название базы данных.
- `automationgateway_pg_host` – IP-адрес или FQDN узла с СУБД.
- `automationgateway_pg_password` – пароль учетной записи СУБД.

- automationgateway_pg_port – порт СУБД.

  Значение по умолчанию: 5432.

- automationgateway_pg_username – название учетной записи СУБД.
- automationgateway_pg_sslmode – режим использования SSL при подключении к СУБД:
  - prefer (по умолчанию) – подключение к СУБД будет защищено с помощью SSL, если сервер поддерживает шифрование;
  - verify-full – защита подключения с помощью SSL необходима, производится строгая верификация сервера. Соединение не будет установлено, если это условие не выполнено.
- automationgateway_postgres_ssl_cert – путь к файлу сертификата SSL/TLS, используемого для защиты подключения к серверу СУБД PostgreSQL.
- automationgateway_postgres_ssl_key – путь к файлу ключа сертификата SSL/TLS, используемого для защиты подключения к серверу СУБД PostgreSQL.
- automationgateway_postgres_use_ssl – использование SSL для защиты подключения к СУБД:
  - true – включено;
  - false – выключено (по умолчанию).

Redis

Параметры подключения Platform Gateway к Redis задают в следующих глобальных настройках:

- automationgateway_redis_host – адрес узла Redis, используемого Platform Gateway.

  По умолчанию используется первый узел из группы [redis].

- automationgateway_redis_port – номер порта узла Redis, используемого Platform Gateway.
- automationgateway_redis_password – пароль для доступа к Redis.

  По умолчанию используется случайно сгенерированная строка.

- automationgateway_redis_username – название учетной записи пользователя Redis.
- automationgateway_redis_ssl – включение TLS для Redis Platform Gateway.
  - true – включено;
  - false – выключено (по умолчанию).
- automationgateway_redis_cert – путь к файлу сертификата Redis для Platform Gateway.
- automationgateway_redis_key – путь к файлу ключа Redis для Platform Gateway.

gRPC

gRPC является встроенным сервисом, обеспечивающим аутентификацию пользователя через внешние системы (SSO, Single Sign-On) и перенаправление запроса пользователя к требуемому компоненту платформы. Следующие переменные влияют на его работу:

- automationgateway_grpc_processes – количество процессов для обработки gRPC-запросов к Platform Gateway.
- automationgateway_grpc_max_threads – максимальное количество потоков, которые каждый процесс gRPC-сервера может создать для обработки запросов к Platform Gateway.
- automationgateway_grpc_timeout – длительность тайм-аута (в секундах) для запросов, осуществляемых к gRPC-сервису Platform Gateway.

Дополнительные переменные

Следующие переменные используются реже, чем представленные ранее:

- automationgateway_extra_settings – дополнительные переменные для использования Platform Gateway при установке.

  Например: automationgateway_extra_settings: - setting: OAUTH2_PROVIDER['ACCESS_TOKEN_EXPIRE_SECONDS'] value: 600

- automationgateway_nginx_headers – список дополнительных заголовков NGINX для добавления в конфигурацию NGINX Platform Gateway.
- automationgateway_client_max_body_size – максимальный допустимый размер (в байтах) данных, отправляемых в Platform Gateway через NGINX.
- automationgateway_web_workers – количество рабочих процессов для обработки веб-запросов к Platform Gateway.

  По умолчанию: количество ядер CPU, умноженное на два, плюс один.

- automationgateway_firewall_zone – зона сети, к которой применяются правила сетевого фильтра Platform Gateway.

  Контролирует, какие сети могут получать доступ к Platform Gateway на основе уровня доверия зоны.

- automationgateway_web_cert_local – указание того, являются ли файлы сертификатов веб-сервера локальными для установочного узла (true) или находятся на управляемом узле (false).
- automationgateway_db_client_cert_local – указание того, являются ли файлы сертификатов клиента PostgreSQL локальными для установочного узла (true) или находятся на управляемом узле (false).
- automationgateway_ca_cert_local – указание того, являются ли файлы сертификатов, предоставленные Platform Gateway, локальными для установочного узла (true) или находятся на управляемом узле (false).
- automationgateway_cache_cert_local – указание того, являются ли файлы сертификатов клиента кэша локальными для установочного узла (true) или находятся на управляемом узле (false).

  Значение по умолчанию: значение параметра ca_cert_local.

- automationgateway_ssl_verify – указание того, следует ли проверять веб-сертификаты Platform Gateway при выполнении вызовов из Platform Gateway к себе во время установки.
- automationgateway_compress_archive – включение сжатия архива для Platform Gateway.
  - true – включено;
  - false – выключено.
- automationgateway_compress_database – включение сжатия базы данных для Platform Gateway.
  - true – включено;
  - false – выключено.
- automationgateway_regenerate_cache_certs – указание того, следует ли повторно создавать сертификаты клиента Platform Gateway для кэша платформы.
  - true – включено;
  - false – выключено (по умолчанию).
- automationgateway_disable_https_for_ui – отключение HTTPS при доступе к UI платформы.

  - `true` – выключено;
  - `false` – включено (по умолчанию).
- `automationgateway_pg_client_cert_auth` – включение аутентификации по сертификату клиента на базе данных PostgreSQL Platform Gateway.
  - `true` – включено;
  - `false` – выключено (по умолчанию).
- `automationgateway_pg_client_cert` – путь к файлу сертификата TLS клиента для подключения к СУБД PostgreSQL.

  Требуется, если включена аутентификация по сертификату клиента (`automationgateway_pg_client_cert_auth`).
- `automationgateway_pg_client_key` – путь к файлу ключа SSL/TLS клиента для подключения к СУБД PostgreSQL.

  Требуется, если включена аутентификация по сертификату клиента (`automationgateway_pg_client_cert_auth`).
- `automationgateway_secret_key` – значение секретного ключа, используемое Platform Gateway для подписи и шифрования данных.
- `automationgateway_envoy_http_port` – номер порта, на котором прокси Envoy прослушивает входящие HTTP-соединения.
- `automationgateway_envoy_https_port` – номер порта, на котором прокси Envoy прослушивает входящие HTTPS-соединения.

Настройка Automation Controller

Переменные узлов Automation Controller задают в соответствующих секциях, а учетные данные администратора и переменные подключения к СУБД – в секции `[all:vars]`.

Настройка узлов

Для узлов Automation Controller доступны переменные, задающие тип узла (`node_type`) и связанную группу узлов (`peers`).

- `node_type` – тип узла:
  - `control` – управляющий;
  - `execution` – исполняющий;
  - `hop` – промежуточный;
  - `hybrid` – гибридный.

  Для узлов из секции `[automationcontroller]` можно использовать только типы `control` и `hybrid` (по умолчанию), однако в рекомендуемых топологиях представлен только тип `control`.

  Для узлов из секции `[execution_nodes]` можно использовать только типы `hop` и `execution` (по умолчанию).
- `peers` – название связанной группы узлов.

  С помощью `peers` настраивается связь между плоскостью управления и плоскостью исполнения.

  Пример использования переменной `peers` см. в описании развертывания *кластера Automation Controller с удаленным исполнением*.

Учетные данные системного администратора

Учетные данные системного администратора Automation Controller задают следующими переменными:

- `admin_email` – адрес электронной почты;
- `admin_password` – пароль;
- `admin_username` – название учетной записи.

  Значение по умолчанию – «admin». В текущей версии изменять это значение нельзя.

Сеть

Параметры сети Automation Controller задают следующими глобальными переменными:

- `mesh_ca_certfile` – путь к файлу сертификата сети Mesh.
- `mesh_ca_keyfile` – путь к файлу ключа сертификата сети Mesh.
- `receptor_listener_port` – номер порта, используемый рецептором.

  Значение по умолчанию: 27199.
- `web_server_ssl_cert` – путь к файлу сертификата веб-сервера.
- `web_server_ssl_key` – путь к файлу ключа сертификата веб-сервера.

СУБД

Параметры подключения Automation Controller к СУБД задают следующими глобальными переменными:

- `pg_database` – название базы данных.

  Значение по умолчанию: awx.
- `pg_host` – IP-адрес или FQDN узла с СУБД.

  Если указана пустая строка (по умолчанию), используется локальное подключение к СУБД, развернутой на том же узле, что и Automation Controller.
- `pg_password` – пароль учетной записи СУБД.
- `pg_port` – порт СУБД.

  Значение по умолчанию: 5432.
- `pg_sslmode` – режим использования SSL при подключении к СУБД:
  - `prefer` – подключение к СУБД будет защищено с помощью SSL, если использование шифрования поддерживается настройками сервера (по умолчанию);
  - `verify-full` – защита подключения с помощью SSL необходима, производится строгая верификация сервера.
- `pg_username` – название учетной записи учетной записи СУБД.

  Значение по умолчанию: awx.
- `postgres_ssl_cert` – путь к файлу сертификата, который используется для защиты подключения к серверу СУБД PostgreSQL, развернутому средствами платформы.
- `postgres_ssl_key` – путь к файлу ключа сертификата, который используется для защиты подключения к серверу СУБД PostgreSQL, развернутому средствами платформы.
- `postgres_use_ssl` – использование SSL для защиты подключения к СУБД:
  - `true` – включено;
  - `false` – выключено (по умолчанию).

Настройка Private Automation Hub

Переменные Private Automation Hub задают в секциях `[automationhub]`, `[automationhub:vars]` и `[all:vars]`.

Настройка узлов

В секции `[automationhub]` задают переменные подключения к узлам Private Automation Hub, например:

```
[automationhub]
hub1.example.com   ansible_host=192.168.56.102
hub2.example.com   ansible_host=192.168.56.103
```

Учетные данные администратора

Учетные данные администратора Private Automation Hub задают следующими глобальными переменными:

- `automationhub_admin_password` – пароль.
- `automationhub_admin_user` – название учетной записи.

  Значение по умолчанию: `admin`.

Сеть

Сетевую настройку Private Automation Hub определяют с помощью следующих переменных:

- `automationhub_main_url` – Private Automation Hub URL.

  Эта переменная определяет внешний URL для доступа к Private Automation Hub при развертывании его на двух или более узлах. Это гарантирует, что все компоненты используют единообразный URL при обращении к реестру контейнеров. Без явного указания этого параметра система использует первый узел из группы `automationhub`.
- `automationhub_ssl_cert` – путь к файлу сертификата TLS, используемого веб-сервером.
- `automationhub_ssl_key` – путь к файлу ключа сертификата TLS, используемого веб-сервером.

СУБД

Параметры подключения Private Automation Hub к СУБД задают следующими глобальными переменными:

- `automationhub_pg_database` – название базы данных.

  Значение по умолчанию: `automationhub`.
- `automationhub_pg_host` – IP-адрес или FQDN узла с СУБД.
- `automationhub_pg_password` – пароль учетной записи СУБД.
- `automationhub_pg_port` – порт СУБД.

  Значение по умолчанию: 5432.
- `automationhub_pg_username` – название учетной записи учетной записи СУБД.

  Значение по умолчанию: `automationhub`.

Настройка хранилища S3

Параметры подключения к хранилищу S3 задают следующими глобальными переменными:

- `s3_access_key` – ключ доступа к объектному хранилищу.

- s3_addressing_style – стиль адресации объектов в хранилище:
  - auto – если возможно, используется стиль virtual, в противном случае – path;
  - path – название хранилища включено в путь URI (по умолчанию);
  - virtual – название хранилища включено в FQDN узла.
- s3_bucket_name – название хранилища.
- s3_default_acl – список контроля доступа (access control list, ACL), используемый при работе с хранилищем по умолчанию.

  Список возможных значений зависит от типа используемого хранилища. Рекомендуется не задавать значение этого параметра в инвентаре, а выполнить необходимые настройки на стороне хранилища.

  Значение по умолчанию: omit.
- s3_endpoint_url – точка доступа к хранилищу.
- s3_region_name – название региона, в котором размещено хранилище.

  Список возможных значений зависит от типа используемого хранилища.
- s3_secret_key – секретный ключ для доступа к хранилищу.
- s3_signature_version – версия подписи, используемая для работы с хранилищем:
  - s3v4 (по умолчанию);
  - s3.

  Примечание

  Это значение считается устаревшим и не рекомендуется к использованию.
- s3_use_sigv4 – подписывание запросов к хранилищу с помощью AWS Signature V4[30]:
  - true – включено (по умолчанию);
  - false – выключено.

Настройка резервного копирования

Глобальная настройка automationhub_backup_collections управляет включением загруженных коллекций в резервную копию, создаваемую с помощью утилиты aa-setup:

- true – коллекции из Private Automation Hub включаются в резервную копию (по умолчанию);
- false – коллекции из Private Automation Hub не включаются в резервную копию.

Когда содержимое Private Automation Hub размещается в хранилище S3, по умолчанию назначается false. Для этого случая оно является рекомендуемым.

Настройка контроллера Event-Driven Automation

Переменные контроллера Event-Driven Automation задают в секциях [automationedacontroller] и [all:vars].

Настройка узла

В секции [automationedacontroller] задают переменные подключения к узлу контроллера Event-Driven Automation, например:

```
[automationedacontroller]
edac.example.com
```

[30] https://docs.aws.amazon.com/AmazonS3/latest/API/sig-v4-authenticating-requests.html

Учетные данные администратора

Учетные данные администратора контроллера Event-Driven Automation задают следующими глобальными переменными:

- `automationedacontroller_admin_password` – пароль;
- `automationedacontroller_admin_username` – название учетной записи.

  Значение по умолчанию – «admin». В текущей версии изменять это значение нельзя.

Сеть

- `automationedacontroller_ssl_cert` – путь к файлу сертификата TLS, используемого веб-сервером.
- `automationedacontroller_ssl_key` – путь к файлу ключа сертификата TLS, используемого веб-сервером.

СУБД

Параметры подключения контроллера Event-Driven Automation к СУБД задают следующими глобальными переменными:

- `automationedacontroller_pg_database` – название базы данных.

  Значение по умолчанию: `automationedacontroller`.
- `automationedacontroller_pg_host` – IP-адрес или FQDN узла с СУБД.
- `automationedacontroller_pg_password` – пароль учетной записи СУБД.
- `automationedacontroller_pg_port` – порт СУБД.

  Значение по умолчанию: 5432.
- `automationedacontroller_pg_username` – название учетной записи СУБД.

  Значение по умолчанию: `automationedacontroller`.

Настройка развертывания без доступа к интернету

Параметры развертывания платформы без доступа к интернету задают следующими глобальными переменными:

- `bundle_dir` – путь к каталогу с файлами установщика платформы.

  Значение по умолчанию: `/opt/rbta/aa/astra-automation-setup/`.
- `bundle_repo_folder` – путь к каталогу, в который будут скопированы DEB-пакеты, необходимые для развертывания платформы. Указанный каталог будет использоваться при настройке репозиториев APT.

  Значение по умолчанию: `/var/lib/astra-automation-setup/bundle/`.
- `bundle_install` – развертывание платформы в окружении без доступа к интернету:
  - `true` – включено;
  - `false` – выключено, используются интернет-репозитории ПАО Группа Астра (по умолчанию).

5.3.2 В кластере Kubernetes

При использовании Kubernetes фаза подготовки включает подготовку кластера Kubernetes для развертывания центральной части Astra Automation и других средств. Она состоит из нескольких стадий:

Вы прошли стадию выбора модели развертывания, выбрав развертывание **в кластере Kubernetes**. Остальные стадии впереди. Для перехода сразу на некоторую стадию выберите соответствующий блок диаграммы.

Планирование развертывания центральных компонентов Astra Automation как комплексного приложения в среде Kubernetes состоит из следующих шагов:

1. *Подготовка рабочей станции администратора*.
2. *Выбор топологии и подготовка ресурсов*.
3. (При необходимости) *Дополнительная подготовка для сегмента, изолированного от интернет*.
4. *Подготовка манифестов для установки секретов и развертывания приложения*.

При развертывании платформы в закрытом контуре, изолированном от интернета, необходимо учитывать *особенности этого варианта*.

Установочный узел

На этой стадии необходимо подготовить рабочую станцию администратора, называемую далее *установочным узлом*.

Для развертывания платформы через кластер Kubernetes потребуются дополнительные программные ресурсы на установочном узле. Установите следующее программное обеспечение:

- утилита kubectl[31];
- утилита Ansible Navigator[32].

Если в качестве установочного узла используется Astra Linux Special Edition, то подробные инструкции по установке Ansible Navigator или всего CDK (Content Development Kit) как при наличии доступа к интернету, так и без доступа приведены в *руководстве для разработчика*.

Топология

На этой стадии необходимо выбрать топологию платформы и подготовить соответствующие ресурсы.

В обзоре архитектуры вы познакомились с двумя протестированными вариантами топологии, их особенностями, преимуществами и недостатками. Это позволило вам определить вариант топологии, наиболее подходящий для ваших бизнес-планов. Следующий шаг состоит в планировании и подготовке ресурсов для выбранной топологии.

Ресурсы Kubernetes

Для развертывания центральных компонентов Astra Automation необходим кластер Kubernetes. Кластер может быть в собственном центре обработки данных или в облаке, см. *Примеры подготовки кластера Kubernetes*.

Необходимое количество ресурсов как в кластере Kubernetes, так и за его пределами существенно зависит от выбранной топологии.

[31] https://kubernetes.io/docs/tasks/tools/
[32] https://docs.ansible.com/projects/navigator/installation/

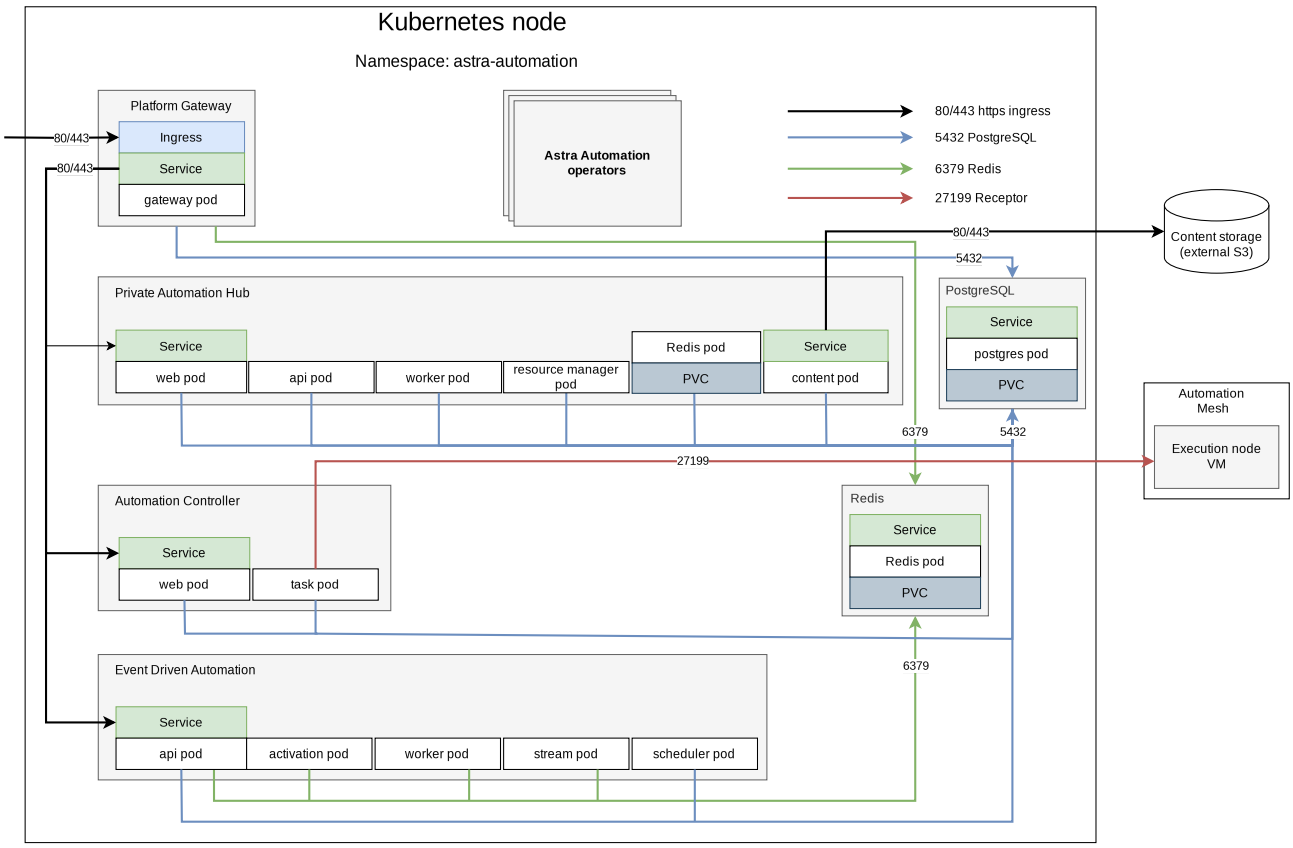
Ресурс	Базовая	Уровень предприятия
PG gateway pod	1	3+
AC web pod	1	3+
AC task pod	1	3+
Automation Mesh ingress	1	3+
PAH web pod	1	3+
PAH api pod	1	3+
PAH worker pod	1	3+
PAH redis pod	1	3+
PAH content pod	1	3+
PAH content storage	External	External
EDA api pod	1	3+
EDA activation pod	1	3+
EDA worker pod	1	3+
EDA stream pod	1	3+
EDA scheduler pod	1	3+
Redis pod	1	6+
PostgreSQL	Pod + PV	External
Execution plane	External execution node	External execution nodes + hop

В таблице использованы следующие сокращения:

- PG – Platform Gateway;
- AC – Automation Controller;
- PAH – Private Automation Hub;
- EDA – контроллер Event-Driven Automation;
- PV – Kubernetes Persistent Volume.

Базовая топология

В базовой топологии для развертывания Astra Automation используется минимальное количество ресурсов.



Для обслуживания различных сегментов инфраструктуры Automation Controller подключается к сети Mesh, в которой установлены исполняющие узлы.

Требования к рабочему узлу Kubernetes

Минимальные требования к рабочему узлу, на котором развертываются все центральные компоненты Astra Automation:

- количество ядер CPU: 4;
- объема RAM: 16 ГБ;
- дисковое пространство: 100 ГБ;
- скорость операций ввода-вывода, IOPS: 3000.

Требования к сетевому хранилищу

Хранить контент приватного реестра рекомендуется в хранилище S3, размеры которого зависят от объема контента. Для хранения контента, полученного из облачного реестра Automation Hub, необходимо зарезервировать 3 ГБ пространства. Для загрузки дополнительного контента, например из Ansible Galaxy, может понадобиться в несколько раз больший объем, но обычно не более 40 ГБ.

Требования к исполняющему узлу Astra Automation

Для ВМ, используемой для установки исполняющего узла (exec node), предъявляются следующие минимальные требования.

Параметр	Значение
Количество ядер CPU	8
Объем RAM	16 GB
Дисковое пространство	60 GB
Тип дискового накопителя	SSD
IOPS	3000

Настройка сетевых фильтров

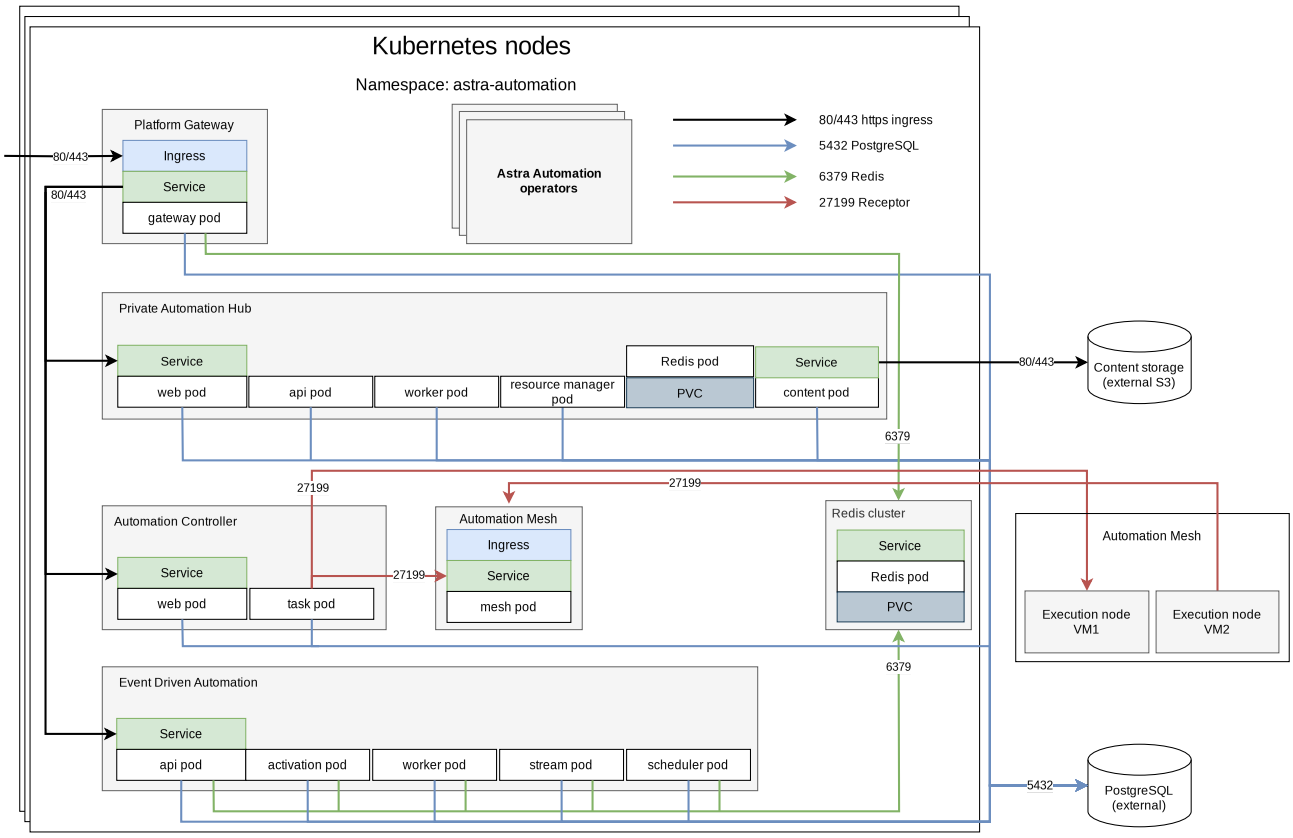
Для обеспечения взаимодействия компонентов платформы необходимо обеспечить установление требуемых сетевых связей по соответствующим портам TCP как представлено в таблице.

Источник	Назначение	Служба	Порт TCP
Admin workstation	Kubernetes cluster	HTTPS	6443
User workstation	Platform Gateway	HTTP/HTTPS	80/443
Content pod	External S3	HTTPS	443
Automation Controller	Execution node	Receptor	27199
Execution node	Platform Gateway	HTTP/HTTPS	80/443
Execution node	Управляемый сегмент	SSH	22

Топология уровня предприятия

Эта топология по сравнению с предыдущей преследует две основные цели, для достижения которых применяются соответствующие изменения:

- Для обеспечения высокой доступности все ресурсы Kubernetes имеют несколько реплик (обычно 3) на разных рабочих узлах кластера. Для еще большего повышения уровня доступности используют кластер Kubernetes, распределенный по зонам доступности.
- Для управления сложными распределенными инфраструктурами задействована сеть Mesh, в которой предусматривают распределение нагрузки по множеству исполняющих узлов (exec nodes) с применением переходных узлов (hop nodes) для маршрутизации трафика. Для критически важных сегментов используют несколько исполняющих узлов.



Требования к рабочему узлу Kubernetes

Минимальные требования к ресурсам для развертывания всех центральных компонентов Astra Automation с учетом только одной реплики для каждого Kubernetes pod, остаются теми же, что и для базовой топологии:

- количество ядер CPU: 4;
- объема RAM: 16 ГБ;
- дисковое пространство: 100 ГБ;
- скорость операций ввода-вывода, IOPS: 3000.

Примечание

Для оценки полного объема требуемых ресурсов надо учитывать количество реплик.

Требования к сетевому хранилищу

Хранить контент приватного реестра рекомендуется в хранилище S3, размеры которого зависят от объема контента. Для хранения контента, полученного из облачного реестра Automation Hub, необходимо зарезервировать 3 ГБ пространства. С учетом дальнейшего расширения рекомендуемый объем составляет 40 ГБ.

Требования к исполняющему и переходному узлам Astra Automation

Для ВМ, используемой для установки переходного или исполняющего узла, предъявляются следующие минимальные требования.

- количество ядер CPU: 8;
- объема RAM: 16 ГБ;
- дисковое пространство: 60 ГБ;
- скорость операций ввода-вывода, IOPS: 3000.

Требования к внешней СУБД PostgreSQL

Astra Automation работает только с PostgreSQL версии 15 с установленным расширением hstore. Критически важными являются следующие характеристики:

- Для каждого из следующих компонентов Astra Automation необходимо наличие базы данных и учетной записи для управления ею:
  - Platform Gateway;
  - Automation Controller;
  - Private Automation Hub;
  - контроллер Event-Driven Automation.

  Пример создания с помощью команд SQL:

```
-- Creating users for Astra Automation
CREATE USER automationgateway WITH PASSWORD 'gateDBpaS$12345';
CREATE USER automationcontroller WITH PASSWORD 'ctrlDBpaS$123456';
CREATE USER automationhub WITH PASSWORD 'hUbPa55I2345b';
CREATE USER automationedacontroller WITH PASSWORD 'edapassword';

-- Creating databases
CREATE DATABASE automationgateway OWNER automationgateway;
CREATE DATABASE automationcontroller OWNER automationcontroller;
CREATE DATABASE automationhub OWNER automationhub;
CREATE DATABASE automationedacontroller OWNER automationedacontroller;
```

- Поддержка ICU (International Components for Unicode). Проверка и инициализация:

```
-- Validating ICU в PostgreSQL
SELECT * FROM pg_available_extensions WHERE name = 'icu';

-- Initializing DB with ICU
initdb -D pgdata --locale-provider=icu --icu-locale=en
```

- Работающий сервис Autovacuum для исключения бесконтрольного увеличения размера таблиц.

Настройка сетевых фильтров

Для обеспечения взаимодействия компонентов платформы необходимо обеспечить установление требуемых сетевых связей по соответствующим портам TCP как представлено в таблице.

Источник	Назначение	Служба	Порт TCP
Admin workstation	Kubernetes cluster	HTTPS	6443
User workstation	Platform Gateway	HTTP/HTTPS	80/443
Content pod	External S3	HTTPS	443
All pods	External PostgreSQL	TCP	5432
Automation Controller	Execution nodes	Receptor	27199
Automation Controller	Hop nodes	Receptor	27199
Execution nodes	Platform Gateway	HTTP/HTTPS	80/443
Execution nodes	Управляемый сегмент	SSH	22

Манифесты

На этом шаге следует подготовить манифесты для установки всех необходимых объектов и развертывания платформы в кластере Kubernetes.

Развертывание платформы с учетом выбранной топологии реализуется путем выполнения манифестов, создающих ресурсы Kubernetes, необходимые для приложения.

Терминология

В Kubernetes используют специфичную терминологию[33], которую следует учитывать при ознакомлении с последующим описанием:

- Ресурс (resource) – это точка доступа (endpoint) в API, например, `Pod` или `Deployment`. Его название может быть представлено в API в единственном (для управления одним объектом) и множественном (для управления списком) числе. С помощью ресурса можно создавать множество объектов (objects), где каждый объект является описанием состояния некой сущности, которой может быть компонент, например Pod, или какой-либо контроллер Kubernetes, например Service. Название ресурса используют при управлении объектами через параметр `kind` манифестов, в командах `kubectl` и прямых запросах к API.
- Собственный (специальный) ресурс (CR, Custom Resource) – расширение API путем создания еще одной точки доступа. Например после создания CR, названного AstraAutomation, в Kubernetes API появляется точка доступа с этим названием.
- Определение собственного ресурса (CRD, Custom Resource Definition) – это особый ресурс, то есть точка доступа в Kubernetes API, названная `CustomResourceDefinition`, которую используют для создания CR. Для создания CR применяют манифест, в котором описывают API требуемого ресурса в стандарте OpenAPI, задавая на верхнем уровне манифеста `kind: CustomResourceDefinition`.

[33] https://kubernetes.io/docs/concepts/extend-kubernetes/api-extension/custom-resources/#custom-resources

Ресурсы платформы

Для развертывания платформы необходимы ресурсы Kubernetes, обеспечивающие функционирование и взаимодействие ее компонентов с внешними системами:

- собственные ресурсы приложения (CR), описанные в манифестах операторов с помощью нескольких CRD;
- ресурсы типа Secret, используемые компонентами Astra Automation для обеспечения взаимодействия с внешними компонентами:
  - TLS secret – приватный ключ и сертификат для HTTPS, используемый контроллером ingress для защиты данных;
  - реквизиты для доступа к хранилищу S3;
  - реквизиты для доступа к базам данных во внешней СУБД PostgreSQL (топология уровня предприятия);
  - encryption secrets – ключи шифрования данных в СУБД (топология уровня предприятия);
  - image pull secret – реквизиты для доступа к репозиторию образов контейнеров;
  - EE image pull secret – реквизиты для доступа к репозиторию образов для среды исполнения;
  - пароль пользователя admin для доступа к Platform Gateway.

Подготовка секретов

Создание секретов – один из важнейших шагов при подготовке к развертыванию платформы, который должен соответствовать регламенту по обеспечению безопасности в организации.

Секрет для TLS

Для обеспечения доступа к Platform Gateway по протоколу HTTPS необходимо предоставить для контроллера Ingress секрет, содержащий приватный ключ и сертификат TLS. Подготовка этого ресурса включает следующие шаги:

1. Закажите ключ и сертификат от удостоверяющего центра или создайте самоподписанный сертификат и ключ к нему.
2. Закодируйте каждый из них в формате Base64.
3. Создайте манифест секрета в виде отдельного файла со следующим содержимым:

```
---
apiVersion: v1
kind: Secret
metadata:
  name: aa-tls-secret
  namespace: astra-automation
data:
  tls.crt: <base64-encoded-cert-chain>
  tls.key: <base64-encoded-key>
type: kubernetes.io/tls
```

   Здесь:

   - <base64-encoded-cert-chain> – сертификат, закодированный в формате Base64;
   - <base64-encoded-key> – приватный ключ, закодированный в формате Base64.

Секрет для доступа к хранилищу S3

Подготовка хранилища S3 и манифеста секрета Kubernetes с реквизитами доступа состоит из следующих шагов:

1. Создайте собственное хранилище или закажите S3 bucket в публичном облаке.
2. Создайте манифест для секрета в виде отдельного файла со следующей структурой (для примера приведены реквизиты для Yandex Cloud):

```
---
apiVersion: v1
kind: Secret
metadata:
  name: pah-s3-credentials-secret
  namespace: astra-automation
stringData:
  s3-access-key-id: "YC********************" # <-- Access key ID
  s3-secret-access-key: "YC****************" # <-- Secret key
  s3-bucket-name: "aa**********************" # <-- Bucket name
  s3-region: ru-central1
  s3-endpoint: "https://storage.yandexcloud.net"
```

В файле манифеста подставьте собственные значения параметров:

- s3-access-key-id – идентификатор ключа доступа;
- s3-secret-access-key – ключ доступа;
- s3-bucket-name – название, которое вы дали для хранилища S3 bucket.

Секреты для доступа к базам данных

При использовании внешней СУБД необходимо создать секреты Kubernetes для всех требуемых баз данных. Создайте манифест секретов с помощью следующих шагов:

1. Убедитесь, что вы *создали базы данных* для всех компонентов Astra Automation и сохранили реквизиты доступа к ним.
2. Создайте манифест секретов в виде отдельного файла со следующим содержимым:

```
---
apiVersion: v1
kind: Secret
metadata:
  name: ac-external-pg-secret
  namespace: astra-automation
stringData:
  host: "X.X.X.X" # <-- IP address or domain name of external PostgreSQL
  port: "5432" # <-- TCP port of external PostgreSQL
  database: "awx" # <-- DB name
  username: "awx" # <-- DB user name with privileges on creation and migration
  password: "awx" # <-- DB user password
  sslmode: "prefer" # <-- SSL mode (disable, require, etc.)
  type: "unmanaged" # <-- Using external DBMS
type: Opaque
---
apiVersion: v1
kind: Secret
metadata:
  name: pah-external-pg-secret
  namespace: astra-automation
stringData:
  host: "X.X.X.X"
  port: "5432"
  database: "automationhub"
  username: "automationhub"
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  password: "automationhub"
  sslmode: "prefer"
  type: "unmanaged"
type: Opaque
---
apiVersion: v1
kind: Secret
metadata:
  name: eda-external-pg-secret
  namespace: astra-automation
stringData:
  host: "X.X.X.X"
  port: "5432"
  database: "automationedacontroller"
  username: "automationedacontroller"
  password: "automationedacontroller"
  sslmode: "prefer"
  type: "unmanaged"
type: Opaque
---
apiVersion: v1
kind: Secret
metadata:
  name: aa-external-pg-secret
  namespace: astra-automation
stringData:
  host: "X.X.X.X"
  port: "5432"
  database: "automationgateway"
  username: "automationgateway"
  password: "automationgateway"
  sslmode: "prefer"
  type: "unmanaged"
type: Opaque
```

Особого внимания требуют параметры `sslmode` и `type: Opaque`.

Параметр `sslmode` определяет режим использования TLS при подключении к внешней базе данных PostgreSQL. Этот параметр переносится в настройки клиента и задает политику работы с шифрованными соединениями:

- `disable` – подключение происходит без использования TLS, данные передаются в открытом виде.
- `prefer` – клиент пытается установить защищенное соединение, если сервер его поддерживает. Если сервер не поддерживает это, соединение будет нешифрованным. Именно такой режим используется в приведенном манифесте.
- `require` – соединение с сервером возможно только по в защищенном режиме. При отсутствии поддержки шифрования соединение не устанавливается.
- `verify-ca` – дополнительная проверка сертификата TLS, нацеленная на то, чтобы клиент удостоверился, что цепочка сертификатов ведет с корневому сертификату CA;
- `verify-full` – дополнительный режим для максимальной безопасности, аналогичный предыдущему, но с проверкой того, что доменное имя или IP-адрес сервера совпадает с тем, что записано в сертификате TLS.

Поле `type: Opaque` используется в описании Kubernetes secret для указания того, что данные секрета представлены как произвольные строки, закодированные в формате Base64.

Секреты для шифрования

Поскольку в базах данных необходимо хранить некоторые данные в зашифрованном виде, для каждой базы требуется создать ключ шифрования с помощью секрета. Создайте манифест секретов с помощью следующих шагов:

1. Для каждой базы данных, кроме базы для Private Automation Hub (PAH) генерируйте ключ шифрования (всего три ключа) с помощью следующей команды и сохраните все ключи:

   ```
   openssl rand -base64 32
   ```

   Для базы данных PAH требуется более длинный ключ, генерируемы следующей командой:

   ```
   openssl rand -base64 32 | base64
   ```

2. Создайте манифест для секретов в виде отдельного файла со следующим примерным содержимым:

   ```
   ---
   apiVersion: v1
   kind: Secret
   metadata:
     name: ac-encryption-secret
     namespace: astra-automation
   data:
     secret_key: I4AD8TR/zsoBEBXNPcQ4jxCixveTepfHg1KvItBVKf4= # Base64 encoded
   type: Opaque
   ---
   apiVersion: v1
   kind: Secret
   metadata:
     name: eda-encryption-secret
     namespace: astra-automation
   data:
     secret_key: f4lj4+FpOWMRY6l2e5p5eE/lz8JdR3gTNqtWkcXvDq8= # Base64 encoded
   type: Opaque
   ---
   apiVersion: v1
   kind: Secret
   metadata:
     name: aa-encryption-secret
     namespace: astra-automation
   data:
     secret_key: 61a260kWBPmPX89TMPZGbLTsaLmnWBnTShX0S3NyFns= # Base64 encoded
   type: Opaque
   ---
   apiVersion: v1
   kind: Secret
   metadata:
     name: pah-encryption-secret
     namespace: astra-automation
   data:
     database_fields.symmetric.key:␣
   ↪clpsTGdYVG1PUmcxNmxNQmEwdENaQVVoVGtEbltekJaNUdxWS9ISVNjdz0K # Base64 encoded
   type: Opaque
   ```

Секрет для доступа к Platform Gateway

Для обеспечения доступа к Platform Gateway через веб-интерфейс или API с помощью учетной записи администратора необходимо подготовить пароль для нее путем создания соответствующего секрета Kubernetes. Создайте манифест секрета с помощью следующих шагов:

1. Подготовьте пароль, например aa-demo-enterprise-password и закодируйте его методом Base64:

```
echo -n 'aa-demo-enterprise-password' | base64
```

2. Создайте манифест секрета в виде отдельного файла со следующим примерным содержимым:

```
---
apiVersion: v1
kind: Secret
metadata:
  name: aa-demo-admin-password
  namespace: astra-automation
data:
  password: YWEtZGVtby1lbnRlcnByaXNlLXBhc3N3b3Jk # Base64 encoded
type: Opaque
```

Манифесты приложения

При создании собственных манифестов для развертывания Astra Automation как приложения примите во внимание представленные здесь типовые манифесты. Их следует проверить и внести необходимые изменения, учитывающие особенности вашей инфраструктуры.

Общими для манифестов являются следующие ключевые моменты:

- создание объекта aa-demo с помощью специального ресурса (CR) AstraAutomation, создаваемого с помощью операторов Kubernetes;
- использование пространства, созданного оператором, – metadata.namespace: astra-automation;
- перевод компонентов Astra Automation в активное состояние:
  - spec.controller.disabled: false
  - spec.hub.disabled: false
  - spec.eda.disabled: false

Базовая топология

Пример манифеста для базовой топологии:

```
---
apiVersion: aa.astra-automation.ru/v1alpha1
kind: AstraAutomation
metadata:
  name: aa-demo
  namespace: astra-automation
spec:
  controller:
    disabled: false
    name: ac-demo
    ee_pull_credentials_secret: "ac-ee-pull-secret"
    image_pull_secrets: ["aa-operators-pull-secret"]
  hub:
    disabled: false
    name: pah-demo
    storage_type: S3
    object_storage_s3_secret: pah-s3-credentials-secret
    image_pull_secrets: ["aa-operators-pull-secret"]
  eda:
    disabled: false
    name: eda-demo
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
    image_pull_secrets: ["aa-operators-pull-secret"]
  image_pull_secrets: ["aa-operators-pull-secret"]
  ingress_type: Ingress
  ingress_class_name: nginx
#  ingress_tls_secret: aa-tls-secret
#  public_base_url: https://aa-demo.example.com
#  hostname: aa-demo.example.com
```

Топология уровня предприятия

Пример манифеста приложения для топологии уровня предприятия:

```
---
apiVersion: aa.astra-automation.ru/v1alpha1
kind: AstraAutomation
metadata:
  name: aa-demo
  namespace: astra-automation
spec:
  controller:
    disabled: false
    name: ac-demo
    database_secret: ac-external-pg-secret
    secret_key_secret: ac-encryption-secret
    ee_pull_credentials_secret: "ac-ee-pull-secret"
    image_pull_secrets: ["aa-operators-pull-secret"]
    replicas: 3
  hub:
    disabled: false
    name: pah-demo
    storage_type: S3
    object_storage_s3_secret: pah-s3-credentials-secret
    database_secret: pah-external-pg-secret
    db_fields_encryption_secret: pah-encryption-secret
    image_pull_secrets: ["aa-operators-pull-secret"]
    api:
      replicas: 3
    web:
      replicas: 3
    content:
      replicas: 3
    worker:
      replicas: 3
    resource_manager:
      replicas: 3
  eda:
    disabled: false
    name: eda-demo
    database_secret: eda-external-pg-secret
    db_fields_encryption_secret: eda-encryption-secret
    image_pull_secrets: ["aa-operators-pull-secret"]
    api:
      replicas: 3
    event_stream:
      replicas: 3
    ui:
      replicas: 3
    default_worker:
      replicas: 3
    activation_worker:
      replicas: 3
    worker:
      replicas: 3
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
    scheduler:
      replicas: 3
  database:
    database_secret: aa-external-pg-secret
  api:
    replicas: 3
  redis_mode: cluster
  image_pull_secrets: ["aa-operators-pull-secret"]
  ingress_type: Ingress
  ingress_class_name: nginx
  ingress_tls_secret: aa-tls-secret
  db_fields_encryption_secret: aa-encryption-secret
  public_base_url: https://aa.demo.example.com
  hostname: aa.demo.example.com
  admin_password_secret: aa-demo-admin-password
```

По сравнению с предыдущим вариантом включает следующие настройки:

- репликация подов;
- секреты для установления соединений с базами данных на внешней СУБД;
- настройка балансировщика Ingress;
- наличие доменного имени для балансировщика Ingress.

Для повышения масштабируемости рекомендуется установить промежуточный узел (hop node) в Kubernetes, который принимает запросы от вновь добавляемых исполняющих узлов (execution nodes) на установление соединения с Automation Controller согласно *топологии*. Для этого подготовьте манифест:

```
---
apiVersion: ac.astra-automation.ru/v1alpha1
kind: AutomationControllerMeshIngress
metadata:
  name: ac-demo-mesh
  namespace: astra-automation
spec:
  deployment_name: ac-demo
  external_hostname: aa.demo.mesh.example.ru
  ingress_class_name: nginx
  ingress_controller: nginx
  ingress_type: Ingress
```

Здесь:

- `external_hostname` – доменное имя для создаваемой точки доступа, к которой будут подключаться внешние исполняющие узлы.

Без доступа в интернет

Развертывание платформы в кластере Kubernetes требует дополнительных подготовительных шагов.

Для развертывания Astra Automation в кластере Kubernetes без доступа в интернет необходимо подготовить все требуемые контейнерные образы локально на каждом рабочем узле (worker node). По умолчанию Kubernetes использует среду исполнения контейнеров *containerd*. При отсутствии доступа к внешнему реестру контейнеров потребуются дополнительные шаги по загрузке этих образов во все рабочие узлы.

Доступ к узлам Kubernetes

Для выполнения последующих действий оператор установочного узла должен иметь доступ по SSH ко всем узлам (master, workers) кластера Kubernetes с административными привилегиями (через утилиту sudo). Настройте доступ как описано в руководстве *Настройка целевых узлов*.

Подготовка образов контейнеров

Пакет для развертывания Astra Automation в сегменте сети без доступа к интернет поставляется в виде архивированного файла `aa-kubernetes-bundle-2-0-preview-offline.tar.gz`, доступного в Личном кабинете[34] при наличии действующей лицензии на Astra Automation.

Выполните следующие подготовительные действия:

1. Загрузите архивированный файл на установочный узел.
2. Подготовьте временный каталог для размещения пакета:

```
mkdir temp_bundle_contents
```

3. Распакуйте архив в этот каталог:

```
tar -xzvf aa-kubernetes-bundle-2-0-preview-offline.tar.gz -C temp_bundle_contents
```

```
cd temp_bundle_contents
```

Структура каталога выглядит следующим образом:

```
temp_bundle_contents/
├── examples/
│   ├── kubernetes/
│   │   ├── base/
│   │   └── enterprise/
│   └── minikube/
│       └── minimal/
├── images/
│   ├── aa_2.0-preview.tar
│   ├── aa-control-ee_latest.tar
│   ├── aa-operator_2.0-preview.tar
│   ├── aa-proxy_2.0-preview.tar
│   ├── ac_2.0-preview.tar
│   ├── eda-ui_2.0-preview.tar
│   ├── kube-rbac-proxy_2.0-preview.tar
│   ├── nginx_2.0-preview.tar
│   ├── pah-operator_2.0-preview.tar
│   ├── postgresql_2.0-preview.tar
│   └── redis_2.0-preview.tar
└── tools/
    ├── inventory.yml
    └── load-containerd-images.yml
```

Подготовленный каталог потребуется не только на этом шаге, но и при дальнейшей подготовке и непосредственном развертывании платформы. Он содержит следующие ресурсы:

- `examples/` – манифесты различных секретов для установки их в кластере Kubernetes в зависимости от выбранной топологии (базовая или уровня предприятия);
- `images/` – образы контейнеров, которые необходимо установить в кластере Kubernetes.
- `tools/` – сценарий Ansible и описание инвентаря, включающего все рабочие узлы Kubernetes, на которых необходимо разместить образы.

Описание инвентаря

В файле инвентаря `tools/inventory.yml` имеется структура описания, в которой необходимо заменить параметры рабочих узлов на параметры узлов вашего кластера, а также задать другие параметры Ansible:

[34] https://lk.astra.ru/

```
---
all:
  children:
    k8s_nodes:
      hosts:
        node1.example.com:
          ansible_host: 10.177.1.1
        node2.example.com:
          ansible_host: 10.177.1.2
        node3.example.com:
          ansible_host: 10.177.1.3
      vars:
        ansible_user: astra
        ansible_become: true
        ansible_ssh_private_key_file: ~/.ssh/astra
        images_dir: /tmp/images
```

В этом файле находятся следующие параметры, подлежащие настройке:

- адреса узлов (доменные имена и IP-адреса);
- `ansible_user` – учетная запись, используемая для подключения по SSH и имеющая привилегии на использование утилиты sudo для повышения привилегий до привилегий учетной записи root;
- `ansible_become` – требует для выполнения модулей Ansible на удаленном узле использовать утилиту sudo для повышения привилегий до привилегий учетной записи root;
- `ansible_ssh_private_key_file` – путь к приватному ключу учетной записи, используемой для подключения к управляемым узлам по SSH;
- `images_dir` – путь к каталогу, в котором хранятся файлы с образами контейнеров.

Размещение образов

Используйте подготовленный сценарий playbook для размещения образов на рабочих узлах Kubernetes:

```
ansible-navigator run tools/load-containerd-images.yml -i tools/inventory.yml
```

Сценарий выполняет следующие действия:

1. Устанавливает сервис `rsync` на узлах, если его нет.
2. Копирует все файлы с расширением `.tar` из каталога `images/` во временный каталог на указанные узлы кластера.
3. Импортирует каждый образ этих файлов для службы `containerd`.
4. После завершения выводит список образов, загруженных для `containerd`.

После успешного выполнения этого сценария образы будут доступны локально на всех указанных узлах, и вы сможете приступать к развертыванию операторов и платформы.

Примеры подготовки кластера Kubernetes

Если у вас отсутствует кластер Kubernetes, приведенные далее примеры помогут в его подготовке.

Подготовка кластера в Yandex Cloud Managed Service for Kubernetes

Yandex Cloud предоставляет услуги по созданию кластеров Kubernetes в облаке[35], которые обеспечены всеми необходимыми сервисами, включая управление узлами master и группами рабочих узлов, автоматическое масштабирование, автоматические обновления, резервное копирование. Также предоставлены услуги по сбору метрик, служба DNS, постоянные тома.

[35] https://yandex.cloud/en/services/managed-kubernetes

Предварительная подготовка

Необходимо предварительно подготовить следующие ресурсы:

- Создание кластера следует привязать к каталогу, на который у вас есть привилегии по созданию и управлению ресурсами.
- Поскольку запросы будут направляться через Yandex Cloud API, необходима сервисная учетная запись (SA, Service Account), которой назначены роли, достаточные для создания ресурсов в каталоге и управления ими. Для упрощения задачи достаточно назначить роль `editor`. Однако можно использовать более гранулярный подход согласно документации Yandex Cloud.
- Если по каким-либо причинам необходимо использовать сеть, привязанную к другому каталогу, необходимо назначить роли для SA на этот каталог:
  - `vpc.privateAdmin`;
  - `vpc.user`;
  - `vpc.bridgeAdmin` (если надо создавать публичные IP-адреса);
  - `vpc.publicAdmin` (если надо создавать публичные IP-адреса).
- Для хранения контента приватного реестра потребуется объектное хранилище в виде S3 bucket, объемом минимум 3 ГБ. Его можно также создать в Yandex Cloud. Назначьте роль `editor` для SA на S3 bucket.
- Для управления кластером со своей локальной рабочей станции используйте утилиту командного интерфейса Yandex Cloud CLI[36] (Command Line Interface).

Создание кластера

Создание кластера происходит в два этапа:

1. В группе ресурсов **Yandex Cloud Managed Service for Kubernetes** создайте кластер:
   - Назначьте созданную ранее учетную запись SA для ресурсов и узлов кластера.
   - Выберите автоматическое или статическое назначение публичного IP-адреса, если необходим доступ через интернет.
   - Выберите тип мастера **Basic** для базовой топологии или **High available** для топологии уровня предприятия.
   - Выберите сеть, обеспечивающую требуемые коммуникации, например сеть, связанную с вашим управляемым сегментом по VPN.
   - Выберите **Enable tunnel mode**.
2. Добавьте группу рабочих узлов для кластера:
   - Задайте необходимые ресурсы согласно рекомендациям:
     - для базовой топологии достаточно одного рабочего узла Kubernetes;
     - для топологии уровня предприятия необходимо не менее трех узлов с автоматическим масштабированием.
   - Выберите тип диска SSD.
   - Если необходимо назначить публичные IP-адреса, выберите автоматическое или статическое назначение публичного IP-адреса.

Управление кластером Kubernetes

Убедитесь, что вы можете управлять кластером со своей рабочей станции. Следуйте инструкциям в документации провайдера для настройки. Например, для Yandex Cloud

[36] https://yandex.cloud/ru/docs/cli/operations/install-cli

Managed Service for Kubernetes, настройте это через Yandex Cloud CLI (Command Line Interface) как описано в инструкции[37]:

1. Убедитесь, что у вас установлена утилита `kubectl` и установите соединение с вашим кластером, следуя указаниям в описании свойств кластера. Например:

```
yc managed-kubernetes cluster get-credentials --id cat8...i7d --external
```

2. Проверьте доступность кластера:

```
kubectl get pods -A
```

Команда выводит список служебных подов.

Общие настройки

Для повышения производительности измените класс хранилища по умолчанию, так чтобы использовать диски SSD:

1. Проверьте список классов:

```
kubectl get storageclass
```

2. Если настроено на использование по умолчанию `yc-network-hdd`, измените это следующими командами:

```
kubectl patch storageclass yc-network-hdd \
-p '{"metadata": {"annotations":{"storageclass.kubernetes.io/is-default-class":
↪"false"}}}'
```

```
kubectl patch storageclass yc-network-ssd \
-p '{"metadata": {"annotations":{"storageclass.kubernetes.io/is-default-class":"true
↪"}}}'
```

[37] https://yandex.cloud/ru/docs/managed-kubernetes/operations/connect/

ГЛАВА 6

Day 1: Развертывание и тестирование

Процесс развертывание платформы различается в зависимости от того, установлена ли у вас предыдущая версия Astra Automation 1.2:

- *Развертывание новой платформы*;
- *Миграция*.

6.1 Развертывание новой платформы

Следующей фазой после *планирования и подготовки ресурсов* является развертывание платформы с использованием выделенных ресурсов и внешних систем. Эта фаза состоит из следующих стадий:

Фаза планирования и подготовки развертывания Astra Automation состоит из нескольких стадий:

1. Настройка балансировщика нагрузки для повышения отказоустойчивости и распределения нагрузки. Настоятельно рекомендуется для топологии уровня предприятия:
   - при развертывании на ВМ и физических серверах;
   - при развертывании в двух и более кластерах Kubernetes, расположенных в разных зонах доступа.
2. Непосредственное развертывание и настройка узлов платформы.
3. Активация подписки.
4. Проверка работоспособности.

6.2 Миграция

Миграция может двух видов:

1. Миграция с предыдущей версии Astra Automation 1.2 на новую версию Astra Automation 2.0 с сохранением прежней модели развертывания на ВМ. После миграции на новую версию вы можете продолжать использование этой модели или мигрировать на модель развертывания в Kubernetes.
2. Миграция с модели развертывания на ВМ на новую модель, использующую Kubernetes.

По сравнению с предыдущей версией Astra Automation 2.0 – это новая архитектура платформы, в которой реализованы централизованное управление пользователями, шлюзовая

модель API и обновленные компоненты Event-Driven Automation. Переход на эту версию означает смену мажорного релиза, которая включает автоматические и ручные этапы переноса данных, пользователей и ролей.

6.2.1 Балансировка нагрузки

На этой стадии происходит настройка балансировщика нагрузки для топологии уровня предприятия.

Протестированным и наиболее распространенным вариантом обеспечения отказоустойчивости и равномерного распределения трафика между компонентами Astra Automation является балансировщик нагрузки HAProxy[38]. HAProxy работает на уровне TCP с использованием режима SSL passthrough, при котором шифрованный HTTPS-трафик передается напрямую на целевые узлы без расшифровки на уровне балансировщика. Для Astra Automation такими узлами являются BM или физические серверы, выделенные под шлюз платформы.

Предварительные требования

Начальные условия:

- у вас есть выделенный сервер или BM с установленной ОС, например, Astra Linux Special Edition, для установки HAProxy;
- подготовлено запланированное количество узлов для последующего развертывания на них шлюза платформы (Platform Gateway);
- у шлюза платформы есть доменное имя, например `aa.demo.example.com`.

Если вы настраиваете балансировщик для использовании модели развертывания в нескольких кластерах Kubernetes, то в качестве узлов надо использовать ingress-интерфейсы кластеров Kubernetes с распределением нагрузки между ними.

Пример настройки

Следующий пример демонстрирует настройку HAProxy для двух узлов будущего шлюза или интерфейсов кластеров Kubernetes.

Название узла	IP-адрес	Доменное имя	Компонент
gw1	192.168.56.11	aa.demo.example.com	Platform Gateway
gw2	192.168.56.12	aa.demo.example.com	Platform Gateway

Установка HAProxy

Для установки HAProxy выполните следующие действия на узле балансировщика:

1. Обновите список доступных пакетов:

```
sudo apt update
```

2. Установите пакет HAProxy:

```
sudo apt install haproxy --yes
```

3. Убедитесь, что служба запущена:

```
sudo systemctl status haproxy
```

   В выводе ожидается наличие следующей примерной строки:

```
Active: active (running) since Wed 2026-11-12 14:22:58 MSK; 1min 33s ago
```

[38] https://docs.haproxy.org/

Настройка HAProxy

Файл настройки HAProxy по умолчанию находится в каталоге `/etc/haproxy/`, обычно это `/etc/haproxy/haproxy.cfg`.

При настройке HAProxy обратите внимание на следующие особенности:

- HAProxy должен работать в режиме TCP, то есть для всех компонентов frontend и backend необходимо указать `mode tcp`, чтобы обеспечить передачу зашифрованного трафика без его расшифровки.
- Для маршрутизации трафика HTTPS по доменному имени сервера (SNI) настройте соответствующие правила, которые направляют соединения с конкретным доменным именем (aa.demo.example.com) в соответствующую группу компонентов backend.
- В настройках backend используйте стратегию балансировки `balance source`, которая обеспечивает постоянное распределение трафика от одного клиента на один и тот же сервер, основываясь на IP-адресе клиента. В документации HAProxy вы найдете также описание и других стратегий балансировки.
- Для каждого сервера в backend настройте проверку доступности с помощью опции check, чтобы HAProxy мог автоматически исключать недоступные серверы из списка рабочих узлов.
- Настройте административный интерфейс статистики HAProxy, доступный через порт `9000` по адресу `/stats`, с включенной базовой аутентификацией для мониторинга состояния компонентов backend и их нагрузки. Интерфейс мониторинга в этом случае доступен по адресу `http://<haproxy-ip>:9000/stats`. Вход осуществляется с помощью учетной записи и пароля, заданных в настройке, например, `admin:password`.

Ниже приведен пример настройки HAProxy для маршрутизации трафика HTTPS по SNI и балансировки нагрузки между несколькими backend компонентами. Каждая секция файла настройки (`загрузить файл`) описана отдельно.

Global

В секции `global` задаются глобальные параметры HAProxy.

```
global
    log /dev/log local0
    log /dev/log local1 notice
    chroot /var/lib/haproxy
    user haproxy
    group haproxy
    daemon
    stats socket /var/lib/haproxy/stats level admin
    tune.ssl.default-dh-param 2048
    maxconn 4096
```

Как правило, они задаются один раз и не требуют изменения после настройки. Некоторые из них имеют эквиваленты в командной строке. Полный список параметров доступен в документации HAProxy[39].

Для настройки HAProxy в Astra Automation используйте следующие параметры:

- `log` – настройка централизованного сбора сообщений от HAProxy через сервер системного журнала. В примере использованы следующие настройки:
  - Первая строка задает отправку записей журнала в системный сокет `/dev/log` с категорией `local0` без ограничения по уровню важности.
  - Вторая строка также задает отправку через сокет `/dev/log`, но с категорией `local1` и ограничением на уровень важности сообщений – будут фиксироваться только события с уровнем важности `notice` и выше.

[39] https://docs.haproxy.org/dev/configuration.html#3

- `chroot` – задание виртуального корневого каталога файловой системы для процессов HAProxy. Это повышает безопасность, ограничивая доступ HAProxy в файловой системе заданным каталогом.
- `user` – учетная запись, привилегии которой будут использовать процессы HAProxy.
- `group` – группа, в которую входит учетная запись для процессов HAProxy.
- `daemon` – запуск HAProxy в фоновом режиме.
- `stats socket` – Unix-сокет для доступа к статистике HAProxy.
- `tune.ssl.default-dh-param` – размер параметра Диффи-Хеллмана для соединений SSL.
- `maxconn` – максимальное количество одновременных соединений, которые HAProxy может обслуживать.

Defaults

Секция `defaults` определяет параметры, которые будут наследоваться всеми другими секциями, если они не переопределены.

```
defaults
    log global
    mode tcp
    option tcplog
    timeout connect 5s
    timeout client 50s
    timeout server 50s
    retries 3
```

Для настройки HAProxy в Astra Automation используйте следующие параметры:

- log – настройка централизованного сбора сообщений, в которой можно указать `log global` для использования параметров из секции `global`;
- mode – режим работы;
- option tcplog – включение расширенного ведения журнала соединений TCP;
- timeout connect – максимально допустимое время ожидания успешной попытки подключения к серверу;
- timeout client – максимально допустимое время ожидания активности от клиента;
- timeout server – максимально допустимое время ожидания активности от сервера;
- retries – максимально допустимое количество попыток переподключения к серверу при сбоях.

Frontend https-in

Frontend `https-in` принимает трафик на порту 443 и маршрутизирует его по SNI. Для разных доменных имен создаются правила, которые перенаправляют запросы соответствующим backend компонентам. В этом примере используется одно доменное имя.

```
# --- HTTPS passthrough frontend ---
frontend https-in
    bind *:443
    mode tcp

    tcp-request inspect-delay 5s
    tcp-request content accept if { req_ssl_hello_type 1 }

    acl is_gateway req.ssl_sni -i aa.demo.example.com
    use_backend gateway_backend if is_gateway
```

Для настройки HAProxy в Astra Automation используйте следующие параметры:

- bind – IP-адрес и порт для входящих соединений.
- mode – режим работы frontend.
- tcp-request – опции обработки запросов на уровне TCP:
  - `inspect-delay` – время ожидания анализа первых пакетов;
  - `content accept if { req_ssl_hello_type 1 }` – условие, при котором соединение принимается, например при обнаружении TLS ClientHello.
- acl – правила фильтрации трафика, например, по SNI.
- use_backend – компонент backend в который необходимо перенаправить трафик при выполнении условия.

Backend gateway_backend

Секция `backend gateway_backend` обеспечивает направление трафика если клиент обратился к домену `aa.demo.example.com`.

```
# --- Gateway backend ---
backend gateway_backend
    mode tcp
    balance source
    server ctrl1 192.168.56.11:443 check
    server ctrl2 192.168.56.12:443 check
```

Для настройки HAProxy в Astra Automation используйте следующие параметры:

- mode – режим работы серверов Automation Controller;
- balance – алгоритм балансировки нагрузки;
- server – сервер Automation Controller с IP-адресом, портом и проверкой доступности.

Listen stats

Секция `listen stats` включает встроенный веб-интерфейс статистики HAProxy, доступный, например, по адресу `http://<ip>:9000/stats`.

```
listen stats
    bind *:9000
    mode http
    stats enable
    stats uri /stats
    stats refresh 10s
    stats auth admin:password
    stats realm Haproxy\ Statistics
```

Для настройки HAProxy в Astra Automation используйте следующие параметры:

- bind – IP-адрес и порт веб-интерфейса статистики.
- mode – режим работы веб-интерфейса.
- stats – настройки веб-интерфейса статистики:
  - `enable` – включение отображения статистики;
  - `uri` – URI страницы статистики;
  - `refresh` – интервал обновления данных;
  - `auth` – логин и пароль для аутентификации;
  - `realm` – текстовое сообщение в диалоге аутентификации.

Добавление технического заголовка для идентификации узла

Для упрощения диагностики балансировки через HAProxy можно добавить дополнительный HTTP-заголовок X-Served-By. Для этого на всех узлах шлюза выполните следующие действия:

1. В каталоге /etc/nginx/conf.d/ создайте конфигурационный файл NGINX со следующим содержимым:

```
server {
      listen 443 ssl;
      server_name <hostname>;

      ssl_certificate     </path/to/cert.pem>;
      ssl_certificate_key </path/to/key.key>;

      location / {
               add_header X-Served-By "<host>";
      }
}
```

   Здесь:

   - <hostname> – доменное имя сервера (SNI), например, aa.demo.example.com;
   - <host> – название узла, например, gw1.

2. Обновите настройки сервиса nginx:

```
sudo systemctl reload nginx
```

3. Проверьте заголовок ответа:

```
curl -I https://aa.demo.example.com
```

   Ожидаемый результат выполнения команды:

```
X-Served-By: gw1
```

Обеспечение отказоустойчивости балансировщика

Отказоустойчивую архитектуру балансировщика нагрузки можно обеспечить путем создания двух или более узлов балансировщика и организации их работы одним из следующих способов:

- Использование VRRP и keepalived.

  При использовании физических серверов в одном сегменте сети доступен уровень L2 сети, что позволяет использовать сетевое оборудование с классическим механизмом VRRP для организации виртуального IP-адреса. В этом случае несколько узлов с установленным HAProxy объединяются в кластер, где один из узлов активен, а остальные работают в режиме ожидания. При отказе активного узла виртуальный IP-адрес автоматически переносится на резервный узел. Пример настройки VRRP с использованием Keepalived для физических серверов приведен в документации Yandex Cloud[40].

- Использование внешнего балансировщика уровня TCP.

  В инфраструктуре, где нет поддержки VRRP (например, в облачных средах без доступа к протоколам L2), рекомендуется использовать внешний сетевой балансировщик, который обеспечивает распределение соединений TCP между несколькими узлами с HAProxy и поддерживает проверку их доступности. В качестве примера можно использовать Yandex Network Load Balancer[41], который работает на уровне TCP и автоматически исключает недоступные узлы из схемы балансировки.

[40] https://yandex.cloud/ru/docs/baremetal/tutorials/bms-vrf-routing
[41] https://yandex.cloud/ru/docs/network-load-balancer/

В результате отказ одного из узлов балансировщика не приведет к недоступности сервисов Astra Automation, сетевое оборудование или внешний балансировщик будет автоматически перенаправлять трафик на резервный узел или узлы.

6.2.2 Развертывание Astra Automation

На этой стадии происходит непосредственное развертывание платформы.

В зависимости от выбранного варианта развертывания далее представлены два способа:

- развертывание *на базе виртуальных машин* (или физических узлов);
- развертывание *в кластере Kubernetes*.

Каждый из этих способов предполагает, что вы выполнили подготовку ресурсов в соответствие с *рекомендациями по планированию и подготовке*.

Развертывание на виртуальных машинах

На этой стадии происходит развертывание платформы на виртуальных машинах.

Развертывание платформы необходимо производить из установочного узла с помощью утилиты `aa-setup`.

Проверка ресурсов

Перед началом развертывания платформы убедитесь, что вы подготовили необходимые для этого ресурсы:

Ресурс	*Базовая топология*	*Топология уровня предприятия*
Установочный узел	+	+
Узлы платформы	6	11+
Сетевое хранилище для Private Automation Hub	-	*Требования*
Настройка сетевых фильтров	*Требования*	*Требования*
Плоскость исполнения	*Исполняющий узел*	*Плоскость исполнения*
СУБД	Будет создана при развертывании Astra Automation	*Внешняя СУБД*
Описание инвентаря	*Шаблон*	*Шаблон*
Балансировщик нагрузки	-	+

Особенности использования `aa-setup`

Утилита `aa-setup` расположена в каталоге `/opt/rbta/aa/astra-automation-setup/`. Ее запускают оттуда с привилегиями администратора, например с помощью `sudo`, или с привилегиями активного пользователя. Во втором случае ее работа может завершиться ошибкой в следующих случаях:

- В описании инвентаря не указаны значения некоторых параметров и нет разрешения на запись в файл инвентаря для установки значений по умолчанию.

  Способы решения проблемы:

  - Укажите в описании инвентаря значения всех обязательных параметров.
  - Предоставьте активному пользователю разрешение на запись в файл инвентаря. Если для этого надо переместить файл в другой каталог, укажите путь к нему при запуске утилиты `aa-setup` с помощью аргумента *`--inventory`* (`-i`).

- `aa-setup` запускается без аргумента *`--log-path`*, но у активного пользователя нет прав на запись в каталог установщика.

Для избежания этой ошибки используйте аргумент *--log-path*, указав с помощью него каталог, в котором утилита будет сохранять журнал и на который у пользователя есть привилегии на запись.

Процедура установки

В каталоге `/opt/rbta/aa/astra-automation-setup/` выполните команду:

```
sudo ./aa-setup
```

Примечание

При установке Astra Automation без использования интернета убедитесь, что установочный узел настроен согласно *инструкции*.

Если описание инвентаря хранится в файле, путь к которому отличается от значения по умолчанию (`inventory` в каталоге установщика), укажите путь к нему в значении аргумента *--inventory* (`-i`):

```
sudo ./aa-setup --inventory=</path/to/inventory.yml>
```

Если секретная информация содержится в отдельном файле, например `secrets.yml`, *защищенном с помощью Ansible Vault*, используйте специальные аргументы для добавления этого файла:

```
sudo ./aa-setup -- --extra-vars @secrets.yml --ask-vault-pass
```

По запросу введите пароль, с помощью которого зашифровано содержимое файла.

Примечание

Синтаксис и полный список параметров утилиты `aa-setup` приведены в *справочнике*.

Дождитесь завершения выполнения команды, это может занять некоторое время. По окончании установки в терминал выводится набор строк вида (названия узлов и их количество должны совпадать со значениями, указанными в файле инвентаря):

```
PLAY RECAP *********************************************************************
10.177.92.104              : ok=98   changed=35   unreachable=0    failed=0
↪skipped=75   rescued=0    ignored=3
10.177.92.128              : ok=231  changed=95   unreachable=0    failed=0
↪skipped=136  rescued=0    ignored=5
10.177.92.135              : ok=190  changed=75   unreachable=0    failed=0
↪skipped=158  rescued=0    ignored=5
10.177.92.48               : ok=227  changed=95   unreachable=0    failed=0
↪skipped=169  rescued=0    ignored=5
10.177.92.64               : ok=330  changed=153  unreachable=0    failed=0
↪skipped=184  rescued=0    ignored=9
10.177.92.69               : ok=144  changed=60   unreachable=0    failed=0
↪skipped=102  rescued=0    ignored=4
localhost                  : ok=0    changed=0    unreachable=0    failed=0    skipped=1
↪   rescued=0    ignored=0

_____________________________________________
Astra Automation Setup has been finished
_____________________________________________
Astra Automation host:
https://10.177.92.135/

Login: admin
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
Password: admin
_____________________________________
```

Развертывание считается успешным, если для всех узлов в поле `failed` указано значение `0`.

Развертывание в кластере Kubernetes

На этой стадии происходит развертывание платформы в кластере Kubernetes.

Перед началом развертывания платформы убедитесь, что вы выбрали вариант топологии и подготовили необходимые для этого ресурсы, минимальные требования к которым представлены в следующей таблице.

Ресурс	*Базовая топология*	*Топология уровня предприятия*
Рабочая станция	Утилиты `kubectl`, `ansible-navigator`	Утилиты `kubectl`, `ansible-navigator`
Кластер Kubernetes	С одним *рабочим узлом*	С тремя *рабочими узлами*
Сетевое хранилище для Private Automation Hub	*Требования*	*Требования*
Настройка сетевых фильтров	*Требования*	*Требования*
Плоскость исполнения	*Исполняющий узел*	*Плоскость исполнения*
СУБД	Будет создана при развертывании Astra Automation	*Внешняя СУБД*
Манифесты секретов	- *Доступ к шлюзу по HTTPS* - *Пароль для доступа к шлюзу* - *Доступ к внешнему хранилищу контента*	- *Доступ к шлюзу по HTTPS* - *Пароль для доступа к шлюзу* - *Доступ к внешнему хранилищу контента* - *Ключи шифрования данных в базах* - *Доступ к базам данных*
Манифест приложения	*Пример*	*Пример*

Установка контроллера Ingress

Если отсутствует контроллер Ingress, установите его с помощью менеджер пакетов Helm:

1. Добавьте пакет Ingress в индексную базу Helm:

   ```
   helm repo add ingress-nginx https://kubernetes.github.io/ingress-nginx
   ```

   Информационное сообщение от команды:

   ```
   "ingress-nginx" has been added to your repositories
   ```

2. Обновите репозиторий Helm:

   ```
   helm repo update
   ```

Информационное сообщение от команды:

```
...Successfully got an update from the "ingress-nginx" chart repository
Update Complete. ⎈Happy Helming!⎈
```

Примечание

Обе предыдущие команды выполняют настройку на вашей рабочей станции, где вы используете утилиту helm. Поэтому, если по какой-либо причине вам надо повторить развертывание, можно эти шаги пропустить.

3. Установите сервис Ingress в Kubernetes:

```
helm install ingress-nginx ingress-nginx/ingress-nginx
```

Информационное сообщение от команды:

Installation output

```
NAME: ingress-nginx
LAST DEPLOYED: Tue Oct 14 11:33:56 2025
NAMESPACE: default
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
The ingress-nginx controller has been installed.
It may take a few minutes for the load balancer IP to be available.
You can watch the status by running 'kubectl get service --namespace default␣
↪ingress-nginx-controller --output wide --watch'

An example Ingress that makes use of the controller:
  apiVersion: networking.k8s.io/v1
  kind: Ingress
  metadata:
    name: example
    namespace: foo
  spec:
    ingressClassName: nginx
    rules:
      - host: www.example.com
        http:
          paths:
            - pathType: Prefix
              backend:
                service:
                  name: exampleService
                  port:
                    number: 80
              path: /
    # This section is only required if TLS is to be enabled for the Ingress
    tls:
      - hosts:
        - www.example.com
        secretName: example-tls

If TLS is enabled for the Ingress, a Secret containing the certificate and key must␣
↪also be provided:

  apiVersion: v1
  kind: Secret
  metadata:
    name: example-tls
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
    namespace: foo
  data:
    tls.crt: <base64 encoded cert>
    tls.key: <base64 encoded key>
  type: kubernetes.io/tls
```

4. Настройте сервис Ingress на возможность установления соединения через WebSocket между сервисами receptor на внешних исполняющих узлах (execution nodes) и внутреннем переходном узле (hop node), не терминируя соединения, использующие TLS/SSL:

```
helm upgrade ingress-nginx ingress-nginx/ingress-nginx --set controller.extraArgs.
↪enable-ssl-passthrough=true
```

Установка операторов

Установите все операторы Astra Automation:

```
kubectl apply -f operators/
```

Информационное сообщение от команды:

Installation output

```
namespace/astra-automation created
customresourcedefinition.apiextensions.k8s.io/aabackups.aa.astra-automation.ru created
customresourcedefinition.apiextensions.k8s.io/aarestores.aa.astra-automation.ru created
customresourcedefinition.apiextensions.k8s.io/aas.aa.astra-automation.ru created
serviceaccount/aa-operator-aa-manager created
role.rbac.authorization.k8s.io/aa-operator-leader-election-role created
role.rbac.authorization.k8s.io/aa-operator-manager-role created
clusterrole.rbac.authorization.k8s.io/aa-operator-metrics-reader created
clusterrole.rbac.authorization.k8s.io/aa-operator-proxy-role created
rolebinding.rbac.authorization.k8s.io/aa-operator-leader-election-rolebinding created
rolebinding.rbac.authorization.k8s.io/aa-operator-manager-rolebinding created
clusterrolebinding.rbac.authorization.k8s.io/aa-operator-proxy-rolebinding created
service/aa-operator-aa-manager-metrics-service created
deployment.apps/aa-operator-aa-manager created
namespace/astra-automation configured
customresourcedefinition.apiextensions.k8s.io/acbackups.ac.astra-automation.ru created
customresourcedefinition.apiextensions.k8s.io/acmeshingresses.ac.astra-automation.ru␣
↪created
customresourcedefinition.apiextensions.k8s.io/acrestores.ac.astra-automation.ru created
customresourcedefinition.apiextensions.k8s.io/acs.ac.astra-automation.ru created
serviceaccount/ac-operator-ac-manager created
role.rbac.authorization.k8s.io/ac-operator-ac-manager-role created
role.rbac.authorization.k8s.io/ac-operator-leader-election-role created
clusterrole.rbac.authorization.k8s.io/ac-operator-metrics-reader created
clusterrole.rbac.authorization.k8s.io/ac-operator-proxy-role created
rolebinding.rbac.authorization.k8s.io/ac-operator-ac-manager-rolebinding created
rolebinding.rbac.authorization.k8s.io/ac-operator-leader-election-rolebinding created
clusterrolebinding.rbac.authorization.k8s.io/ac-operator-proxy-rolebinding created
configmap/ac-operator-ac-manager-config created
service/ac-operator-automationcontroller-manager-metrics-service created
deployment.apps/ac-operator-ac-manager created
namespace/astra-automation configured
customresourcedefinition.apiextensions.k8s.io/pahbackups.pah.astra-automation.ru created
customresourcedefinition.apiextensions.k8s.io/pahrestores.pah.astra-automation.ru created
customresourcedefinition.apiextensions.k8s.io/pahs.pah.astra-automation.ru created
serviceaccount/pah-operator-sa created
role.rbac.authorization.k8s.io/pah-operator-leader-election-role created
role.rbac.authorization.k8s.io/pah-operator-pah-operator-role created
role.rbac.authorization.k8s.io/pah-operator-proxy-role created
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
clusterrole.rbac.authorization.k8s.io/pah-operator-metrics-reader created
clusterrole.rbac.authorization.k8s.io/pah-operator-pah-operator-cluster-role created
rolebinding.rbac.authorization.k8s.io/pah-operator-leader-election-rolebinding created
rolebinding.rbac.authorization.k8s.io/pah-operator-pah-operator-rolebinding created
rolebinding.rbac.authorization.k8s.io/pah-operator-proxy-rolebinding created
clusterrolebinding.rbac.authorization.k8s.io/pah-operator-pah-operator-cluster-
↪rolebinding created
configmap/pah-operator-pah-operator-config created
service/pah-operator-pah-manager-metrics-service created
deployment.apps/pah-operator-pah-manager created
namespace/astra-automation configured
customresourcedefinition.apiextensions.k8s.io/edabackups.eda.astra-automation.ru created
customresourcedefinition.apiextensions.k8s.io/edarestores.eda.astra-automation.ru created
customresourcedefinition.apiextensions.k8s.io/edas.eda.astra-automation.ru created
serviceaccount/eda-operator-eda-manager created
role.rbac.authorization.k8s.io/eda-operator-eda-manager-role created
role.rbac.authorization.k8s.io/eda-operator-leader-election-role created
clusterrole.rbac.authorization.k8s.io/eda-operator-metrics-reader created
clusterrole.rbac.authorization.k8s.io/eda-operator-proxy-role created
rolebinding.rbac.authorization.k8s.io/eda-operator-eda-manager-rolebinding created
rolebinding.rbac.authorization.k8s.io/eda-operator-leader-election-rolebinding created
clusterrolebinding.rbac.authorization.k8s.io/eda-operator-proxy-rolebinding created
service/eda-operator-eda-manager-metrics-service created
deployment.apps/eda-operator-eda-manager created
```

Команда создает новое пространство имен – `astra-automation` – с операторами внутри нее. В этом пространстве будут создаваться все объекты центральной части платформы. Проверьте готовность подов:

```
kubectl -n astra-automation get pods
```

Они должны быть в состоянии Running:

```
NAME                                          READY   STATUS    RESTARTS   AGE
aa-operator-aa-manager-76d84985d9-bpll6       2/2     Running   0          2m19s
ac-operator-ac-manager-748f66f8b8-jnf4x       2/2     Running   0          2m16s
eda-operator-eda-manager-644b75748d-7dlzr     2/2     Running   0          2m11s
pah-operator-pah-manager-68c859659b-6jsv4     2/2     Running   0          2m14s
```

Создание секретов

Для каждой группы секретов или одиночных секретов у вас есть файл с манифестом. Создайте необходимые секреты с помощью этих файлов, используя команду вида:

```
kubectl apply -f <path/to/file.yml>
```

Здесь <path/to/file.yml> – путь к файлу с манифестом.

Развертывание базовой топологии Astra Automation

Развертывание полной платформы состоит из следующих этапов.

Развертывание центральных компонентов

Для развертывания центральной части необходимо применить ранее подготовленный манифест:

```
kubectl apply -f <path/to/manifest.yml>
```

Процесс может занять около 20 минут. Окончание развертывания проверяйте по журналу:

```
kubectl logs -n astra-automation aa-operator-aa-manager-76d84985d9-bpll6 -c manager
```

Завершение обозначено заключительной секцией как при обычном выполнении набора сценариев Ansible.

Первичная настройка

Проверьте с помощью браузера, что вы можете подключиться к шлюзу, используя URL https://<ip or domain.name>, по его IP-адресу или доменному имени и пройти аутентификацию с учетной записью администратора.

После этого *активируйте подписку*, чтобы перейти в панель управления Astra Automation.

Добавление исполняющего узла

Согласно *схеме* необходимо добавить внешний исполняющий узел, связанный с управляющим узлом через рецепторы:

1. На панели навигации выберите **Automation Execution > Infrastructure > Instances** и начните добавлять узел, нажав на кнопку **Create instance**.
2. В окне **Create instance** настройте следующие поля:
   - **Host name**: <IP address>;
   - **Listener port**: 27199;
   - **Instance type**: Execution;
   - **Options**: выберите все опции: **Enable instance**, **Managed by policy** и **Peers from control panel**.
3. После применения этих настроек открывается окно с кнопкой **Download bundle** для выгрузки установочного пакета. Выгрузите этот пакет на вашу станцию и распакуйте этот пакет. Для использования загруженного пакета на станции должен быть установлен пакет Ansible Core. Перейдите в распакованный каталог.
4. Убедитесь, что в файле описания инвентаря `inventory.yml` правильно настроены следующие параметров (согласно настройкам в добавляемом узле):
   - `ansible_user` – название учетной записи с административными привилегиями, например admin;
   - `ansible_ssh_private_key_file` – путь к приватному ключу SSH для этой учетной записи, например `~/.ssh/id_ed25519`.
5. Выполните сценарий настройки узла:

   ```
   ansible-navigator run --eei hub.astra-automation.ru/aa-2.0/aa-full-ee:latest \
   install_receptor.yml -i inventory.yml -m stdout
   ```

   По завершении выполнения сценария выводится строка вида:

   ```
   PLAY RECAP␣
   ↪*********************************************************************************************
   remote-execution  : ok=71   changed=40   unreachable=0    failed=0    skipped=10  ␣
   ↪rescued=0    ignored=0
   ```

6. В графической консоли платформы убедитесь, что состояние добавленного узла стало Ready.

Развертывание топологии уровня предприятия

Процесс состоит из следующих стадий.

Развертывание центральной части

Выполните следующие шаги для развертывания отказоустойчивой платформы:

1. Задайте количество реплик, например 3, для каждого установленного оператора:

```
kubectl scale deployment aa-operator-aa-manager --namespace astra-automation --
↪replicas=3
kubectl scale deployment ac-operator-ac-manager --namespace astra-automation --
↪replicas=3
kubectl scale deployment eda-operator-eda-manager --namespace astra-automation --
↪replicas=3
kubectl scale deployment pah-operator-pah-manager --namespace astra-automation --
↪replicas=3
```

   После выполнения команд, каждый объект deployment должен содержать три реплики:

```
kubectl -n astra-automation get deployments
```

2. Убедитесь, что на стадии подготовки вы настроили файлы с манифестами секретов в соответствии с рекомендациями, и примените каждый из них с помощью команды вида:

```
kubectl apply -f <path/to/secret.yml>
```

   где <path/to/secret.yaml> – путь к файлу, содержащему манифест соответствующих секретов в формате YAML.

   У вас должны быть файлы с манифестами для следующих секретов:

   - доступ к хранилищу S3 для Private Automation Hub;
   - доступы к базам данных PosgreSQL для всех компонентов платформы;
   - секреты для шифрования чувствительных данных в базах данных;
   - пароль администратора платформы;
   - параметры TLS для защиты данных, передаваемых по сети;

3. Запустите процесс развертывания платформы:

```
kubectl apply -f <path/to/application-manifest.yml>
```

   где <path/to/application-manifest.yml> – путь к файлу, содержащему *манифест приложения*.

4. Если необходимо обеспечить возможность установления соединений от исполнительных узлов к Automation Controller, добавьте Mesh Ingress:

```
kubectl apply -f <path/to/mesh-ingress.yml>
```

   где <path/to/mesh-ingress.yml> – путь к файлу, содержащему *манифест Mesh Ingress*.

Контроль процесса развертывания

После выполнения команды развертывания платформы потребуется некоторое время на запуск и настройку всех подов. Состояние развертывания можно проверить с помощью команды:

```
watch -n 5 "kubectl get aas aa-demo -n astra-automation -o jsonpath='{.status.conditions}
↪'"
```

где `aa-demo` – название приложения, которое вы указали в манифесте приложения.

Выход из команды производится с помощью комбинации клавиш `Ctrl+C`.

При установленной утилите `jq` можно получить более удобный формат вывода:

```
watch -n 5 "kubectl get aas aa-demo -n astra-automation -o jsonpath='{.status.conditions}
↪' | jq "
```

По завершении развертывания платформы вывод имеет следующий вид:


```
[
   {
      "lastTransitionTime": "2025-10-27T14:45:52Z",
      "message": "",
      "reason": "",
      "status": "False",
      "type": "Failure"
   },
   {
      "lastTransitionTime": "2025-10-27T14:45:52Z",
      "message": "Last reconciliation succeeded",
      "reason": "Successful",
      "status": "True",
      "type": "Successful"
   },
   {
      "lastTransitionTime": "2025-10-27T14:45:52Z",
      "message": "Awaiting next reconciliation",
      "reason": "Successful",
      "status": "True",
      "type": "Running"
   }
]
```


Ключевые моменты:

- В секции, где `"type": "Failure"`, поле `message` пустое.
- В секции, где `"type": "Successful"`, поле `message` содержит сообщение `Last reconciliation succeeded`.
- В секции, где `"type": "Running"`, поле `message` содержит сообщение `Awaiting next reconciliation`.

Для более подробной информации следует изучить журнал выполнения сценария развертывания, предварительно узнав полное название пода `aa-operator-aa-manager`.

1. Узнайте полное название пода `aa-operator-aa-manager`:

   ```
   kubectl get pods -n astra-automation | grep aa-operator
   ```

   Выходные данные имеют следующий вид:

   ```
   aa-operator-aa-manager-76d84985d9-hmswf     2/2     Running   0          6m33s
   ```

2. Используйте полное название пода `aa-operator-aa-manager` для просмотра журнала выполнения сценария развертывания:

   ```
   kubectl logs -n astra-automation -f <aa-operator-aa-manager full name> -c manager
   ```

   Аргумент `-f` требует постоянного обновления данных, полученных из журнала.

Журнал показывает ход выполнения сценария и возникающие ошибки. Итоговый вывод PLAY RECAP сообщает об успешном или неудачном завершении сценария. Продукт полностью развернут при успешном выполнении сценария и PLAY RECAP с полем failed=0 вида:


```
{"level":"info","ts":"2025-10-21T11:10:20Z","logger":"runner","msg":"Ansible-runner␣
↪exited successfully","job":"182878699511911518","name":"aa-demo","namespace":"astra-
↪automation"}

----- Ansible Task Status Event StdOut (aa.astra-automation.ru/v1alpha1,␣
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
↪Kind=AstraAutomation, aa-demo/astra-automation) -----


PLAY RECAP *********************************************************************
localhost                  : ok=195  changed=2    unreachable=0    failed=0    ␣
↪skipped=103  rescued=0    ignored=0
```

Активация лицензии

С помощью браузера обратитесь по URL, настроенному в манифесте приложения, например `https://aa.demo.example.com/`. Активируйте *лицензию подписки*.

Добавление плоскости исполнения

Согласно *рекомендуемой топологии* необходимо добавить два исполнительных узла, один из которых должен быть связан с плоскостью управления через промежуточный узел. Здесь можно применить один из подходов:

- Добавить каждый из узлов последовательно, как описано в *базовой топологии*. Для подключения узла типа execution через узел hop, надо сначала добавить узел hop, а затем execution.
- Добавить все узлы сразу с помощью утилиты `ansible-navigator`.

Рассмотрим второй вариант, поскольку предыдущий был уже описан ранее.

1. Подготовьте приватный ключ SSH, который предоставляет пользователю доступ к добавляемым узлам с привилегиями администратора.
2. Убедитесь, что у вас подготовлены узлы для плоскости исполнения в соответствии с *рекомендациями*.
3. Создайте токен доступа для пользователя admin:
   - В панели навигации веб-интерфейса перейдите к **Access Management > Users**.
   - В списке пользователей выберите admin.
   - Переключитесь на вкладку **Tokens** и нажмите кнопку **Create token**. В поле **Scope** выберите Write. Нажмите внизу кнопку **Create token** и сохраните токен в надежном месте, поскольку после закрытия окна он не будет более доступен.
4. Подготовьте файл описания инвентаря, например `inventory.yml`, следующего вида:

```yaml
---
all:
  vars:
    ac_host: "https://aa.demo.example.com"
    ac_token: "6JShbAmkz9k5QyMKGX6ZUGTCjfZzJs"
    ansible_ssh_common_args: "-o StrictHostKeyChecking=no -o UserKnownHostsFile=/
↪dev/null"

  children:
    hop_nodes:
      hosts:
        hop-node-01:
          ansible_host: 10.177.93.56
          ansible_user: admin
          ansible_ssh_private_key_file: keys/id_ed25519
    execution_nodes:
      hosts:
        exec-node-01:
          ansible_host: 10.177.93.13
          ansible_user: admin
          ansible_ssh_private_key_file: keys/id_ed25519
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
        hop_node_related: "hop-node-01"
        peers_from_control_nodes: false
```

Файл содержит настройки общих переменных и настройки параметров доступа для переходных узлов (группа hop_nodes) и исполняющих узлов (группа execution_nodes). В файле необходимо задать следующие переменные в соответствии с параметрами добавляемых узлов:

- ac_host: URL к шлюзу платформы;
- ac_token: подготовленный токен для пользователя admin;
- ansible_host: IP-адрес соответствующего узла;
- ansible_user: название учетной записи с привилегиями администратора на соответствующем узле;
- ansible_ssh_private_key_file: относительный путь к приватному ключу SSH для этой учетной записи;
- hop_node_related (требуется, если подключение идет через hop node): параметр подключения к узлу hop в виде значения inventory_hostname или receptor_address ID, если узел hop уже существует;
- peers_from_control_nodes: true, если подключение будет напрямую к плоскости управления (предыдущий параметр игнорируется), или false, если через hop.

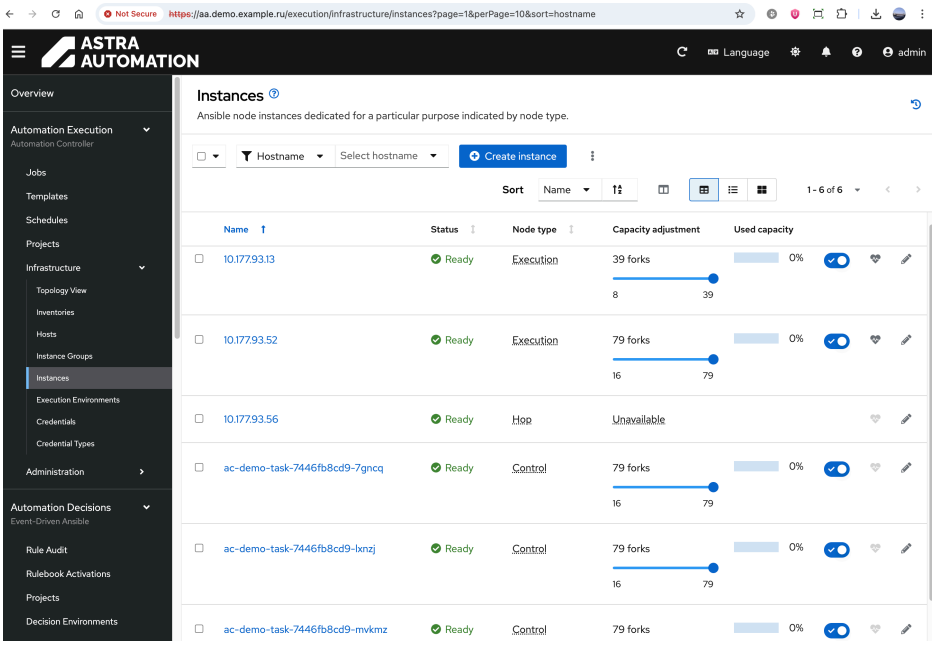
5. Выполните сценарии настройки и подключения узлов:

```
ansible-navigator run --eei hub.astra-automation.ru/aa-2.0/aa-full-ee:latest \
<playbook.yml> -i inventory.yml -m stdout
```

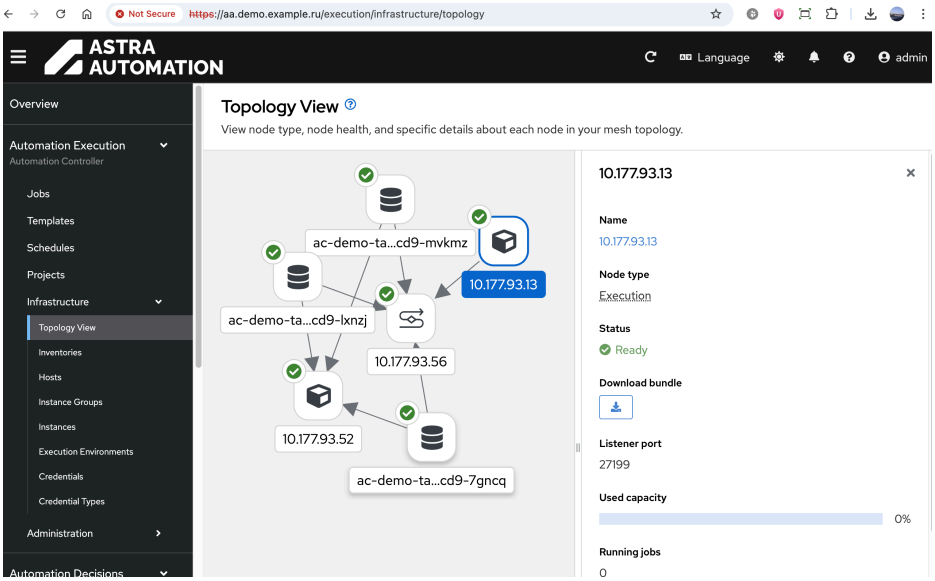
По завершении выполнения сценария Ansible выводит типовую секция PLAY RECAP.

6. В графической консоли платформы убедитесь, что появились новые узлы плоскости исполнения и их состояние Ready. Для этого на панели выберите следующие окна:

- **Automation Execution > Infrastructure > Instances** для просмотра списка узлов, например:



- **Automation Execution > Infrastructure > Topology View** для просмотра топологии, например:



6.2.3 Управление подпиской

На этой стадии происходит активация лицензии на использование Astra Automation.

Для разблокировки всех возможностей платформы необходимо иметь активированную подписку на Astra Automation. Управление подпиской включает следующие шаги:

1. *Создание подписки*.
2. Активация подписки:
   - *в окружении с доступом к интернету*;
   - *в окружении без доступа к интернету*.
3. *Управление лицензиями*.

Создание подписки

Для создания подписки выполните следующие действия:

1. Авторизуйтесь в личном кабинете[42] и перейдите в раздел **Активация лицензий**.
2. Нажмите кнопку *Создать подписку*.
3. В открывшемся окне **Конструктор подписок** задайте количество узлов, на которые активируется подписка:
   - чтобы активировать лицензию на все узлы, доступные по лицензии, нажмите кнопку *Активировать*;
   - чтобы активировать лицензию только на часть узлов, доступных по лицензии, укажите нужное значение в поле **Использовать** и нажмите кнопку *Собрать и активировать*.

   На странице **Активация лицензий** отобразится созданная подписка со статусом **Активирован**.
4. Если платформа развернута в окружении без доступа к интернету, нажмите кнопку *Скачать манифест*.

   После загрузки ZIP-архива с манифестом статус подписки изменится на **Использован**, а тип активации на **Оффлайн**.

[42] https://lk.astra.ru/license-activations

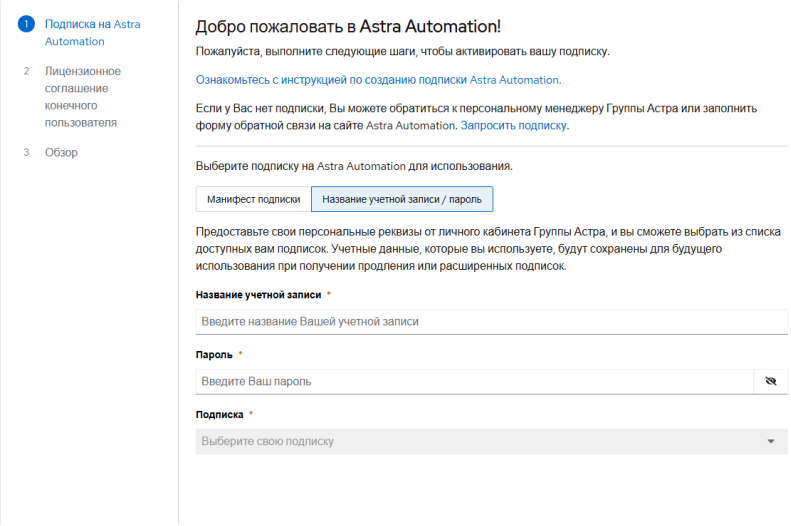
Активация в окружении с доступом к интернету

Если платформа развернута в окружении с доступом к интернету, для активации выполните следующие действия:

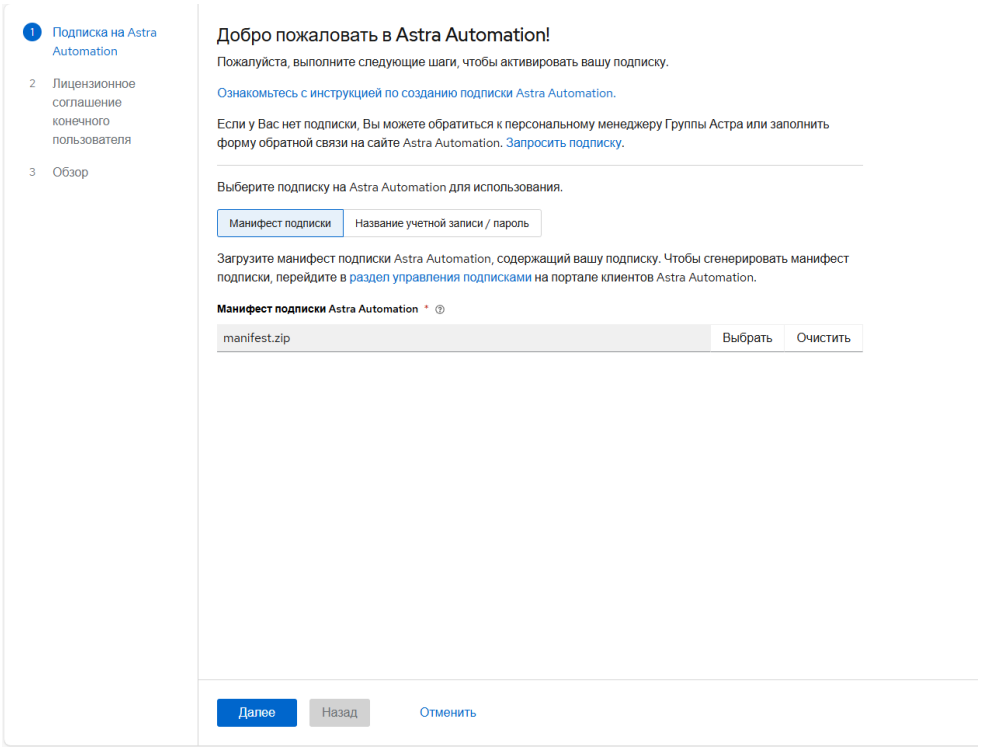
1. Откройте графическую консоль Astra Automation с привилегиями администратора.

   Язык текста зависит от настроек браузера. Выберите удобный для себя язык с помощью соответствующего переключателя в верхнем правом углу экрана.

2. В окне активации выберите вкладку **Имя пользователя / пароль** (Username / password).



3. В полях **Имя пользователя** (Username) и **Пароль** (Password) укажите учетные данные, используемые для доступа к Личному кабинету.
4. В поле **Подписка** (Subscription) выберите нужную подписку из выпадающего списка.



5. Нажмите кнопку *Далее* (Next) внизу.
6. Ознакомьтесь с лицензионным соглашением и нажмите кнопку *Далее* (Next).
7. В окне **Обзор** (Review) нажмите кнопку **Завершить** (Submit).

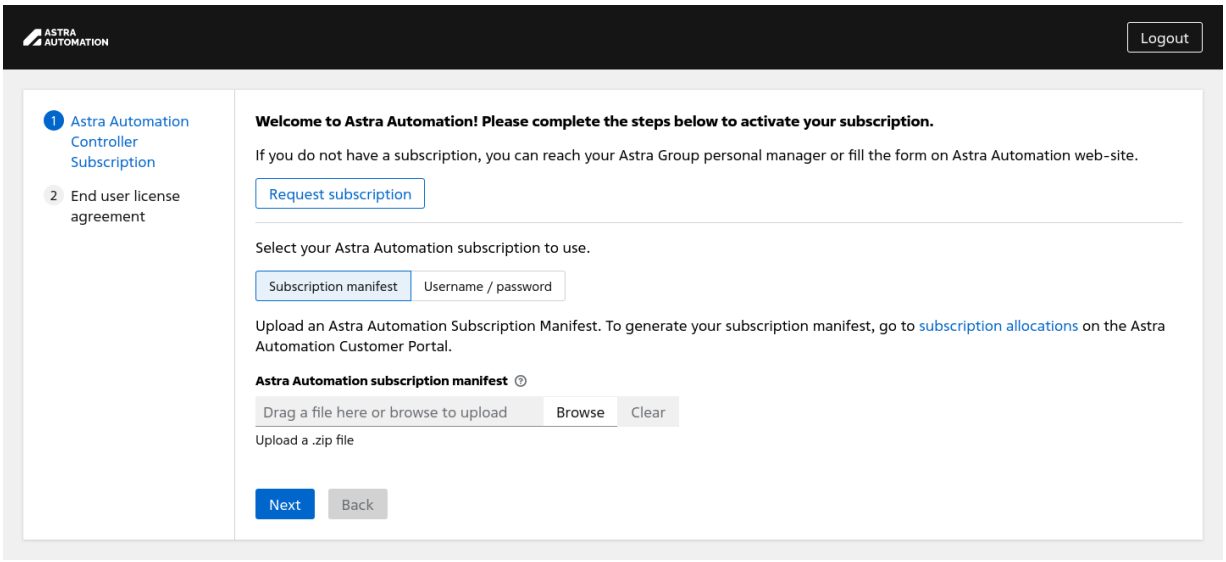
Система активирует подписку и автоматически отобразит в браузере информационную панель графической консоли.

В Личном кабинете статус подписки изменится на **Использован**, а тип активации на **Онлайн**.

Активация в окружении без доступа к интернету

Если платформа развернута в окружении без доступа к интернету, для активации выполните следующие действия:

1. Авторизуйтесь в Automation Controller от имени пользователя с правами администратора.
2. В окне активации выберите вкладку **Манифест подписки** (Subscription manifest).



3. В поле **Манифест подписки Astra Automation** (Astra Automation subscription manifest) выберите ZIP-архив с файлом манифеста, загруженный ранее из Личного кабинета.
4. Нажмите кнопку *Продолжить* (Next).
5. Ознакомьтесь с лицензионным соглашением и нажмите кнопку *Отправить* (Submit).

   Automation Controller активирует подписку и автоматически отобразит в браузере информационную панель[43].

6.2.4 Предварительное тестирование

На этой стадии происходит активация лицензии на использование Astra Automation.

Предварительное тестирование (smoke-test) позволяет быстро убедиться, что платформа готова к работе после установки. Набор следующих проверок предназначен для выполнения вручную в веб-консоли. Тестирование не зависит от того, была ли платформа развернута на виртуальных машинах или в кластерной среде Kubernetes.

Основные проверки

Следующие тесты проверяют доступность и базовую функциональность ключевых компонентов через веб-интерфейс шлюза платформы:

1. Проверка доступности веб-интерфейса:
   - Перейдите в браузере по адресу платформы и убедитесь, что страница входа загружается без ошибок.
   - Проверьте корректность сертификата HTTPS и наличие защищённого соединения.

[43] https://docs.astra-automation.ru/latest/controller/ui/views/dashboard/#aac-ui-dashboard

- Выполните аутентификацию под учетной записью администратора и убедитесь, что основная панель загружается корректно.

2. Проверка каждого компонента интерфейса:
   - **Automation Execution**:
     - Перейдите в раздел, предоставляющий функциональность Automation Controller.
     - Убедитесь, что доступны списки узлов платформы, шаблонов заданий и потоков заданий.
   - **Automation Content**:
     - Перейдите в раздел управления контентом (Private Automation Hub).
     - Проверьте доступность репозиториев и синхронизацию с Ansible Galaxy.
   - **Automation Decision**:
     - Перейдите в раздел Event-Driven Automation (EDA).
     - Убедитесь, что отображаются правила событий и состояния сервисов.
   - **Access Management**:
     - Перейдите в раздел управления доступом.
     - Проверьте возможность просмотра и редактирования пользователей, ролей и групп.
3. Запуск заданий из раздела **Automation Execution**:
   - В списке узлов (**Hosts**) в Demo Inventory выполните команду (кнопка **Run command**) с помощью модуля `ping` и убедитесь, что она завершилась со статусом *Success*.
   - Запустите задание из тестового шаблона и убедитесь, что оно завершилось со статусом *Success*.

Оценка результатов

Smoke-test через веб-интерфейс считается успешным, если все основные элементы интерфейса доступны, а тестовое задание завершилось без ошибок. В случае появления сбоев необходимо проверить конфигурацию сервисов Controller, Redis и PostgreSQL, а также сетевое взаимодействие между компонентами. Успешное завершение теста подтверждает готовность платформы к дальнейшей эксплуатации и интеграционному тестированию.

6.2.5 Миграция на версию 2.0

Подготовка и миграция Astra Automation на версию 2.0 состоит из нескольких стадий:

Для перехода сразу на некоторую стадию выберите соответствующий блок диаграммы.

Миграция платформы на виртуальных машинах в облачном пространстве или в собственном центре обработки данных является традиционным способом. Если вам больше подходит использование физических серверов вместо ВМ, то приведенные далее инструкции не меняются по сути.

Цель миграции

Цель миграции – безопасно обновить Astra Automation до версии 2.0, сохранив данные, проекты, коллекции, среды выполнения, расписания и историю заданий.

Обновление выполняется путем создания экземпляра платформы на новых узлах с обновлением ее внутренних ресурсов из резервной копии. Такой подход исключает риски простоя и обеспечивает полный контроль за изменениями в новой архитектуре и модели доступа. Рабочее окружение может быть на серверах в вашем центре обработки данных или в окружении облачного провайдера.

В процессе обновления используется утилита `aa-setup`.

Что переносится автоматически

При обновлении Astra Automation до версии 2.0 выполняется автоматическая миграция следующих элементов:

- База данных Automation Controller и Private Automation Hub.
- Проекты, инвентари и шаблоны заданий.
- Коллекции в Private Automation Hub и *EE*.
- Пользователи, команды пользователей и организации:
  - При наличии дублированных учетных записей приоритет отдается записям из Automation Controller.
  - Роли сохраняются на уровне компонентов до завершения связывания с ними учетных записей.
  - После завершения связывания учетных записей все роли автоматически переносятся в шлюз.
- История выполнения заданий.

Что не переносится автоматически

Следующие элементы не переносятся автоматически и требуют дополнительной ручной настройки:

- привязка ролей к пользователям (новая модель *RBAC* на уровне шлюза);
- настройка SSO и LDAP – требуется полное изменение через настройку шлюза;
- своды правил, их активации и источники событий Event-Driven Automation;
- база данных Event-Driven Automation;
- OAuth-приложения Automation Controller;
- конфигурации групп узлов контроллера;
- пользовательские настройки с прямыми URL внутренних узлов.

Изменения в управлении пользователями

В версии 2.0 реализована новая архитектура управления пользователями:

- В версии 1.2 пользователи управлялись напрямую через API контроллера – `/api/v2/users/`.
- В версии 2.0 все пользователи управляются через шлюз – `/api/gateway/v1/users/`.

Основные принципы новой модели:

- RBAC реализован централизованно на уровне шлюза.
- Аутентификация и авторизация отделены от исполнения заданий.
- Роли из 1.2 не совпадают напрямую с реализацией 2.0.
- Automation Controller, Private Automation Hub и Event-Driven Automation используют единое пространство для аутентификации пользователей.

Что перестанет и что продолжит работать

Для поддержания обратной совместимости в версии 2.0 допускается прямой доступ к компонентам платформы (например, к узлам контроллера). Однако это не обеспечивает полной обратной совместимости.

Сценарий	Статус	Примечание
Сценарии, использующие точки доступа API `/api/v2/users/` и `/api/v2/role_assignments/`	Не поддерживаются	См. *API, требующие миграции на шлюз*
Интеграции с HR-системами, создающие пользователей через API контроллера	Не поддерживаются	См. *API, требующие миграции на шлюз*
Рабочие процессы CI/CD, управляющие пользователями через контроллер	Не поддерживаются	Миграция на шлюз
Запуск шаблонов заданий через API контроллера	Работает	Точка доступа API `/api/v2/job_templates/{id}/launch/` остается доступной
Работа с проектами, инвентарями и полномочиями	Работает	API контроллера временно сохраняет совместимость
Получение истории выполнения заданий через API	Работает	API контроллера временно сохраняет совместимость
Работа с коллекциям Private Automation Hub через его API	Работает	API Private Automation Hub временно сохраняет совместимость
SSO и LDAP	Требуется повторная настройка	Настройки провайдеров аутентификации не переносятся на шлюз автоматически

API, требующие миграции на шлюз

В Astra Automation 2.0 изменены точки доступа API для управления пользователями и ролями. Все вызовы, ранее выполнявшиеся к API контроллера, должны быть переведены на API шлюза.

Старый путь (1.2)	Новый путь (2.0)
`/api/v2/users/`	`/api/gateway/v1/users/`
`/api/v2/teams/`	`/api/gateway/v1/teams/`
`/api/v2/organizations/`	`/api/gateway/v1/organizations/`
`/api/v2/role_assignments/`	`/api/gateway/v1/role_assignments/`
`/api/v2/settings/authentication/`	`/api/gateway/v1/authenticators/`

Подготовка инфраструктуры

На этой стадии происходит подготовка инфраструктуры для миграции платформы на версию 2.0.

Перед началом миграции необходимо самостоятельно подготовить инфраструктуру, на которой будет выполняться новая установка Astra Automation версии 2.0. Требования к ресурсам на целевых узлах должны быть не менее чем на исходных узлах и не менее того, что требуется для версии 2.

Необходимая информация об узлах представлена в *инструкции по подготовке узлов*.

Создание копии платформы версии 1.2

На этой стадии происходит создание копии уже имеющейся платформы на новых узлах.

Перед началом обновления необходимо создать резервную копию текущей платформы Astra Automation 1.2. Это позволит перенести данные на новые узлы и при необходимости выполнить откат без потери текущих настроек.

Копия включает все основные компоненты платформы:

- базу данных PostgreSQL;

- параметры Automation Controller и Private Automation Hub;
- параметры пользователей и ролей;
- проекты, инвентари, шаблоны и полномочия;
- параметры Event-Driven Automation;
- образы *EE*.

Создание резервной копии

Для создания резервной копии платформы выполните следующие действия:

1. Подключитесь к установочному узлу платформы версии 1.2 с правами администратора.
2. Перейдите в каталог установщика:

```
cd /opt/rbta/aa/astra-automation-setup/
```

3. Выполните команду резервного копирования:

```
sudo ./aa-setup -b -- --extra-vars backup_dest=/tmp/
```

Если инвентарь хранится в файле формата YAML, укажите путь к нему в значении аргумента `--inventory` (`-i`):

```
sudo ./aa-setup -b --inventory=/path/to/inventory.yml -- --extra-vars backup_dest=/
↪tmp/
```

4. Скопируйте архив на новый установочный узел, где будет выполняться обновление до версии 2.0:

```
scp /tmp/<backup_filename> <user>@<host>:/tmp/
```

где:

- <backup_filename> – название файла резервной копии, например: `astra-automation-backup-26-11-2025-080615.tar.gz`.
- <user> – название учетной записи пользователя целевого установочного узла;
- <host> – FQDN или IP-адрес установочного узла целевого кластера.

Восстановление из резервной копии

Для восстановления платформы из резервной копии подключитесь к целевому установочному узлу и выполните следующие действия:

1. Конфигурационный файл `/etc/apt/sources.list.d/astra-automation.list` приведите к одному из следующих видов в зависимости от используемой версии Astra Linux Special Edition:

1.7

```
deb https://dl.astralinux.ru/aa/aa-debs-for-alse-1.7 1.2 main
```

1.8

```
deb https://dl.astralinux.ru/aa/aa-debs-for-alse-1.8 1.2 main
```

2. Обновите список доступных пакетов:

```
sudo apt update
```

3. Установите пакет `astra-automation-setup`:

```
sudo apt install astra-automation-setup --yes
```

4. Перейдите в каталог `/opt/rbta/aa/astra-automation-setup/`:

```
cd /opt/rbta/aa/astra-automation-setup/
```

5. Запустите восстановление платформы из резервной копии:

```
sudo aa-setup --restore /tmp/<backup_filename>
```

Если инвентарь хранится в файле формата YAML, укажите путь к нему в значении аргумента `--inventory` (`-i`):

```
sudo ./aa-setup --restore /tmp/<backup_filename> --inventory=/path/to/inventory.yml
```

Проверка работы сервисов

Убедитесь, что ключевые компоненты успешно восстановились и работают. Для этого выполните следующие команды в терминале узла, который имеет сетевой доступ к только что созданному кластеру:

```
curl -k https://controller.aa.example.com/api/v2/ping/
```

```
curl -k https://hub.aa.example.com/api/v2/_ui/v1/status/
```

Действия перед обновлением

На этой стадии происходит сохранение данных платформы, которые не могут быть перенесены на новую версию Astra Automation автоматически. Позже эти данные потребуется восстановить на обновленной платформе.

Перед запуском процесса обновления Astra Automation до версии 2.0 необходимо выполнить ряд подготовительных шагов:

- *Подготовка PostgreSQL*.
- *Подготовка Event-Driven Automation*.
- *Экспорт ролевых привязок (RBAC)*.
- *Актуализация инвентаря*.

Эти шаги обеспечивают совместимость компонентов, предотвращают потерю данных и гарантируют корректное восстановление сервисов после миграции.

Подготовка PostgreSQL

Astra Automation версии 2.0 поддерживает только PostgreSQL 15. Если текущая версия платформы использует PostgreSQL версии ниже указанной, перед началом обновления ее необходимо обновить.

- Для встроенной СУБД утилита `aa-setup` выполнит обновление автоматически.
- Для внешней СУБД необходимо выполнить обновление вручную, согласно инструкции PostgreSQL.

Для проверки текущей версии СУБД выполните команду:

```
psql -U <pg_user> -h <pg_host> -p <pg_port> -d awx -c "SELECT version();"
```

Здесь:

- <pg_user> – название учетной записи администратора СУБД.
- <pg_host> – IP-адрес или *FQDN* СУБД;
- <pg_port> – порт для подключения к СУБД.

Подготовка Event-Driven Automation

База данных контроллера Event-Driven Automation версии платформы 1.2 не совместима с версией 2.0. Необходимо сохранить конфигурацию, удалить старую базу данных и создать новую.

Сохранение конфигурации

Перед удалением базы данных необходимо сохранить текущие объекты Event-Driven Automation: своды правил, их активации и источники событий. Эти данные не будут перенесены автоматически и потребуют ручного восстановления после обновления.

1. Создайте каталог для хранения резервных данных:

```
mkdir -p /tmp/eda_backup/
```

2. Сохраните своды правил:

```
curl -sk -u admin:<password> https://eda.example.com/api/eda/v1/rulebooks/ > /tmp/
↪eda_backup/rulebooks.json
```

3. Сохраните активации сводов правил:

```
curl -sk -u admin:<password> https://eda.example.com/api/eda/v1/activations/ > /tmp/
↪eda_backup/activations.json
```

4. Сохраните источники событий:

```
curl -sk -u admin:<password> https://eda.example.com/api/eda/v1/event-streams/ > /
↪tmp/eda_backup/event_sources.json
```

Остановка активаций и удаление БД

Старая база данных должна быть удалена, для этого выполните следующие шаги:

1. Остановите все активации сводов правил:
   - Через графический интерфейс:
     1. Перейдите в графический интерфейс Event-Driven Automation.
     2. Выберите на панели навигации *Активации сводов правил* (*Rulebook Activations*).
     3. В строке каждой активации переведите переключатель активности свода правил в положение **Выключено**.
   - Через API:

     Остановите все активации сводов правил, используя следующий цикл:

```
curl -sk -u admin:<password> https://eda.example.com/api/eda/v1/activations/ | \
  jq -r '.results[].id' | while read id; do
    curl -sk -u admin:<password> -X POST \
      https://eda.example.com/api/eda/v1/activations/$id/disable/
  done
```

2. На узле с развёрнутой БД выполните следующие команды для удаления БД:

```
sudo su postgres
psql -p <automationedacontroller_pg_port>
DROP DATABASE <automationedacontroller_pg_database>;
\q
```

   Здесь:

   - <automationedacontroller_pg_port> – порт для подключения к СУБД;

- <automationedacontroller_pg_database> – название БД.

3. Только для топологии **уровня предприятия с внешней СУБД** создайте пустую БД для Event-Driven Automation после ее удаления:

```
CREATE DATABASE <automationedacontroller_pg_database> WITH OWNER
↪<automationedacontroller_pg_username>;
```

Здесь:

- <automationedacontroller_pg_database> – название новой БД;
- <automationedacontroller_pg_username> – название учетной записи администратора СУБД.

Подготовка нового узла

Для подготовки нового узла СУБД скопируйте файл `/etc/astra_automation.version` с существующего узла Event-Driven Automation на новый в каталог `/etc/`:

```
scp /etc/astra_automation.version <user>@<host>:/etc/
```

где:

- <user> – название учетной записи пользователя целевого узла Event-Driven Automation;
- <host> – FQDN или IP-адрес целевого узла Event-Driven Automation.

Экспорт ролевых привязок (RBAC)

Перед миграцией необходимо экспортировать следующие настройки из текущей платформы:

- назначения ролей, их определения и контекст;
- составы команд.

Это требуется для восстановления данных вручную после миграции.

Экспорт ролей

Для экспорта ролей, их определения и контекста выполните следующие шаги:

1. Создайте каталог для хранения резервных данных:

```
mkdir -p /tmp/rbac_backup/
```

2. Сохраните назначения ролей:

```
curl -sk -u admin:<password> https://controller/api/v2/role_user_assignments/ > /
↪tmp/rbac_backup/role_user_assignments.json
```

```
curl -sk -u admin:<password> https://controller/api/v2/role_team_assignments/ > /
↪tmp/rbac_backup/role_team_assignments.json
```

3. Сохраните определения ролей:

```
curl -sk -u admin:<password> https://controller/api/v2/role_definitions/ > /tmp/
↪rbac_backup/role_definitions.json
```

4. Сохраните контекст:

```
curl -sk -u admin:<password> https://controller/api/v2/organizations/ > /tmp/rbac_
↪backup/organizations.json
```

```
curl -sk -u admin:<password> https://controller/api/v2/teams/ > /tmp/rbac_backup/
↪teams.json
```

```
curl -sk -u admin:<password> https://controller/api/v2/users/ > /tmp/rbac_backup/
↪users.json
```

Экспорт состава команд пользователей

При миграции команды создаются пустыми, члены команд и роли команд не мигрируют. Для их экспорта выполните следующие шаги:

1. Установите утилиту `jq`:

```
sudo apt install jq
```

2. Получите список всех команд:

```
curl -sk -u admin:<password> https://controller.example.com/api/v2/teams/ | \
  jq -r '.results[] | .id' > /tmp/team_ids.txt
```

3. Экспортируйте членов каждой команды:

```
while read team_id; do
  echo "Экспорт Team ID: $team_id"

  # Информация о команде
  curl -sk -u admin:<password> \
    "https://controller.example.com/api/v2/teams/$team_id/" \
    > "/tmp/rbac_backup/team_${team_id}_info.json"

  # Члены команды
  curl -sk -u admin:<password> \
    "https://controller.example.com/api/v2/teams/$team_id/users/" \
    > "/tmp/rbac_backup/team_${team_id}_users.json"

  # Роли команды
  curl -sk -u admin:<password> \
    "https://controller.example.com/api/v2/teams/$team_id/roles/" \
    > "/tmp/rbac_backup/team_${team_id}_roles.json"
done < /tmp/team_ids.txt
```

Проверка

Проверьте экспортированные данные с помощью следующих команд:

```
echo "=== Статистика RBAC ==="
echo "Назначения пользователям: $(jq '.count' /tmp/rbac_backup/role_user_assignments.
↪json)"
echo "Назначения командам: $(jq '.count' /tmp/rbac_backup/role_team_assignments.json)"
echo "Команд: $(jq '.count' /tmp/rbac_backup/teams.json)"

echo -e "\n=== Экспорт команд ==="
ls -1 /tmp/rbac_backup/team_*_users.json | wc -l
echo "команд экспортировано"
```

Актуализация инвентаря

Перед запуском обновления убедитесь, что файл инвентаря содержит актуальные сведения об узлах и параметрах подключения. Примеры файлов инвентаря для различных топологий представлены в *описании топологий*.

Обновление

На этой стадии происходит непосредственно обновление платформы.

Обновление Astra Automation выполняется с помощью утилиты `aa-setup`. Перед началом убедитесь, что все *подготовительные шаги* завершены:

- Развернута внешняя СУБД PostgreSQL 15 или подготовлен узел для автоматического обновления БД средствами платформы.
- Event-Driven Automation: конфигурация сохранена, активации правил остановлены, БД удалена, файл `/etc/astra_automation.version` скопирован с существующего узла на новый.
- RBAC: назначения ролей (`role_user_assignments`), назначения команд (`role_team_assignments`) и их определения (`role_definitions`) экспортированы.
- Резервная копия всей системы создана и протестирована.
- Определено окно обслуживания.
- Подготовлен план отката.

Для обновления платформы выполните следующие шаги:

1. Конфигурационный файл `/etc/apt/sources.list.d/astra-automation.list` приведите к одному из следующих видов в зависимости от используемой версии Astra Linux Special Edition:

   1.7

   ```
   deb https://dl.astralinux.ru/aa/aa-debs-for-alse-1.7 2.0 main
   ```

   1.8

   ```
   deb https://dl.astralinux.ru/aa/aa-debs-for-alse-1.8 2.0 main
   ```

2. Обновите список доступных пакетов:

   ```
   sudo apt update
   ```

3. Обновите пакет `astra-automation-setup`:

   ```
   sudo apt install astra-automation-setup --yes
   ```

4. Перейдите в каталог `/opt/rbta/aa/astra-automation-setup/`:

   ```
   cd /opt/rbta/aa/astra-automation-setup/
   ```

5. Запустите обновление, выполнив одну из следующих команд в зависимости от используемой версии Astra Linux Special Edition:

   1.7

   ```
   sudo aa-setup --upgrade \
     --repo-url https://artifactory.astralinux.ru/artifactory/aa-debs-for-alse-1.7 \
     --product-version 2.0 \
     -- -vv
   ```

   1.8

   ```
   sudo aa-setup --upgrade \
     --repo-url https://artifactory.astralinux.ru/artifactory/aa-debs-for-alse-1.8 \
     --product-version 2.0 \
     -- -vv
   ```

Внимание

Указание аргументов *--repo-url* и *--product-version* является обязательным.

6. После окончания обновления на всех узлах проверьте версию Astra Automation:

```
ansible all -i inventory -m shell -a "cat /etc/astra-automation/version"
```

Ожидаемый вывод: `2.0`.

Действия после обновления

На этой стадии происходит восстановление данных, которые не могли быть автоматически перенесены и были сохранены на этапе *Действия перед обновлением*.

Проверка миграции методов аутентификации

После обновления до версии 2.0 все методы аутентификации из версии 1.2 автоматически мигрируют в шлюз как *унаследованные* (legacy) методы (доступны только в для чтения). Проверьте их наличие:

1. Экспортируйте все методы аутентификации

```
curl -sk -u admin:<password> https://gateway.com/api/gateway/v1/authenticators/ > /
↪tmp/authenticators_after_upgrade.json
```

2. Проверьте активность методов:

```
jq '.results[] | {id, name, type, enabled}' /tmp/authenticators_after_upgrade.json
```

3. Изучите список унаследованных методов:

```
jq '.results[] | select(.type | contains("legacy")) | {name, type}' /tmp/
↪authenticators_after_upgrade.json
```

Унаследованные методы создаются автоматически с префиксом `legacy_`.

Настройка новых методов аутентификации описана в следующих разделах.

Связывание учетных записей

В версии 2.0 шлюз является единой точкой аутентификации. Пользователи больше не подключаются напрямую к Automation Controller, Private Automation Hub или Event-Driven Automation. Для сохранения доступа к их прежним объектам (проектам, инвентарям, заданиям) требуется связать старые учетные записи с новыми. Для этого выполните следующие действия:

1. Перейдите в веб-интерфейс шлюза.
2. В окне аутентификации выберите один из предложенных вариантов:
   - **I have an automation controller account** – выберите этот вариант, если у вас есть учётные данные пользователя Automation Controller.
   - **I have an automation hub account** – выберите этот вариант, если у вас есть учётные данные пользователя Private Automation Hub.
3. Введите учётные данные пользователя выбранного на предыдущем шаге компонента Astra Automation и нажмите кнопку *Log in*.
4. После связывания учетных записей на экране отобразится информация с названием учетной записи текущего пользователя.
5. Если у вас есть учётная запись второго компонента Astra Automation (учётная запись Private Automation Hub, если ранее был выбран Automation Controller, и наоборот), можно также связать ее в открытом окне **Link your |AA| accounts**.

6. Нажмите кнопку *Submit*.

Для проверка статуса связывания:

1. На панели навигации выберите *Access Management* ▸ *Users*.
2. Выберите пользователя.
3. Проверьте значение поля **Last login**.

   В значении поля должна быть указана дата и время входа в текущий сеанс.

Управление ролями

После связывания учетных записей выполняется автоматическая миграция ролевых назначений (*RBAC*) на шлюз. Однако из-за изменений в модели RBAC может потребоваться ручная корректировка прав доступа.

Проверка наличия системного администратора

В платформе Astra Automation должен быть системный администратор. Для проверки его наличия выполните следующие действия:

1. Перейдите в веб-интерфейс шлюза.
2. Выберите на панели навигации *Access Management* ▸ *Users.*
3. Найдите учетную запись, у которой в значении поля столбца **User type** будет **System administrator**.

Если системный администратор отсутствует, создайте его с помощью команды, подключившись по SSH к узлу шлюза:

```
awx-manage create_superuser --username=admin
```

Проверка команд и назначение ролей в организациях

Проверьте, что роли мигрировали и назначены на команды внутри организаций:

1. Выберите на панели навигации *Access Management* ▸ *Organizations* ▸ *<Organization>* ▸ *Users*.
2. Перейдите во вкладку **Astra Automation** и проверьте наличие всех пользователей.

   Если пользователя нет, добавьте их, используя кнопку *Add users*.
3. Выберите на панели навигации *Access Management* ▸ *Organizations* ▸ *<Organization>* ▸ *Teams* и проверьте список команд организации.
4. Выберите на панели навигации *Access Management* ▸ *Teams* ▸ *<Team>* ▸ *Roles* и проверьте какие роли назначены команде.
5. Выберите на панели навигации *Access Management* ▸ *Teams* ▸ *<Team>* ▸ *Users* и проверьте состав команды.

Проверка прямых ролей пользователей

Проверьте, что роли, назначенные на пользователей, мигрировали корректно. Для этого выберите на панели навигации *Access Management* ▸ *Users* ▸ *<User>* ▸ *Roles* и проверьте установленные роли.

Внимание

При миграции с Astra Automation версии 1.2 на версию 2.0 не все пользователи могут быть перенесены в организации для Astra Automation.

Для решения ситуации:

1. Выберите на панели навигации *Organizations* ▸ *<Organization>* ▸ *Users* и перейдите во вкладку **Astra Automation**.
2. Нажмите кнопку *Add users* и выберите пользователей из списка.
3. Проверьте, что пользователи добавлены.

Восстановление команд и их ролей

После миграции команды создаются пустыми. Необходимо вручную восстановить требуемые связи:

- члены команд;
- роли, назначенные на команды.

Восстановление через графический интерфейс (рекомендуется)

Используйте графический интерфейс для восстановления пользователей и ролей:

1. Выберите на панели навигации *Access Management* ▸ *Teams -> <Team>* ▸ *Users*.
2. Нажмите кнопку *Add users*.
3. Добавьте пользователей из `/tmp/rbac_backup/team_*_users.json`.
4. Выберите конкретный ресурс в разделах **Projects**/**Templates**/**Inventories**.
5. Перейдите во вкладку **User Access**, выберите команду и добавьте роли с помощью кнопки *Add roles*.

Пример восстановления через API

Для восстановления пользователей и команд пользователей через API воспользуйтесь следующими вызовами HTTP:

```
# Получить ID команды в новой системе
TEAM_NAME="Operators"
TEAM_ID=$(curl -sk -u admin:<password> \
  "https://gateway.com/api/gateway/v1/teams/?name=$TEAM_NAME" | jq -r '.results[0].id')

# Получить список пользователей из бэкапа (замените 7 на ваш team_id из 1.2)
OLD_TEAM_ID=7
jq -r '.results[].username' /tmp/rbac_backup/team_${OLD_TEAM_ID}_users.json | \
while read username; do
  echo "Добавление $username в команду $TEAM_NAME"

  USER_ID=$(curl -sk -u admin:<password> \
    "https://gateway.com/api/gateway/v1/users/?username=$username" | jq -r '.results[0].
↪id')

  curl -sk -u admin:<password> -X POST \
    -H "Content-Type: application/json" \
    "https://gateway.com/api/gateway/v1/teams/$TEAM_ID/users/" \
    -d "{\"id\": $USER_ID}"
done
```

Восстановление ролей команды

Список 1: Пример запроса для JSON, который содержит выгрузку из legacy API

```
jq '.results[] | {
  role: .name,
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  resource_type: .summary_fields.resource_type,
  resource_name: .summary_fields.resource_name,
  resource_id: .summary_fields.resource_id
}' /tmp/rbac_backup/team_${OLD_TEAM_ID}_roles.json
```

Список 2: Пример запроса для JSON, который содержит выгрузку из нового API

```
jq '.results[] | {
  role: .summary_fields.role_definition.name,
  object: .summary_fields.content_object.name
}' /tmp/rbac_backup/team_${OLD_TEAM_ID}_roles.json
```

Примечание

Формат JSON различается в унаследованном (legacy) и новом API:

- Унаследованный формат (Astra Automation 1.2):
  - Роль в `.name`.
  - Объект в `.summary_fields.resource_name`.
- Новый формат (Astra Automation 2.0):
  - Роль в `.summary_fields.role_definition.name`.
  - Объект в `.summary_fields.content_object.name`.

Для назначения роли команде на ресурсы Automation Controller выполните следующие действия:

1. Выберите на панели навигации *Automation Execution* ▸ *Projects/Templates/Inventories*
2. Выбрать нужный ресурс и перейдите во вкладку **Team Access**.
3. Нажмите кнопку *Add*.
4. Выберите необходимую команду и добавьте ей роли.

Примечание

Если используется Private Automation Hub с мигрированными namespaces, роли назначаются в разделе *Automation Content* ▸ *Namespaces* ▸ *Permissions*.

Настройка внешней аутентификации

При создании нового метода SAML или LDAP обязательно включите автоматическую миграцию. Для этого выполните следующие действия:

1. Выберите на панели навигации *Access Management* ▸ *Authentication Methods*.
2. Нажмите кнопку *Create authentication*.
3. Перейдите во вкладку **Authentication details** и выберите все унаследованные методы в поле **Auto migrate users from**:

- `controller: legacy_password`;
- `hub: legacy_password`;
- `eda: legacy_password`;
- `legacy_sso-saml-*` (если эти методы были в Astra Automation версии 1.2).

Это предотвратит создание дублей пользователей при первом входе.

Настройка SAML

Предварительные требования:

- настроенный IdP;
- ключи и сертификат SP (saml.crt, saml.key);
- данные от IdP: Entity ID, SSO URL, публичный сертификат.

Для настройки SAML на сервере KeyCloak выполните следующие действия:

1. Перейдите в Keycloak.
2. Выберите необходимый Realm.
3. Создайте клиента SAML (*Create client* ▸ *SAML*), указав для поля **Client ID** в качестве значения URL-адрес шлюза, например: `https://gateway.com/`.
4. Настройте клиента SAML в соответствии со следующими данными:
   - General Settings
     - **Client ID**: https://gateway.com/
     - **Valid redirect URIs**: https://gateway.com/*
     - **Force POST binding**: ON
     - **Include AuthnStatement**: ON
     - **Sign documents**: ON
   - Keys
     - **Signing keys config**: OFF
     - **Encryption keys config**: ON
     - **Encrypt assertions**: ON

     После включения нажмите кнопку *Generate* и сохраните `private.key`.
   - Advanced Settings
     - **Logout Service POST Binding URL**: https://gateway.com/#/login
     - **Logout Service Redirect Binding URL**: https://gateway.com/#/login
5. Получите сертификат IdP, для этого:
   1. Выберите на панели навигации *Realm Settings* ▸ *Endpoints* ▸ *SAML 2.0 Identity Provider Metadata*.
   2. Скопируйте `<ds:X509Certificate>`.

Настройте SAML на шлюзе Astra Automation, для этого:

1. Выберите на панели навигации шлюза *Access Management* ▸ *Authentication Methods* ▸ *Create authentication*.
2. Заполните форму следующими данными:
   - **Authentication type**: SAML
   - **Name**: Keycloak SAML
   - **Auto migrate users from**: выберите все `legacy_password`
   - **SP Entity ID**: http://gateway.com/
   - **SP Public Certificate**: содержимое `saml.crt`
   - **SP Private Key**: содержимое `saml.key`
   - **IdP Login URL**: http://keycloak.com:8080/realms/master/protocol/saml
   - **IdP Public Cert**: из Keycloak

- **Entity ID**: http://keycloak.com:8080/realms/master

Маппинг атрибутов:

- **User Email**: email
- **Username**: username
- **User Last Name**: last_name
- **User First Name**: first_name
- **User Permanent ID**: email

3. Сохраните настройки SAML.
4. Скопируйте **ACS URL**.
5. Добавьте **ACS URL** в **Valid redirect URIs** в Keycloak.
6. Включить метод (Enabled = ON).

Проверьте корректность настройки SAML, для этого:

1. Выйдите из Keycloak.
2. Откройте браузер в режиме инкогнито.
3. Перейдите в веб-интерфейс шлюза.
4. Нажмите кнопку *SSO*.
5. Авторизуйтесь в Keycloak.
6. Вернитесь в шлюз.

Настройка OIDC

Для настройки OIDC выполните следующие действия на сервере KeyCloak:

1. Перейдите в Keycloak.
2. Выберите необходимый Realm.
3. Создайте клиента OIDC (*Create client* ▸ *OIDC*), указав:
   - для поля **Client type** значение `OpenID Connect`.
   - для поля **Client ID** значение `aa-gateway-oidc`.
4. Настройте клиента OIDC в соответствии со следующими данными:
   - Capability Config
     - **Client authentication**: ON
     - **Authorization**: OFF
     - **Standard flow**: ON
     - **Direct access grants**: ON
   - Login Settings
     - **Root URL**: https://gateway.com
     - **Home URL**: https://gateway.com
     - **Valid redirect URIs**: https://gateway.com/, *http://gateway.com/*
     - **Valid post logout redirect URIs**: https://gateway.com/, *http://gateway.com/*
     - **Web origins**: https://gateway.com, http://gateway.com, *

> **Внимание**
>
> Важно указывать значения для этого поля без пробелов.

5. Получите **Client Secret**, выбрав на панели навигации *Clients* ▸ *Credentials* ▸ *Client Secret*.
6. Настройте **Mappers** в соответствии со следующими данными:

   Для scope `[aa-gateway-oidc]-dedicated`:

Claim	Property	Token Claim Name
username	username	preferred_username
email	email	email
first name	firstName	given_name
last name	lastName	family_name

Настройте OIDC на шлюзе Astra Automation, для этого:

1. Выберите на панели навигации шлюза *Access Management* ▸ *Authentication Methods* ▸ *Create authentication*.
2. Заполните форму следующими данными:
   - **Authentication type**: OIDC
   - **Name**: Keycloak OIDC
   - **Auto migrate users from**: выбрать все `legacy_password`
   - **OIDC Provider URL**: http://keycloak.com:8080/realms/master
   - **Client ID**: aa-gateway-oidc
   - **Client Secret**: полученный секрет
   - **JWT Algorithms**: RS256
   - **Username Key**: preferred_username
   - **ID Key**: sub
   - **Create objects**: ON
   - **Enabled**: ON

Проверьте корректность настройки OIDC, для этого:

1. Выйдите из Keycloak.
2. Откройте браузер в режиме инкогнито.
3. Перейдите в веб-интерфейс шлюза.
4. Нажмите кнопку *OIDC*.
5. Пройдите авторизацию.
6. Вернитесь в шлюз.

Возможные проблемы OIDC

Ошибка	Возможная причина	Решение
Ошибка 500 при нажатии кнопки «мыши»	Используется `/.well-known`	Убрать этот суффикс
`/auth_failed`	Нет RS256	Добавить
`/auth_failed`	Нет **claims**	Настроить **mappers**
`/auth_failed`	Пробелы в значении поля **Web origins**	Удалить пробелы

Группы пользователей OIDC

Для настроек групп пользователей OIDC выполните следующие действия:

1. Перейдите в Keycloak.
2. На панели навигации выберите *Groups*.
3. Нажмите кнопку *Create group* и создайте группу с названием `aa-admins`.
4. На панели навигации выберите *Clients* ▸ *[aa-gateway-oidc]* ▸ *Mappers*.
5. Нажмите кнопку *Add mapper* и добавьте маппер типа `Group Membership` со следующими параметрами:
   - **Token Claim Name**: groups
   - **Full path**: OFF
   - **Add to all tokens**: ON
6. Перейдите в веб-интерфейс шлюза Astra Automation.
7. В методе OIDC установите для параметра **Groups Claim** значение `groups`.
8. На панели навигации выберите *Teams*.
9. Нажмите кнопку *Create team* и создайте команду с названием `aa-admins`.
10. Назначьте роли команде.

Группы пользователей SAML

Для настроек групп пользователей SAML выполните следующие действия:

1. Перейдите в Keycloak.
2. На панели навигации выберите *Client Scopes*.
3. Нажмите кнопку *Create* и создайте группу `group_list`.
4. Нажмите кнопку *Add mapper* и добавьте маппер типа `Group list` со следующими параметрами:
   - **Name**: memberOf
   - **Group attribute name**: memberOf
   - **Full group path**: OFF
5. Перейдите в веб-интерфейс шлюза Astra Automation.
6. Добавьте `scope` в клиент SAML шлюза.
7. На панели навигации выберите *Teams*.
8. Нажмите кнопку *Create team* и создайте команду с названием `aa-core-team`.
9. Назначьте роли команде.

Восстановление Event-Driven Automation

В Astra Automation версии 2.0 объекты Event-Driven Automation не переносятся автоматически. После обновления необходимо вручную восстановить все конфигурации.

Создание полномочий

Для продолжения работы с Event-Driven Automation заново создайте полномочия для Automation Controller и Git:

1. Получите токен аутентификации Controller с помощью следующего запроса:

```
curl -sk -X POST https://controller.example.com/api/v2/tokens/ \
  -u testadmin:<password> \
  -H "Content-Type: application/json" \
  -d '{"description":"EDA Integration","scope":"write"}' | jq -r '.token'
```

2. На панели навигации выберите *Infrastructure* ▸ *Credentials*.
3. Нажмите кнопку *Create credential* и создайте полномочие со следующими параметрами:
   - **Name**: Controller API Token
   - **Credential type**: Astra Automation Platform
   - **Host**: адрес контроллера
   - **OAuth Token**: полученный токен
   - **Verify SSL**
4. Нажмите кнопку *Create credential* и создайте полномочие типа **Source Control** со следующими параметрами:
   - **Name**: GitFlic SSH Key
   - **Username**
   - **SSH Private Key**

Создание среды принятия решений

Для продолжения работы с Event-Driven Automation заново создайте среду принятия решений:

1. На панели навигации выберите *Automation Decisions* ▸ *Decision Environments*.
2. Нажмите кнопку *Create* и создайте среду принятия решений.

Создание проектов

Для продолжения работы с Event-Driven Automation заново создайте проекты:

1. На панели навигации выберите *Automation Decisions* ▸ *Projects*.
2. Нажмите кнопку *Create* и создайте проект.

Создание потоков событий

Для продолжения работы с Event-Driven Automation заново создайте потоки событий:

1. На панели навигации выберите *Automation Decisions* ▸ *Event Streams*.
2. Нажмите кнопку *Create* и создайте поток событий.
3. Скопируйте **Event Stream URL**.
4. Настройте внешний источник.

Создание активации сводов правил

Для продолжения работы с Event-Driven Automation заново создайте активации сводов правил:

1. На панели навигации выберите *Automation Decisions* ▸ *Rulebook Activations*.
2. Нажмите кнопку *Create* и создайте активации сводов правил.

Проверка и замена URL в шаблонах и интеграциях

После обновления до версии 2.0 изменяется архитектура сетевого доступа: все сервисы теперь доступны через шлюз, а прямой доступ к балансировщикам (`ac.example.com`, `hub.example.com`) может быть ограничен.

Необходимо проверить и обновить все URL (*hardcoded FQDN*) в шаблонах, webhook, интеграциях и внешних системах.

Матрица замены адресов

Проверьте все настроенные для версии платформы 1.2 интеграции, сценарии и конфигурации, в которых используются жестко прописанные URL. Примеры возможных мест использования:

- Extra vars templates;
- динамические инвентари;
- webhooks GitHub/GitLab;
- внешние системы: Zabbix, Prometheus, Jenkins, GitLab CI, ServiceNow;
- скрипты (curl, requests, Ansible uri).

Выполните замену старых адресов на новые в соответствии со следующей таблицей:

Компонент	Пример старого адреса	Новый адрес	Назначение
API Automation Controller	`https://controller.aa.example.com/api/v2/`	`https://aa.example.com/api/controller/v2/`	Доступ через API шлюза
UI Automation Controller	`https://controller.aa.example.com/`	`https://aa.example.com/` (адрес шлюза)	Все компоненты доступны в едином графическом интерфейсе
API Private Automation Hub	`https://hub.aa.example.com/api/galaxy/`	`https://aa.example.com/api/galaxy/`	Доступ через API шлюза
UI Private Automation Hub	`https://hub.aa.example.com/`	`https://aa.example.com/` (адрес шлюза)	Все компоненты доступны в едином графическом интерфейсе
API Event-Driven Automation	`https://eda.aa.example.com/api/eda/v1/`	`https://aa.example.com/api/eda/v1/`	Доступ через API шлюза
UI Event-Driven Automation	`https://eda.aa.example.com/`	`https://aa.example.com/` (адрес шлюза)	Все компоненты доступны в едином графическом интерфейсе
API шлюза	(отсутствует)	`https://aa.example.com/api/gateway/v1/`	API шлюза
UI шлюза	(отсутствует)	`https://aa.example.com/`	Единая точка входа для всех сервисов

Примечание

Остальные точки доступа к API Automation Controller и API Private Automation Hub **временно** сохраняются для обратной совместимости с интеграциями и сценариями версии 1.2. Они будут отключены в будущих релизах.

Рекомендуется планировать постепенный перевод всех интеграций на API шлюза.

Проверка после обновления

На этой стадии происходит проверка работоспособности платформы после обновления на версию 2.0.

После завершения обновления и ручной настройки необходимо убедиться, что все сервисы работают корректно.

Проверка сервисов

Проверьте состояние всех компонентов Astra Automation 2.0:

```
sudo ./aa-setup --status
```

Ожидаемый результат: все сервисы в состоянии `running` или `healthy`.

Также можно выполнить проверку различных компонентов платформы через API шлюза:

```
curl -sk https://aa.example.com/api/gateway/v1/ping/
curl -sk https://aa.example.com/api/controller/v2/ping/
curl -sk https://ac.example.com/api/v2/ping/
curl -sk https://hub.example.com/api/galaxy/v3/plugin/ansible/content/published/
↪collections/
curl -sk https://aa.example.com/api/eda/v1/activations/
```

Проверка авторизации

Убедитесь, что пользователи могут авторизоваться на платформе:

```
TOKEN=$(curl -sk -X POST https://aa.example.com/api/gateway/v1/tokens/ \
  -H "Content-Type: application/json" \
  -d '{"username":"admin","password":"<password>"}' | jq -r '.token')

curl -sk -H "Authorization: Bearer ${TOKEN}" https://aa.example.com/api/controller/v2/me/
curl -sk -H "Authorization: Bearer ${TOKEN}" https://aa.example.com/api/gateway/v1/role_
↪assignments/
```

Функциональные тесты

Выполните несколько функциональных тестов для проверки работы платформы с помощью следующих команд на своей рабочей станции или любом узле, имеющем сетевой доступ к установленной платформе:

- Запуск шаблона задания:

  ```
  JOB_ID=$(curl -sk -X POST -H "Authorization: Bearer ${TOKEN}" \
    https://ac.example.com/api/v2/job_templates/1/launch/ | jq -r '.id')
  ```

- Мониторинг выполнения:

  ```
  watch -n 2 "curl -sk -H 'Authorization: Bearer ${TOKEN}' \
    https://ac.example.com/api/v2/jobs/${JOB_ID}/ | jq '{status, failed, elapsed}'"
  ```

- Последние 5 заданий:

```
curl -sk -H "Authorization: Bearer ${TOKEN}" \
  'https://ac.example.com/api/v2/jobs/?page_size=5&order_by=-id' | \
  jq '.results[] | {id, name, status, failed}'
```

Проверка сред исполнения

Проверьте среды исполнения:

- Список EE:

```
curl -sk -H "Authorization: Bearer ${TOKEN}" \
  https://ac.example.com/api/v2/execution_environments/ | jq '.results[] | {name,␣
↪image}'
```

- Проверка pull образа:

```
podman login hub.example.com -u <registry_user> -p <registry_pass>
podman pull hub.example.com/ee-minimal-rhel8:latest
```

ГЛАВА 7

Day 2: Операционные задачи

Astra Automation, как комплексная платформа, требует оперативного управления, постоянного мониторинга, технического обслуживания и оптимизации. Эти операции выполняют системные администраторы, операционные инженеры, инженеры DevOps и технической поддержки.

После успешного развертывания и настройки Astra Automation (Day 1) начинается критически важная фаза, охватывающая весь остальной жизненный цикл платформы с момента ее запуска и включающая управление, мониторинг, обслуживание, оптимизацию и эволюцию системы.

Операции направлены на обеспечение работоспособности и оптимальных характеристик платформы:

- **надёжности и доступности** платформы и ее сервисов;
- **оптимальной производительности** компонентов и заданий автоматизации;
- **безопасности** через мониторинг, обновления и соответствие политикам;
- **масштабируемости** для удовлетворения растущих потребностей организации;
- **соответствия нормативным требованиям и внутренним политикам**.

7.1 Администрирование

Администрирование платформы охватывает набор операционных задач, необходимых для обеспечения бесперебойной работы Astra Automation. В основном это осуществляется через графический интерфейс и включает следующие группы задач:

- Настройка платформы и ее компонентов:
  - управление подпиской на Astra Automation;
  - управление доступом к интерфейсу платформы для пользователей и приложений;
  - создание организаций, предоставление им ресурсов и делегирование прав;
  - настройка параметров компонентов;
  - оптимизация производительности;
  - интеграция с другими системами.
- Управление журналами:
  - настройка уровня регистрируемой информации;

  - архивация, ротация и удаление журналов;
  - централизованное хранение журналов во внешних системах хранения;
  - анализ журналов.

Для выполнения операций администрирования может также потребоваться API с административными привилегиями.

Для выполнения своих функций администратор должен обладать следующими знаниями и умением:

- понимание принципов работы платформы и ее компонентов;
- умение работать с графическим интерфейсом платформы;
- умение использовать API;
- обладать знаниями и навыками работы с операционной системой Astra Linux Special Edition;
- понимание сетевых концепций (DNS, маршрутизация, порты TCP, proxy).

7.1.1 Настройка системы

Общие настройки платформы направлены на обеспечение удобной и безопасной работы, интеграции с внешними сервисами, достижения оптимальной производительности и масштабируемости.

В отличие от настроек, специфичных для конкретных организаций, проектов или заданий, системные параметры (system settings) применяются на **уровне всей платформы** и доступны только администраторам.

Доступ к системным настройкам осуществляется через графическую консоль платформы в разделе *Настройки (Settings)* и не требует прямого редактирования конфигурационных файлов для большинства операций.

Лучшие практики настройки системы

Ознакомьтесь с рекомендациями по настройке системы, чтобы обеспечить максимальную безопасность и производительность платформы.

Документируйте изменения

Ведите журнал всех изменений системных параметров, отмечая следующие данные для каждого изменения:

- когда было сделано изменение;
- кто его сделал;
- какой параметр был изменён и с какого значения на какое;
- причина изменения.

Это поможет при устранении неполадок и откате изменений.

Проверяйте перед применением в рабочем окружении

Если возможно, сначала примените изменения в **тестовом окружении** и убедитесь, что они не вызывают нежелательных эффектов.

Назначайте версии для критичных параметров

Для параметров, которые часто меняются (например, списки прокси), рассмотрите систему версионирования или документирование истории изменений.

Синхронизируйте параметры между узлами

Если какой-либо компонент Astra Automation реплицирован на нескольких узлах, убедитесь, что системные параметры синхронизированы на всех узлах.

Регулярно проверяйте применённые параметры

Периодически (например, ежемесячно) просматривайте текущие системные параметры, чтобы убедиться, что они остаются актуальными и соответствуют политикам организации.

Будьте осторожны с параметрами безопасности

Параметры, влияющие на безопасность (CSRF Trusted Origins, прокси-список, управление журналами), требуют особого внимания. Неправильная настройка может привести к появлению уязвимости.

Типичные сценарии настройки

Для лучшего понимания процесса настройки, рассмотрите следующие сценарии:

Сценарий 1: Настройка подписки

При изменении параметров подписки выполните следующие действия:

1. Убедитесь, что подписка в Личном кабинете соответствует заданным параметрам.
2. Перейдите в **Настройки > Подписка** (Settings > Subscription).
3. Нажмите *Изменить подписку* (Edit subscription).
4. Нажмите **Название учетной записи / пароль** (Username / password).
5. В поле **Название учетной записи** (Username) введите название учетной записи, используемой для доступа в Личный кабинет.
6. В поле **Пароль** (Password) введите пароль для доступа в Личный кабинет.
7. В поле **Подписка** (Subscription) выберите подписку из выпадающего списка.
8. Нажмите *Далее* (Next).
9. Ознакомьтесь с лицензионным соглашением и нажмите кнопку *Далее* (Next).
10. В окне **Обзор** (Review) нажмите кнопку *Завершить* (Submit).

Система активирует подписку и автоматически отобразит в браузере информационную панель графической консоли. В Личном кабинете статус подписки изменится на **Использован**, а тип активации на **Онлайн**.

Сценарий 2: Сохранение журналов в централизованной системе

Настройте платформу на посылку журналов в централизованную систему:

1. Получите адрес и порт TCP сервера внешней системы.
2. Перейдите в **Настройки > Регистрация** (Settings > logging).
3. Нажмите *Редактировать* (Edit).
4. Заполните параметры внешней системы согласно *инструкции*.
5. С помощью параметра **Logging Aggregator Level Threshold** установите необходимый уровень информации, подлежащей регистрации.
6. Сохраните и убедитесь, что во внешней системе появились записи из журналов платформы.

Сценарий 3: Управление средой исполнения

Если вы хотите хотите добавить в платформу собственный Execution Environment (EE), выполните действия, описанные в *инструкции*.

Если вы хотите сменить EE, действующий по умолчанию для всех заданий, выполните следующие действия:

1. Получите адрес образа EE в реестре.
2. Перейдите в **Настройки > Система** (Settings > System) и нажмите *Редактировать* (Edit).
3. В поле **Global default execution environment** укажите адрес собственного образа.
4. Сохраните настройку.

Все новые задания будут по умолчанию использовать этот EE (существующие шаблоны не изменятся).

Для настройки EE по умолчанию для заданий от конкретной организации выполните следующие действия:

1. Перейдите в **Настройки > Организации** (Settings > Organizations) и выберите организацию.
2. Нажмите *Редактировать организацию* (Edit organization).
3. В поле **Execution environment** выберите один из образов, известный системе.
4. Сохраните настройку.

Все новые задания в пределах этой организации будут по умолчанию использовать указанный EE (существующие шаблоны не изменятся).

7.1.2 Использование журналов контроллера

Для хранения служебных сообщений в виде журналов в Automation Controller используется syslog-ng[44].

Общие сведения

`syslog-ng` позволяет передавать данные журналов в различные *приемники*: файлы на локальных и сетевых дисках, серверы журналирования, а также во внешние *агрегаторы журналов*.

Если сетевой приемник становится недоступным, `syslog-ng` сохраняет сообщения на локальном диске. При восстановлении соединения сообщения автоматически передаются из локальной копии в приемник в том же порядке, в котором они были получены.

При каждом изменении настроек журналирования происходит следующее:

- обновляется содержимое конфигурационного файла `/var/lib/awx/syslogng/syslog-ng.conf`;
- для применения новых настроек перезапускается служба `awx-syslogng`.

Совет

Настройки журналирования рекомендуется изменять через *графический интерфейс*.

`syslog-ng` поддерживает два режима дисковой буферизации:

- Надежная (reliable):
  - Сообщения из буфера удаляются только при получении от приемника подтверждения записи.

[44] https://github.com/syslog-ng/syslog-ng

  - Низкая производительность (по сравнению с нормальной дисковой буферизацией).
  - Обеспечивается сохранность сообщений при перезагрузке и перезапуске службы, потере подключения к приемнику или сбоя в работе самого `syslog-ng`.
- Обычная (normal):
  - Высокая производительность.
  - Возможна потеря сообщений в случае сбоя `syslog-ng`.

Если Automation Controller настроен на передачу журналов во внешний агрегатор, для `syslog-ng` используются настройки, реализующие следующее поведение:

- Исходящие сообщения помещаются непосредственно в очередь вывода, если она не заполнена.
- Если очередь вывода заполнена (количество сообщений в очереди достигло значения `mem-buf-length`), а параметру `reliable` присвоено значение `no`, сообщения помещаются в обычный дисковый буфер.

Если контроллер настроен на передачу журналов во внешний агрегатор с использованием HTTP/TCP/UDP, по умолчанию используются следующие настройки дисковой буферизации:

```
disk-buffer(
  dir("/var/lib/awx")
  disk-buf-size(1073741824)
  qout-size(1000)
  mem-buf-length(131072)
  reliable(no)
)
```

Некоторые из указанных настроек можно изменить только через API (точка доступа – `/api/v2/settings/logging/`):

- `dir` – полный путь к локальному каталогу для хранения журналов.

  Параметр API – `LOG_AGGREGATOR_MAX_DISK_USAGE_PATH`.
- `disk-buf-size` – размер дискового буфера в байтах.

  Параметр API – `LOG_AGGREGATOR_ACTION_MAX_DISK_USAGE_GB`.

  Важно

  При изменении через API значение этого параметра следует указывать в гигабайтах.
- `mem-buf-length` – максимальное количество сообщений в очереди.

  Параметр API – `LOG_AGGREGATOR_ACTION_QUEUE_SIZE`.

Агрегаторы

Automation Controller может быть настроен на передачу журналов в следующие внешние агрегаторы:

- Logstash[45];
- Splunk[46];
- Loggly[47];
- Sumo Logic[48].

[45] https://www.elastic.co/guide/en/logstash/current/index.html
[46] https://docs.splunk.com/Documentation
[47] https://documentation.solarwinds.com/en/success_center/loggly/content/loggly_documentation.htm
[48] https://help.sumologic.com/docs/get-started/

В агрегатор передаются только сообщения, важность которых выше или равна указанной (в порядке возрастания важности):

- **DEBUG** – все сообщения журнала;
- **INFO** – информационные сообщения;
- **WARNING** – предупреждения;
- **ERROR** – информация об ошибках;
- **CRITICAL** – информация о критических событиях.

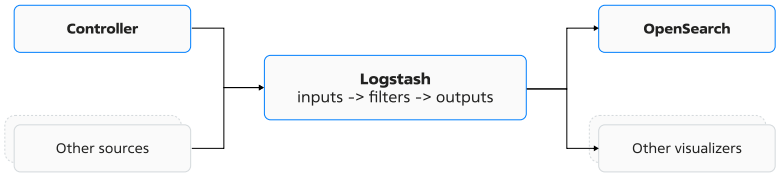
По умолчанию в агрегатор передаются записи, полученные от всех внутренних регистраторов Automation Controller:

- `awx` – общие журналы сервера;
- `activity_stream` – запись изменений объектов в Automation Controller;
- `job_events` – данные, возвращаемые модулем обратного вызова Ansible;
- `system_tracking` – данные о фактах, собранные модулем `setup`;
- `broadcast_websocket` – данные об отправленных сообщениях по WebSocket.

Обработка журналов с помощью Logstash и OpenSearch

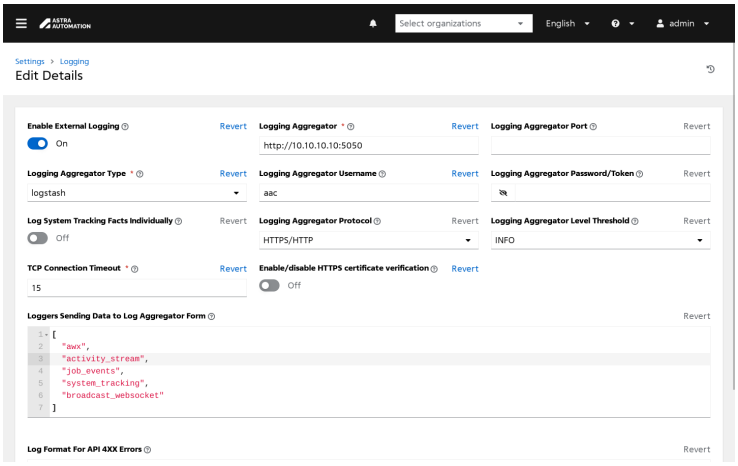
Программное обеспечение Logstash используется для хранения записей журналов различных форматов. Поддержка форматов реализуется расширениями. Для визуализации хранящихся данных могут быть использованы различные внешние системы, например, OpenSearch, Elasticsearch и так далее.

Logstash может одновременно обрабатывать данные, полученные от разных поставщиков.



Чтобы настроить Automation Controller на передачу журналов в систему OpenSearch через агрегатор Logstash, выполните следующие действия:

1. В *настройках журналирования* задайте значения следующих параметров:
   - **Агрегатор протоколирования**: `http://10.10.10.10:5050`.
   - **Тип агрегатора ведения журнала**: **logstash**.
   - **Имя пользователя агрегатора ведения журнала**: aac.
   - **Пароль/токен агрегатора ведения журнала**: пароль для подключения к Logstash.
   - **Протокол ведения журналов**: **HTTPS/HTTP**.
   - **Включить/отключить проверку сертификата HTTPS**: выкл.
   - **Порог уровня агрегатора ведения журнала**: **INFO**.
   - **Включить внешнее ведение журнала**: вкл.



2. Установите Logstash согласно инструкции[49].
3. Для работы с Automation Controller отредактируйте файл конфигурации агрегатора `logstash.conf`:

Logstash 7

```
input {
  http {
    port => <listen_port>
    user => <username>
    password => "<password>"
    codec => json {
      charset => "UTF-8"
    }
  }
}

output {
  opensearch {
    hosts => ["https://<opensearch_host>:<opensearch_port>"]
    index => "<aac-index-name>-%{+YYYY.MM.dd}"
    user => "<opensearch_user>"
    password => "<opensearch_password>"
    ssl => true
    ssl_certificate_verification => false
  }
}
```

Logstash 8

```
input {
  http {
    port => <listen_port>
    user => <username>
    password => "<password>"
    codec => json {
      charset => "UTF-8"
    }
  }
}

filter {
  json {
    source => "message"
  }
```

(продолжается на следующей странице)

[49] https://opensearch.org/docs/latest/tools/logstash/index/

(продолжение с предыдущей страницы)

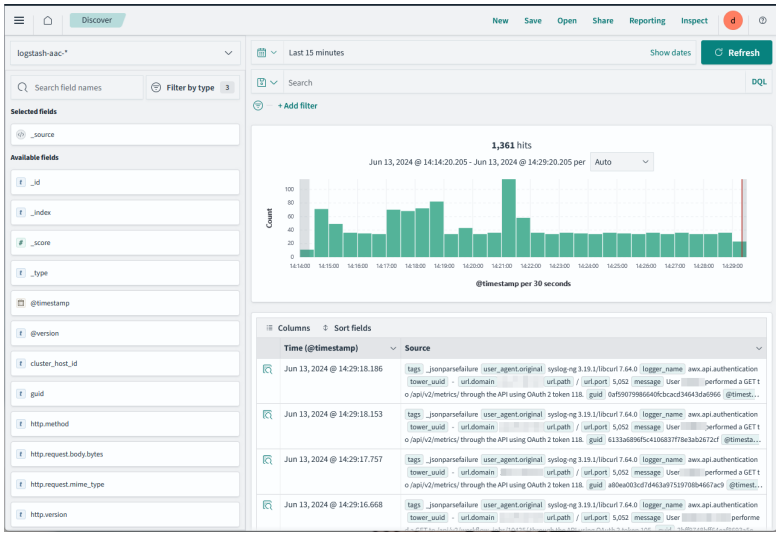
```
}

output {
  opensearch {
    hosts => ["https://<opensearch_host>:<opensearch_port>"]
    index => "<aac-index-name>-%{+YYYY.MM.dd}"
    user => "<opensearch_user>"
    password => "<opensearch_password>"
    ssl => true
    ssl_certificate_verification => false
  }
}
```

где:

- `listen_port` – порт, на котором Logstash будет ожидать соединение;
- `username` – название учетной записи для подключения к серверу Logstash;
- `password` – пароль для подключения к серверу Logstash;
- `opensearch_host` – IP-адрес или FQDN сервера OpenSearch;
- `opensearch_port` – порт сервера OpenSearch;
- `opensearch_user` – название учетной записи пользователя для подключения к серверу OpenSearch;
- `opensearch_password` – пароль для подключения к серверу OpenSearch.

Пример визуализации журналов Automation Controller в режиме Discover[50]:



Обработка журналов с помощью Logstash и Elasticsearch

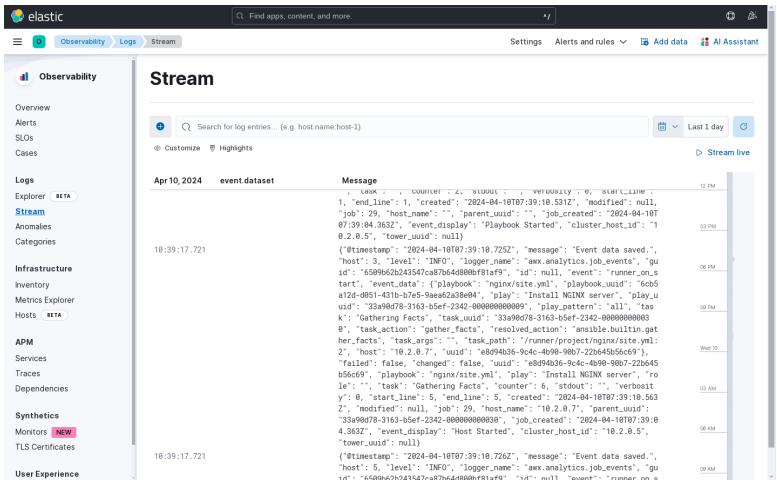
Чтобы настроить Automation Controller на передачу журналов в систему Elasticsearch через агрегатор Logstash, в *настройках журналирования* укажите значения следующих параметров:

- **Агрегатор протоколирования** – FQDN сервера Elasticsearch, например, `test-elastic.zapto.org`.
- **Порт агрегатора протоколирования** – порт сервера Elasticsearch, например, `50000`.
- **Тип агрегатора ведения журнала** – **logstash**.
- **Протокол ведения журналов** – **TCP**.
- **Порог уровня агрегатора ведения журнала** – **INFO**.

[50] https://opensearch.org/docs/2.4/dashboards/discover/index-discover/

- **Включить внешнее ведение журнала** – вкл.

Пример визуализации журналов контроллера в режиме **Stream**:



7.1.3 Мониторинг и аналитика

Для анализа состояния платформы и устранения неполадок необходима система мониторинга и аналитики, которая постоянно следит за состоянием платформы и уведомляет о необходимости выполнения необходимых действий.

Для этих целей можно использовать следующие инструменты:

- **Prometheus** – система сбора, хранения и запроса метрик;
- **Grafana** – платформа визуализации и анализа собранных метрик.

7.2 Настройка производительности

Правильная оценка производительности и планирование ресурсов являются критически важными факторами для стабильной работы платформы Astra Automation в промышленной эксплуатации. Недостаточная емкость компонентов приводит к накоплению очередей заданий, задержкам в обработке событий и деградации отклика веб-интерфейса, что негативно влияет на процессы автоматизации всей IT-инфраструктуры. Платформа включает множество компонентов с различными требованиями к ресурсам:

- Automation Controller для управления и планирования;
- узлы плоскости исполнения (execution plane) для выполнения сценариев автоматизации;
- Private Automation Hub для хранения коллекций Ansible и контейнерных образов;
- Event-Driven Automation для обработки событий в режиме реального времени;
- базы данных PostgreSQL и Redis для хранения данных и кеширования соответственно.

Каждый компонент требует индивидуального подхода к расчету производительности с учетом количества управляемых узлов, интенсивности запуска заданий, уровня параллелизма и объема генерируемых событий. Расчет мощности узлов (capacity) и настройка параметров распараллеливания (forks) позволяют оптимально использовать вычислительные ресурсы и обеспечить масштабируемость платформы. Для типичных рабочих нагрузок рекомендуются узлы с конфигурацией 4 CPU, 16 ГБ RAM и дисковой подсистемой производительностью не менее 3000 IOPS.

Представленные материалы помогут спроектировать архитектуру платформы для инфраструктур различного масштаба – от тестовых стендов до распределенных топологий уровня предприятия с десятками исполняющих узлов.

7.2.1 Концепции

Важно

Эта часть документации находится в стадии разработки.

7.2.2 Система автоматизации процессов

Система автоматизации процессов является центральным звеном Astra Automation, поэтому планирование ресурсов платформы критично для обеспечения общей производительности. Обычно она представляет собой кластер, состоящий из управляющих и исполняющих узлов. Приводимые расчеты и рекомендации помогают оценить требуемое количество узлов и их характеристики в зависимости от объема выполняемых заданий автоматизации.

Расчеты основаны на усредненных показателях потребления системных ресурсов и предназначены для предварительного планирования инфраструктуры. Точные значения могут различаться в зависимости от сложности выполняемых заданий, используемых коллекций, частоты запуска и степени параллельности процессов.

Ниже приведены примерные формулы, позволяющие определить оптимальные параметры кластера.

Общие принципы расчета производительности

Astra Automation использует **capacity-based** модель, в которой доступные ресурсы CPU и памяти определяют, сколько процессов выполнения заданий узел может обрабатывать одновременно. Параметр `forks` влияет на уровень параллельности внутри каждого задания и учитывается при планировании мощности.

Этот подход применяется ко всем типам узлов – как управляющим, так и исполняющим.

Расчет по памяти

Следующая формула показывает, сколько параллельных задач узел может поддерживать с учетом объема оперативной памяти:

$$mem\_capacity = \frac{mem - 2048}{mem\_per\_fork}$$

Здесь:

- `mem` – общий объем оперативной памяти узла в мегабайтах;
- `2048` – память зарезервированная для системных служб и фоновых процессов;
- `mem_per_fork` – объем памяти, потребляемый одной задачей (по умолчанию 100 МБ).

Например, узел с 16 ГБ памяти способен выполнять следующее количество параллельных задач:

$$(16384 - 2048)/100 = 143$$

Расчет по CPU

Следующая формула показывает, сколько задач узел может выполнять одновременно, исходя из числа доступных ядер:

$$cpu\_capacity = cpus \times fork\_per\_cpu$$

Здесь:

- `cpus` – количество ядер;
- `fork_per_cpu` – количество задач, допускаемых на одно ядро (по умолчанию 4).

Например, узел с 4 ядрами способен выполнять следующее количество параллельных задач:

$$4 \times 4 = 16$$

Итоговая мощность узла

Следующая формула объединяет показатели CPU и памяти, учитывая баланс между ними через коэффициент `capacity_adjustment`:

$$final\_capacity = cpu\_capacity + (mem\_capacity - cpu\_capacity) \times capacity\_adjustment$$

Здесь:

- `cpu_capacity` – рассчитанная мощность по CPU;
- `mem_capacity` – рассчитанная мощность по памяти;
- `capacity_adjustment` – коэффициент баланса (0.0–1.0):
  - `0.0` – учитывать только CPU;
  - `1.0` – учитывать только память;
  - промежуточные значения определяют смешанный расчет.

> **Примечание**
>
> В большинстве случаев используется расчет по памяти (`capacity_adjustment = 1.0`), но при высоком числе задач, нагружающих процессор, значение можно уменьшить до `0.5`.

Например, при `capacity_adjustment = 1.0` итоговая мощность узла рассчитывается по следующей формуле:

$$16 + (143 - 16) \times 1.0 = 143$$

Оценка ресурсов и количества управляющих узлов

Управляющие узлы Astra Automation планируют задания, обрабатывают события, координируют кластер и обеспечивают работу пользовательского интерфейса (UI и API).

Минимальные системные требования

Приведенные параметры задают минимальную конфигурацию виртуальной машины для стабильной работы управляющего узла:

Характеристика	Значение
Объем RAM, ГБ	16
Количество ядер CPU	4
Дисковое пространство, ГБ	40 (не менее 20 под каталог `/var/lib/awx`)
Производительность, IOPS	3000

Производительность управляющего узла

Для типовой конфигурации: 4 CPU, 16 ГБ RAM и 3000 IOPS, теоретическая мощность управляющего узла составляет 143 forks (см. *расчет*). Однако часть ресурсов постоянно занята фоновыми процессами: PostgreSQL, Redis, API, UI, receptor и так далее. Эти процессы потребляют около 5–10 % общей мощности узла. С учетом этих расходов эффективная производительность снижается до 130–140 forks. Для расчетов используется усредненное значение 137 forks, которое отражает практическую мощность типового управляющего узла Astra Automation.

При типовой конфигурации достигаются следующие показатели:

- примерная мощность – 137 задач или событий, обрабатываемых одновременно;
- скорость обработки событий при 100% загрузке – 1100–1600 событий/сек;
- рекомендуемое соотношение – 1 управляющий узел на каждые 5 исполняющих узлов, что обеспечивает сбалансированную производительность и устойчивость системы.

Расчет числа управляющих узлов

Чтобы рассчитать количество управляющих узлов, требуемых для обслуживания заданного количества параллельных задач, воспользуйтесь следующей формулой:

$$number\_of\_control\_nodes = \frac{peak\_concurrent\_jobs}{control\_final\_capacity}$$

Здесь:

- `peak_concurrent_jobs` – максимальное количество заданий, выполняющихся одновременно;
- `control_final_capacity` – мощность одного управляющего узла (в среднем 137).

Например, для 10 одновременных заданий необходимо следующее количество управляемых узлов:

$$10/137 = 0.08$$

Полученное значение следует округлить в большую сторону. В этом случае минимальное количество управляющих узлов равно 1, но для обеспечения высокой доступности (HA) рекомендуется использовать 2 узла.

Оценка ресурсов и количества исполняющих узлов

Исполняющие узлы Astra Automation выполняют задания, запущенные управляющим узлом.

Минимальные системные требования

Приведенные параметры задают минимальную конфигурацию виртуальной машины для стабильной работы исполняющего узла:

Характеристика	Значение
Объем RAM, ГБ	16
Количество ядер CPU	4
Дисковое пространство, ГБ	60
Производительность, IOPS	3000

Расход производительности по типам задач

Каждое задание, выполняемое на исполняющем узле, потребляет часть его вычислительных ресурсов. Это потребление выражается в единицах производительности (capacity units) – условных числах, которые используются системой планирования Astra Automation для расчета доступной мощности узлов.

Если доступных единиц производительности на узле недостаточно, новое задание ставится в очередь. Ниже приведены типовые значения, показывающие, сколько единиц производительности расходуется на выполнение разных типов заданий:

Тип задания	Значение	Описание
Задания наборов сценариев	`min(forks, number_of_hosts) + 1`	Основной тип задания. Один управляющий процесс и до `forks` параллельных процессов выполнения задач на узлах
Обновление инвентаря	1	Служебное задание, использующее один процесс
Обновление проекта	1	Служебное задание для синхронизации проекта из SCM
Системные задания	5	Административные или технические процессы, такие как очистка журналов, обновление репозиториев и прочее

Например, если задание набора сценариев содержит `forks = 5` и запускается на `10` управляемых узлах, единицы производительности рассчитываются по следующей формуле:

$$capacity\_units = min(5, 10) + 1 = 5 + 1 = 6$$

Расчет необходимого числа исполняющих узлов

Для расчета числа исполняющих узлов выполните следующие действия:

1. Определите рабочую нагрузку. Для корректного расчета необходимо собрать следующие исходные данные:
   - количество управляемых узлов;
   - частота выполнения задач;
   - максимальное количество одновременных заданий;
   - количество параллельных задач (forks);
   - скорость обработки событий.

   Пример исходных данных:

Параметр	Значение
Управляемые узлы	300
Частота выполнения задач	≈ 200 заданий в день
Одновременные задания	10
Forks	5

2. Вычислите требуемую производительность:

   $$execution\_capacity\_required = (concurrent\_jobs \times forks) + (concurrent\_jobs \times 1)$$

   Здесь:
   - `concurrent_jobs` – максимальное количество одновременных заданий;
   - `concurrent_jobs × forks` – рабочие процессы;
   - `concurrent_jobs × 1` – управляющие процессы каждого задания.

   Например, для исходных данных, приведенных выше, производительность рассчитывается по следующей формуле:

   $$(10 \times 5) + (10 \times 1) = 50 + 10 = 60$$

3. Рассчитайте количество узлов:

   $$number\_of\_execution\_nodes = \frac{execution\_capacity\_required}{final\_capacity}$$

   Здесь:

- `execution_capacity_required` – производительность исполняющего узла;
- `final_capacity` – мощность одного исполняющего узла.

Примечание

Несмотря на одинаковую базовую формулу расчета мощности, реальные значения `final_capacity` для управляющих и исполняющих узлов различаются:

- Для исполняющего узла при конфигурации 4 CPU и 16 ГБ RAM теоретическая мощность составляет около 143 единиц.
- Для управляющего узла – около 137 единиц, так как часть ресурсов постоянно потребляют системные сервисы.

Например, для исходных данных, приведенных выше, количество узлов рассчитывается по следующей формуле:

$$60/143 = 0.42$$

Полученное значение следует округлить в большую сторону. В этом случае минимальное количество исполняющих узлов равно 1, но для обеспечения высокой доступности (HA) рекомендуется использовать 2 узла.

7.3 Техническое обслуживание

Astra Automation представляет собой сложную платформу со значительным количеством различных компонентов. В связи с большой значимостью процессов автоматизации вопросы технической поддержки платформы являются одними из наиболее важных. Для уверенного применения платформы следует изучить приведенные здесь методы и средства технического обслуживания.

При необходимости обращайтесь в службу технической поддержки AA|. Предварительно ознакомьтесь с положением о технической поддержке[51].

Перед созданием обращения в службу технической поддержки убедитесь, что подходящего решения нет в документации Astra Automation и Базе знаний[52]. При создании обращения через Личный кабинет[53] решения из Базы знаний предлагаются автоматически.

Инструкции по использованию Личного кабинета доступны в справочном центре в разделе Личный кабинет пользователя Astra Linux[54].

7.3.1 Резервное копирование и восстановление

Резервное копирование – это процесс создания резервной копии, то есть данных, которые необходимы для восстановления платформы Astra Automation.

Резервная копия представляет собой единый файл архива, включающий в себя следующие данные:

- информация о настройках системы;
- ключи шифрования паролей и секретов базы данных;
- каталог с локальными копиями проектов;
- резервная копия базы данных;
- сертификаты.

Архив состоит из файлов:

[51] https://astra.ru/support/support-docs/polozhenie-o-tekhnicheskoy-podderzhke-astra-automation/
[52] https://wiki.astralinux.ru/kb/
[53] https://lk.astra.ru/
[54] https://wiki.astralinux.ru/pages/viewpage.action?pageId=128615802

- <node>.tar.gz – настройки системы и каталог локальных проектов (<node> – название первого узла в группе [automationcontroller]);
- ca.tar.gz – сертификаты;
- common.tar.gz – ключи шифрования паролей и секретов базы данных;
- postgres.tar.gz – резервная копия базы данных.

Требования к узлу

Рекомендуется выполнять резервное копирование и восстановление на установочном узле или на узле, удовлетворяющем следующим требованиям:

- установленный пакет astra-automation-setup;
- корректно заполненный файл /opt/rbta/aa/astra-automation-setup/inventory.

Утилита назначает владельцем создаваемого файла пользователя, указанного в переменной ansible_user раздела [all:vars] инвентаря. Если эта переменная отсутствует, то этим пользователем будет admin.

Пример настройки:

```
[all:vars]
ansible_user='astra'
```

Предупреждение

Восстанавливать кластер следует в той же конфигурации, которая была на момент создания резервной копии.

При установке с помощью коллекции эти требования соблюдаются на узле с примененной ролью astra.aa_controller.aac.

Резервное копирование

Для выполнения резервного копирования выполните следующие действия:

1. Убедитесь, что в файле inventory в разделе [all:vars] значением параметра ansible_user является название учетной записи пользователя ОС установочного узла, например:

   ```
   [all:vars]
   ansible_user='astra'
   ```

2. Для выполнения резервного копирования выполните команду:

   ```
   sudo ./aa-setup -b
   ```

По умолчанию резервная копия сохраняется в каталог /tmp/backups/astra-automation/. Чтобы сохранить резервную копию в другой каталог, укажите полный путь нему в качестве позиционного аргумента после аргумента -b, например:

```
sudo ./aa-setup -b /opt/backups/
```

В каталоге для хранения резервных копий также создается символическая ссылка astra-automation-backups-latest.tar.gz, которая всегда указывает на самую новую резервную копию в каталоге.

Восстановление из резервной копии

Для восстановления Astra Automation из резервной копии выполните следующие действия:

1. Скопируйте архив с резервной копией в каталог `/tmp/backups/astra-automation/`.
2. Перейдите в каталог `/opt/rbta/aa/astra-automation-setup/` и запустите утилиту `aa-setup` с аргументом `-r`:

   ```
   sudo ./aa-setup -r
   ```

   По умолчанию утилита `aa-setup` восстанавливает резервную копию из самого нового архива, размещенного в каталоге `/tmp/backups/astra-automation/`. Чтобы использовать для восстановления другой архив, укажите полный путь к нему в качестве позиционного аргумента после аргумента `-r`, например:

   ```
   sudo ./aa-setup -r /opt/backups/astra-automation-backup-16-06-2024-155236.tar.gz
   ```

3. Если резервная копия была создана в версии платформы отличающейся от той, в которой она восстанавливается, утилиту `aa-setup` обязательно нужно запустить еще раз:

   ```
   sudo ./aa-setup
   ```

   Это обусловлено несоответствием структуры таблиц базы данных в разных версиях платформы.

4. Для проверки корректности восстановления выполните вход в веб-интерфейс контроллера от имени администратора и проверьте содержимое разделов.

7.3.2 Восстановление работоспособности платформы

При выходе из строя платформы или отдельных ее узлов для восстановления работоспособности выполните описанные ниже действия.

Восстановление отдельных узлов плоскостей управления и исполнения

Если из строя вышли отдельные гибридные, исполняющие, управляющие или промежуточные узлы, для их восстановления выполните следующие действия:

1. *Подготовьте новые узлы* для включения в состав платформы взамен вышедших из строя.
2. Подключитесь к установочному узлу.
3. В инвентаре установщика замените данные вышедших из строя узлов данными новых узлов.
4. Запустите развертывание платформы, следуя *инструкции*.

Восстановление платформы

В случае выхода из строя одного из компонентов платформы целиком (Automation Controller, Private Automation Hub или Event-Driven Automation) порядок действий по его восстановлению одинаковый. Рассмотрим процесс на примере восстановления узла Automation Controller:

1. Подготовьте новые узлы для развертывания контроллера.
2. Выполните процедуру развертывания платформы, следуя *инструкции*. Описание инвентаря должно включать все компоненты платформы согласно ее текущему составу и учитывать все последние успешные изменения в платформе.
3. Скопируйте на установочный узел самую свежую резервную копию платформы и *восстановите состояние платформы из нее*.

7.3.3 Миграция

В организациях может возникнуть необходимость мигрировать платформу Astra Automation с одних узлов на другие. При этом часто изменяются доменные имена и IP-адреса узлов платформы.

Чтобы выполнить такую миграцию, следует воспользоваться функциональностью утилиты `aa-setup`. Здесь рассматривается пример миграции платформы Astra Automation с одной конфигурации на другую.

Важно

- Версии ОС на установочном узле и узлах платформы не изменяются.
- Версия платформы не изменяется.
- Допускается миграция платформы между конфигурациями с разным количеством узлов.

Пусть исходная конфигурация платформы имеет следующие параметры:

- Развернут кластер Automation Controller из трех узлов:

Узел	Назначение	FQDN	IP-адрес
`node1`	Управляющий узел	`node1.example.com`	`192.168.56.11`
`node2`	Управляющий узел	`node2.example.com`	`192.168.56.12`
`node3`	Узел СУБД	`node3.example.com`	`192.168.56.13`

- Для администрирования узлов используется учетная запись `administrator`.
- Сервер СУБД PostgreSQL развернут средствами платформы.

Далее эта платформа будет называться исходной платформой, а соответствующий ей установочный узел – исходным установочным узлом.

Пусть целевая конфигурация платформы имеет следующие параметры:

- Для развертывания платформы подготовлены три узла:

Узел	Назначение	FQDN	IP-адрес
`ac1`	Управляющий узел	`ac1.example.com`	`10.120.0.11`
`ac2`	Управляющий узел	`ac2.example.com`	`10.120.0.12`
`rdbs`	Узел СУБД	`rdbs.example.com`	`10.120.0.101`

- Установочный узел и узлы платформы настроены согласно инструкций из документа *Подготовка узлов*.
- Для администрирования узлов платформы используется учетная запись `devops`.
- Сервер СУБД PostgreSQL будет развернут средствами платформы.

Далее эта платформа будет называться целевой платформой, а соответствующий ей установочный узел – целевым установочным узлом.

Совет

В качестве целевого установочного узла можно использовать исходный установочный узел.

Создание резервной копии

Для создания резервной копии подключитесь к исходному установочному узлу и выполните следующие действия:

1. Перейдите в каталог `/opt/rbta/aa/astra-automation-setup/`:

```
cd /opt/rbta/aa/astra-automation-setup/
```

2. Запустите утилиту `aa-setup` с аргументом `--backup` (`-b`):

```
sudo ./aa-setup --backup
```

   Если инвентарь хранится в файле формата YAML, укажите путь к нему в значении аргумента `--inventory` (`-i`):

```
sudo ./aa-setup --backup --inventory=/path/to/inventory.yml
```

3. Если необходимо использовать другой установочный узел, любым удобным способом скопируйте на него самый новый архив с резервной копией. Например, чтобы скопировать архив по сети, выполните команду:

```
scp /var/backups/astra-automation/astra-automation-backup-latest.tar.gz <user>@
↪<host>:/var/backups/astra-automation/astra-automation-backup-latest.tar.gz
```

   где:

   - <user> – название учетной записи пользователя целевого установочного узла;
   - <host> – FQDN или IP-адрес установочного узла целевого кластера.

Восстановление из резервной копии

Для восстановления платформы из резервной копии подключитесь к целевому установочному узлу и выполните следующие действия:

1. Перейдите в каталог `/opt/rbta/aa/astra-automation-setup/`:

```
cd /opt/rbta/aa/astra-automation-setup/
```

2. В инвентарь установщика внесите изменения, соответствующие новой конфигурации платформы. Для описанной выше конфигурации файл инвентаря необходимо привести к следующему виду:

   INI

```
[automationcontroller]
ac1.example.com   ansible_host=10.120.0.11
ac2.example.com   ansible_host=10.120.0.12

[database]
rdbs.example.com   ansible_host=10.120.0.101

[all:vars]
ansible_user='devops'
ansible_ssh_private_key_file='/path/to/private/ssh/key'
admin_username='admin'
admin_email='admin@example.com'
admin_password='p@ssW0rd!'
pg_host=10.120.0.101
pg_port=5432
```

YAML

```
automationcontroller:
  hosts:
    ac1.example.com:
      ansible_host: 10.120.0.11
    ac2.example.com:
      ansible_host: 10.120.0.12

database:
  hosts:
    rdbs.example.com:
      ansible_host: 10.120.0.101

all:
  vars:
    ansible_user: devops
    ansible_ssh_private_key_file: /path/to/private/ssh/key
    admin_username: admin
    admin_email: admin@example.com
    admin_password: p@ssW0rd!
    pg_host: 10.120.0.101
    pg_port: 5432
```

3. Запустите развертывание платформы:

```
sudo ./aa-setup
```

Если инвентарь хранится в файле формата YAML, укажите путь к нему в значении аргумента `--inventory` (`-i`):

```
sudo ./aa-setup --inventory=/path/to/inventory.yml
```

4. Запустите восстановление платформы из резервной копии:

```
sudo ./aa-setup --restore
```

Если инвентарь хранится в файле формата YAML, укажите путь к нему в значении аргумента `--inventory` (`-i`):

```
sudo ./aa-setup --restore --inventory=/path/to/inventory.yml
```

5. Для проверки корректности восстановления выполните вход в веб-интерфейс контроллера от имени администратора и проверьте содержимое разделов.

7.3.4 Обслуживание процессов автоматизации

Дополнительные настройки контроллера

Большинство настроек контроллера можно изменить через графический интерфейс, CLI или API. При этом изменения автоматически применяются на всех узлах управляющей плоскости.

Значения по умолчанию для многих настроек хранятся в конфигурационном файле `/var/lib/awx/venv/awx/lib/python3.9/site-packages/awx/settings/defaults.py`. Также в этом файле задаются значения для нескольких настроек, которые нельзя изменить указанными выше способами.

Если для некоторых настроек необходимо задать значения, отличающиеся от значений по умолчанию, на всех узлах управляющей плоскости выполните следующие действия:

1. В каталоге `/etc/tower/conf.d/` создайте файл `custom.py`, в котором укажите нужные значения соответствующих настроек.
2. Перезапустите службы контроллера:

```
sudo automation-controller-service restart
```

Предупреждение

Хотя в некоторых случаях управление настройками указанным способом может быть оправданным, описанный способ не рекомендуется по следующим причинам:

- Поведение многих параметров не документировано.
- Контроллер не проверяет корректность настроек в `/etc/tower/conf.d/custom.py` на этапе заполнения.
- Контроллер не обеспечивает синхронизацию содержимого файла `/etc/tower/conf.d/custom.py` между узлами. Любые изменения в этом файле нужно будет распространять на узлах из плоскости управления вручную, в том числе при добавлении новых узлов.
- Настройки, заданные в `/etc/tower/conf.d/custom.py`, могут быть переопределены значениями, хранящимися в базе данных.

Ниже приводятся примеры изменения значений настроек контроллера, которые невозможно изменить через графический интерфейс, CLI или API.

- Период проверки статуса узлов в секундах:

```
CLUSTER_NODE_HEARTBEAT_PERIOD = 30
```

- Количество неуспешных проверок статуса узла, при котором он считается вышедшим из строя:

```
CLUSTER_NODE_MISSED_HEARTBEAT_TOLERANCE = 1
```

- Изменение пути к каталогу для хранения проектов:

```
PROJECT_ROOT = '/var/share/projects/'
```

- Дополнительный URL, по которому будет доступен API контроллера:

```
OPTIONAL_API_URLPATTERN_PREFIX = 'controller'
```

  Теперь обращаться к API контроллера можно будет по двум URL:

  - `https://controller.example.com/api/`;
  - `https://controller.example.com/api/controller/`.

- Ограничение на отправку cookie при кросс-доменных запросах.

```
SESSION_COOKIE_SAMESITE = True
```

  Если значение настройки `SESSION_COOKIE_SAMESITE` равно `True`, для cookie `userLoggedIn` устанавливается политика `SameSite` со значением `Lax`[55]. При использовании этой политики браузер не передает cookie на сторонние сайты, тем самым предотвращая их перехват злоумышленником, например, при загрузке фреймов или изображений по незащищенному соединению.

  Значение по умолчанию: `False`.

  В большинстве браузеров политика `SameSite=Lax` включена по умолчанию.

[55] https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie#samesitesamesite-value

Решение проблем с помощью awx-manage

При работе контроллера возможен выход из строя как отдельных его узлов, так и групп узлов. Повторный запуск сценария настройки не приводит к автоматическому исключению из топологии таких узлов и групп узлов, поскольку у утилиты aa-setup нет возможности определить причину выхода узла или группы узлов из строя. Для восстановления работоспособности контроллера можно использовать утилиту awx-manage.

Удаление одного узла

Для удаления одного узла выполните следующие действия:

1. Подключитесь по SSH к любому управляющему узлу контроллера.
2. Запустите утилиту awx-manage с параметром deprovision_instance:

   ```
   sudo awx-manage deprovision_instance --hostname=<hostname>
   ```

   Здесь <hostname> – название удаляемого узла в инвентаре установщика платформы.

Примечание

Узел необходимо удалить из инвентаря установщика платформы, если он не должен появиться в кластере при повторном запуске сценария настройки.

Удаление группы узлов

Для удаления группы узлов выполните следующие действия:

1. Удалите каждый узел отдельно как описано ранее.
2. Удалите группу узлов:

   ```
   sudo awx-manage unregister_queue --queuename=<queue_name>
   ```

   Здесь <queue_name> – название группы узлов.

7.3.5 Использование базы знаний

Важно

Эта часть документации находится в стадии разработки.

7.3.6 Поиск причин неисправностей

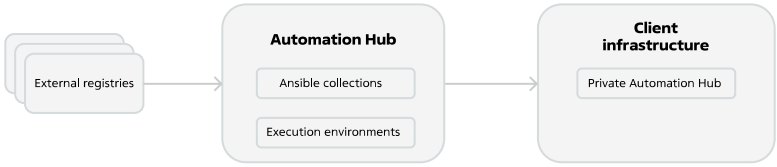
Важно

Эта часть документации находится в стадии разработки.

ГЛАВА 8

Глобальная библиотека контента

Реестр Automation Hub предоставляет коллекции Ansible и образы среды исполнения, разработанные ПАО Группа Астра и ее партнерами. Содержимое этого реестра рекомендуется загрузить на *Private Automation Hub* и использовать последний в своих проектах.



Хранящиеся в Automation Hub образы среды исполнения созданы на базе ОС Astra Linux Special Edition.

Примечание

Примеры кода с использованием контента Automation Hub доступны на сайте https://source.astragroup.ru/ с аутентификацией через Личный кабинет[56].

8.1 Коллекции Ansible

Использование коллекций позволяет значительно упростить создание сценариев для решения самых разных задач по развертыванию и настройке ПО с помощью Ansible.

8.1.1 Назначение

Коллекции Ansible, доступные в реестре Automation Hub[57], предназначены для развертывания и настройки продуктов ПАО Группа Астра и вспомогательного стороннего ПО.

Совет

Рекомендуется загрузить необходимые коллекции из Automation Hub в Private Automation Hub.

От коллекций, имеющихся в свободном доступе, они отличаются следующим:

[56] https://lk.astra.ru/
[57] https://hub.astra-automation.ru/

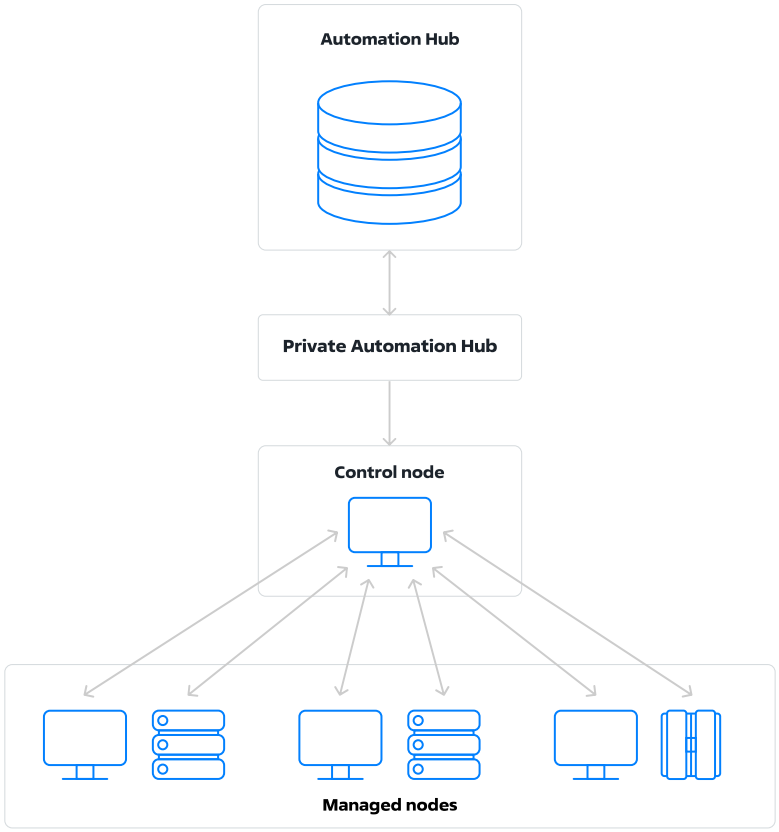
- Все роли распространяются в составе коллекций Ansible, в которые включаются также модули, специфичные для продуктов ПАО Группа Астра.
- При разработке коллекций учитываются особенности продуктов ПАО Группа Астра.
- Длительный срок поддержки.
- Доработка функциональности исходя из потребностей клиентов ПАО Группа Астра.
- Оперативное исправление всех найденных ошибок.
- Многократная проверка кода, в том числе в реальных условиях.
- Единая логика и стандартизированные подходы при написании кода, повышающие качество продукта.
- Названия всех коллекций используют пространство имен `astra`.
- Версии коллекций формируются с использованием семантического версионирования[58] (semantic versioning).

Роли, размещенные в Automation Hub, обладают следующими свойствами:

- Распространение только в составе коллекций.
- Каждая роль используется для установки и настройки только определенной функциональности. Например, роль `astra.ald_pro.client` используется только для настройки клиентов домена ALD Pro. Для настройки контроллера или его реплики следует использовать роли `astra.ald_pro.controller` и `astra.ald_pro.replica` соответственно.

8.1.2 Структура рабочего окружения

При использовании коллекций из реестра Automation Hub схема работы Ansible имеет вид:



Перед выполнением набора сценариев следует установить на управляющий узел все необходимые коллекции. Инструкции по установке коллекций приведены ниже в секции *Применение*.

[58] https://semver.org/

8.1.3 Применение

Использование коллекций Ansible из реестра Automation Hub имеет следующие особенности:

- Необходимо настроить доступ к реестру Automation Hub согласно *инструкции*.
- Для описания зависимостей создайте в каталоге проекта файл `requirements.yml` с записями следующего вида:

```
---
collections:
  - name: astra.<collection_name>
  # ...
```

  где <collection_name> – название коллекции. Например, для коллекции `astra.ceph` указанная запись должна иметь вид:

```
---
collections:
  - name: astra.ceph
```

- При использовании роли в сценариях указывайте *FQCN*, например:

```
---
- name: Set up ALD Pro domain controller
  hosts: dc01
  # ...
  roles:
    - role: astra.ald_pro.controller
      vars:
        aldpro_domain: aldpro.example.com
        aldpro_pdc_ip: 192.168.56.11
        aldpro_pdc_name: dc01
        aldpro_admin_password: p@ssW0rD!
```

- Для установки необходимых ресурсов согласно зависимостям, определенным в файле `requirements.yml`, используйте команду:

```
ansible-navigator exec \
   --eei hub.astra-automation.ru/aa-1.2/aa-minimal-ee \
   --eev ~/.ssh/hub.astra-automation.ru:/root/.ssh/id_rsa \
   -- ansible-galaxy install -r requirements.yml
```

- Для запуска набора сценариев используйте команду:

```
ansible-navigator run playbook.yml --eei hub.astra-automation.ru/aa-1.2/aa-minimal-
↪ee -m stdout
```

Для выполнения команд `ansible-navigator` использует *среду исполнения*, заданную аргументом `--eei`.

8.2 Версионирование коллекций

При описании зависимостей Ansible рекомендуется всегда явно указывать номер версии используемой коллекции. Это позволит избежать незапланированных изменений в существующем окружении и несовместимости новой версии коллекции с ним.

Коллекции версионируются согласно правилу семантического версионирования[59]. Версия коллекции выглядит как:

```
<major_version>.<minor_version>.<patch>[-rc.<rc_version>]
```

где:

[59] https://semver.org/

- <major_version> – номер мажорной версии. Версия 0 используется для начальной разработки. Номер мажорной версии увеличивается, если в код коллекции добавляются изменения API.
- <minor_version> – номер минорной версии. Увеличивается при добавлении новой функциональности или существенных изменениях в уже существующей функциональности без нарушения совместимости с более ранним API. Минорная версия обнуляется при увеличении мажорной версии.
- <patch> – номер срочного обновления при исправлении ошибок. Увеличивается после исправления ошибок в имеющейся функциональность без потери обратной совместимости. Новая функциональность не вносится, существенные изменения в уже существующей не производятся. Это поле обнуляется при увеличении мажорной и минорной версий.
- (опционально) -rc.<rc_version> – версия Release Candidate (RC) указывает на то, что это еще не финальная версия, но функциональность почти готова к выпуску. Номер версии RC увеличивается по мере подготовки коллекции к выпуску.

8.2.1 Получение списка версий коллекции

Для получения списка доступных версий коллекции выполните следующие действия:

1. Перейдите на страницу коллекции в соответствующем разделе Automation Hub[60].
2. Нажмите на выпадающий список **Версия** (Version).

8.2.2 Указание версии коллекции в файле зависимостей

Чтобы зафиксировать номер версии используемой коллекции, добавьте к соответствующей записи в *файле зависимостей* строку с параметром version, например:

```
---
collections:
  - name: astra.ald_pro
    version: "<version>"
```

Если это поле отсутствует, будет загружена самая свежая версия коллекции из доступных. В значении поля version можно задать условия, определяющие допустимый диапазон номеров устанавливаемой версии коллекции:

- * – самая свежая (latest) версия. Это значение используется по умолчанию.
- != – использовать любую версию, кроме указанной.
- == – использовать строго указанную версию.
- >= – использовать указанную версию или более новую.
- > – использовать более новую версию, чем указанная.
- <= – использовать указанную версию или более старую.
- < – использовать версию более старую, чем указанная.

Смотрите более подробную информацию о версионировании коллекций в документации Ansible[61].

8.2.3 Примеры

- Строгое указание версии

```
version: "==0.1.1"
```

- Указанная или более новая версия

[60] https://hub.astra-automation.ru/
[61] https://docs.ansible.com/ansible/latest/collections_guide/collections_installing.html#installing-an-older-version-of-a-collection

```
version: ">=0.1"
```

8.3 Сертифицированные и проверенные коллекции

8.3.1 Форматы распространения кода Ansible

Код для автоматизации с помощью Ansible может включать следующие элементы:

- сценарии;
- определения переменных;
- модули и другие.

Код может иметь произвольную структуру каталогов в проекте. Для обеспечения модульности и переиспользования Astra Automation предлагает контент в виде коллекций Ansible, которые объединяют элементы кода в стандартную структуру. Смотрите более подробную информацию в *описании коллекций Ansible*.

Все коллекции Astra Automation доступны в Automation Hub[62] и являются *проверенными*.

8.3.2 API

Коллекции Ansible используются посредством подключения их к другому коду Ansible (например, к сценарию или другим коллекциям) в качестве зависимостей и «библиотек» – подобно тому, как аналогичный подход применяется в высокоуровневых языках программирования. У пользователя коллекции нет необходимости редактировать ее код, чтобы использовать необходимую функциональность в своем окружении.

Пользователь взаимодействует с коллекцией с помощью следующих элементов интерфейса:

- переменные ролей и сценариев, объявленные в коллекции;
- инвентарь в формате, определенном коллекцией;
- параметры модулей, содержащихся в коллекции;
- названия ролей, модулей и наборов сценариев.

Совокупность перечисленных интерфейсов можно воспринимать как API коллекции, который для каждой коллекции описан в ее документации. API определяется функциональными возможностями коллекции, но также на него может влиять состав *Execution Environment*.

8.3.3 Проверенный контент

В условиях динамического развития продуктов ПАО Группа Астра, а также продуктов других компаний, потенциальные пользователи Astra Automation хотели бы получать код автоматизации на тестирование как можно раньше, даже если он не полностью охватывает все запланированные возможности и не стабилизирован. К такому коду со стороны Astra Automation предъявляется ограниченный набор требований:

- Код организован в соответствии со структурой коллекции Ansible.
- Коллекция версионируется в соответствии с SemVer (Semantic Versioning).
- Код не требует интерактивного взаимодействия с пользователем.
- Коллекция содержит документацию на все компоненты (роли, модули, общий `README.md`).

После прохождения проверки инженерами Astra Automation коллекция, отвечающая данным требованиям, получает статус проверенной.

Проверенный контент имеет следующие ограничения:

[62] https://hub.astra-automation.ru/

- Отсутствует фиксированный жизненный цикл и выпуски с длительным сроком поддержки.
- Код может быть неидемпотентным[63].

8.3.4 Сертифицированный контент

Коллекции Ansible могут быть сертифицированы в рамках определенной версии Astra Automation. К сертифицированным коллекциям предъявляются следующие требования (помимо требований, предъявляемых к проверенным коллекциям):

- Код не должен дублироваться. Если коллекция collection1 разработана для настройки сервиса service1, которому для функционирования необходим сервис service2, код настройки которого уже есть в составе отдельной сертифицированной коллекции collection2, то запрещается дублировать его в коллекцию collection1.
- Коллекция настраивает только тот объект (сервис, устройство), для которого разрабатывается. Коллекция не должна содержать реализации функций настройки каких-либо других объектов – такие функции должны быть добавлены в соответствующие сертифицированные коллекции.
- Код коллекции должен быть идемпотентным.
- Возможности коллекции должны быть покрыты тестами.

8.3.5 Жизненный цикл

Для сертифицированных коллекций поддерживаются следующие типы выпусков:

- **FR (feature release)** – выпуски с новыми возможностями коллекции. Периодичность выпуска составляет 4 недели, если есть готовые к публикации изменения.
- **LTS (long-term support)** – выпуски с длительным сроком поддержки в течение 24 месяцев. В таких выпусках зафиксированы функциональность коллекции и версии поддерживаемых продуктов. Выпуски получают только обновления безопасности и устранение ошибок, а также совместимость с новой версией Automation Controller, если подходит к концу срок поддержки текущей версии контроллера.

Порядок выпуска FR

В рамках выпуска FR в сертифицированных коллекциях могут происходить следующие изменения, отражающиеся в соответствующем изменении версии коллекции согласно семантическому версионированию:

- **Исправления** – исправления обнаруженных ошибок, в том числе исправление и дополнение документации.
- **Минорные обновления** – добавление новых функций или новой версии коллекции, не затрагивающие обратную совместимость с текущей мажорной версией API. Это означает, что сценарии и другие коллекции, использующие данную коллекцию, не потребуют адаптации для уже задействованных в них возможностей коллекции.
- **Мажорные обновления** – значительные изменения в коллекции, которые меняют формат входных/выходных параметров ролей, сценариев и других компонентов коллекции (API). Для таких версий теряется обратная совместимость с предыдущими мажорными версиями коллекции. Изменения могут быть связаны с добавлением поддержки новой версии продукта, исправлениями критических ошибок или удалением компонентов коллекции.

При внесении изменений в служебные файлы (`taskfile.yml`, конфигурации linter и другие) и тесты версия коллекции не меняется.

Версия FR выпускается каждые 4 недели или чаще в случае обнаружения значительных ошибок. Если изменений в коллекции за цикл выпуска не произошло, новая версия не выпускается.

[63] https://en.wikipedia.org/wiki/Idempotence

В рамках выпусков FR может добавляться поддержка новых версий продуктов. Для сертифицированных коллекций это должно происходить при очередном выпуске FR после даты релиза продукта или ранее.

Порядок выпуска LTS

Использование долговременно-поддерживаемых версий (LTS, long-term support) позволяет сопровождать инфраструктуру с помощью коллекции с зафиксированным API в течение длительного времени. Новый выпуск LTS формируется каждые 18 месяцев на основе текущего выпуска FR, что позволяет плавно мигрировать с предыдущего LTS-выпуска.

Выпуски LTS поддерживаются в течение 24 месяцев со дня выпуска. В течение этого периода коллекция получает следующие обновления:

- исправления обнаруженных ошибок;
- исправления безопасности;
- новые функции, если это требуется для исправлений.

Мажорная и минорные значения версии коллекции всегда зафиксированы для выпуска LTS. В версии коллекции изменяется только поле patch.

8.3.6 Поддержка

Для сертифицированных и проверенных коллекций клиент может направлять запросы о нововведениях или исправлениях ошибок в поддержку Astra Automation. Запросы будут отрабатываться инженерами Astra Automation и партнерами, сопровождающими сертифицированную коллекцию.

8.3.7 Удаление коллекций

Коллекции могут терять свой статус сертифицированной или проверенной, а также быть удалены с Automation Hub[64].

Коллекция теряет статус сертифицированной в следующих случаях:

- Если сопровождающий коллекцию не обеспечивает устранение ошибок для выпусков LTS.
- Если сопровождающий коллекцию не обеспечил совместимость с новой версией Astra Automation.
- Если закончилась поддержка версии Astra Automation, для которой сертифицировалась коллекция.

Коллекция теряет статус проверенной, когда версия Automation Controller, для которой она предназначена, перестает поддерживаться.

8.4 Настройка доступа к Automation Hub

Для получения содержимого из Automation Hub необходим токен. Чтобы создать и использовать его, выполните следующие действия:

1. Авторизуйтесь на портале Automation Hub[65].
2. На панели навигации выберите *Коллекции* ▸ *Токен API* (*Collections* ▸ *API Token*).
3. Нажмите кнопку *Загрузить токен* (Download token).
4. Используйте показанный на странице токен для выполнения следующих операций:
    - *синхронизация коллекций* в Private Automation Hub с Automation Hub;
    - *создание образа среды исполнения*, содержащего необходимые коллекции.

[64] https://hub.astra-automation.ru/
[65] https://hub.astra-automation.ru/

Важно

Период действия токена – один год. По истечении указанного времени его нужно будет создать заново.

При создании нового токена предыдущий перестает действовать немедленно.

8.5 Справочные данные

В реестре Automation Hub[66] доступны следующие коллекции Ansible:

Коллекция	Название ПО	Комментарии
`astra.aa-controller`	Astra Automation	Развертывание и настройка компонентов платформы Astra Automation
`astra.ald_pro`	ALD Pro[67]	Домен, управляемый заданным количеством контроллеров, на базе ALD Pro
`astra.astralinux`	Astra Linux Special Edition[68]	Автоматизация базовых операций по управлению ОС Astra Linux Special Edition
`astra.brest`	ПК СВ «Брест»[69]	Программный комплекс средств виртуализации
`astra.ceph`	Ceph[70]	Система хранения данных Ceph с вариациями: • распределенная; • на одном сервере.
`astra.chrony`	Chrony[71]	Сервер времени
`astra.cups`	CUPS[72]	Служба печати
`astra.dcimanager`	DCImanager[73]	Платформа для управления физической и информационной инфраструктурой
`astra.dhcp`	ISC DHCP[74]	Служба динамического назначения IP-адресов и сетевых имен
`astra.docker`	Docker[75]	Система контейнеризации
`astra.etcd`	etcd[76]	Распределенное хранилище данных в формате «ключ-значение»
`astra.freeipa`	FreeIPA[77]	Система идентификации пользователей на базе FreeIPA в составе: • сервер; • реплика; • клиент.
`astra.grafana`	Grafana[78]	Система мониторинга
`astra.haproxy`	HAProxy[79]	Балансировщик нагрузки
`astra.hardening`	Astra Linux Special Edition	Автоматизация настройки систем безопасности Astra Linux Special Edition

продолжается на следующей странице

[66] https://hub.astra-automation.ru/

Таблица 1 – продолжение с предыдущей страницы

Коллекция	Название ПО	Комментарии
astra.iscsi	Open-iSCSI[80]	Open-iSCSI в составе: • сервер-инициатор; • сервер-цель.
astra.keepalived	Keepalived[81]	Балансировщик нагрузки
astra.keycloak	Keycloak[82]	Поставщик услуг аутентификации (SSO)
astra.memcached	Memcached[83]	Система кеширования
astra.nfs	NFS[84]	Настройка клиента и сервера NFS
astra.nginx	NGINX[85]	Web-сервер
astra.ocfs2	OCFS2[86]	Кластерная файловая система *OCFS2* на заданном количестве узлов. Требует предварительной установки с помощью коллекций astra.ceph или astra.iscsi.
astra.patroni	Patroni[87]	Сервис кластеризации СУБД PostgreSQL
astra.pgbouncer	PgBouncer[88]	Менеджер подключений к кластеру СУБД PostgreSQL
astra.postgresql	PostgreSQL[89]	Кластер СУБД PostgreSQL
astra.prometheus	Prometheus[90]	Система мониторинга
astra.rabbitmq	RabbitMQ[91]	Брокер сообщений
astra.repo_mirror	Astra Linux Special Edition	Создание зеркал DEB-репозиториев и управление ими
astra.rubackup	RuBackup[92]	Система резервного копирования
astra.rupost	RuPost[93]	Почтовая система на базе RuPost. Коллекция зависит от коллекции PostgreSQL. Для полноценной работы требует настроенного домена LDAP.
astra.termidesk	Termidesk[94]	Диспетчер подключений виртуальных рабочих мест (VDI)
yandex.cloud	–	Управление ресурсами в Yandex Cloud

Полный комплект коллекций поставляется в составе *EE* – в образе `aa-full-ee`. Выберите *версию aa-full-ee*, содержащую необходимые версии коллекций.

[67] https://www.aldpro.ru/
[68] https://astralinux.ru/
[69] https://astra.ru/software-services/application-software-astra-group/brest/
[70] https://ceph.io/
[71] https://chrony-project.org/
[72] https://www.cups.org/
[73] https://www.ispsystem.ru/dcimanager
[74] https://www.isc.org/dhcp/
[75] https://www.docker.com/
[76] https://etcd.io/
[77] https://www.freeipa.org/
[78] https://grafana.com/
[79] https://www.haproxy.org/
[80] https://www.open-iscsi.com/
[81] https://keepalived.org/
[82] https://www.keycloak.org/
[83] https://memcached.org/
[84] https://ru.wikipedia.org/wiki/Network_File_System
[85] https://nginx.org/
[86] https://archive.kernel.org/oldwiki/ocfs2.wiki.kernel.org/
[87] https://patroni.readthedocs.io/en/latest/
[88] https://www.pgbouncer.org/
[89] https://www.postgresql.org/
[90] https://prometheus.io/
[91] https://www.rabbitmq.com/
[92] https://www.rubackup.ru/
[93] https://www.rupost.ru/
[94] https://termidesk.ru/

ГЛАВА 9

Интерфейс платформы

Astra Automation с помощью шлюза платформы (Platform Gateway) предоставляет единую точку входа для управления всеми компонентами платформы. Шлюз обрабатывает аутентификацию и авторизацию, объединяя все сервисы в единый пользовательский интерфейс и предоставляя доступ к компонентам платформы двумя способами:

- веб-интерфейс (web interface);
- RESTful API.

Объединенный интерфейс позволяет пользователям аутентифицироваться один раз и получить доступ ко всем функциям платформы, включая:

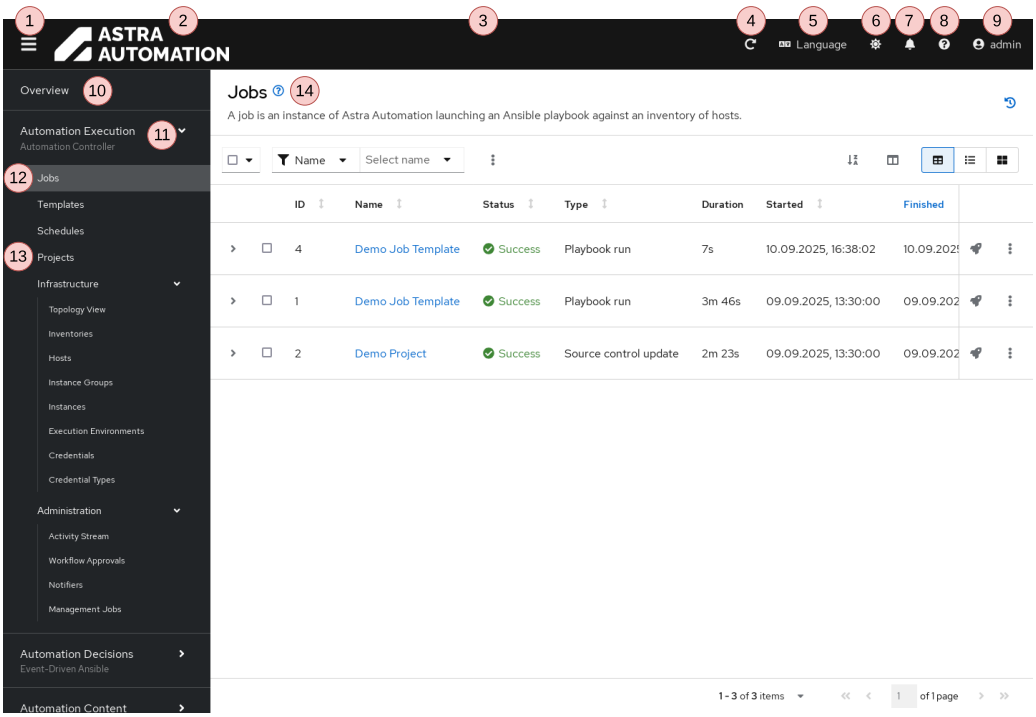
- *Автоматизация процессов* (Automation execution);
- *Управление контентом* (Automation content);
- *Обработка событий* (Automation decisions);
- *Управление доступом* (Access management).

9.1 Графический интерфейс

Платформа Astra Automation предоставляет для пользователей различного уровня единый пользовательский интерфейс (называемый также UI, веб-интерфейс, графический интерфейс) для управления проектами автоматизации, инвентарем, рабочими процессами и политиками безопасности. Единое пространство интерфейса облегчает централизованное выполнение задач, мониторинг процессов и контроль над всеми аспектами автоматизации из одной панели управления, обеспечивая удобство и прозрачность для пользователей всех уровней опыта.

9.1.1 Обзор элементов навигационной панели

На схеме представлены основные компоненты пользовательского интерфейса:



Здесь:

1. Кнопка переключения видимости панели навигации.
2. Логотип.
3. Панель инструментов.
4. Кнопка обновления данных на странице.
5. Кнопка смены языка.
6. Кнопка смены темы оформления.
7. Кнопка-индикатор, показывающая количество уведомлений.
8. Кнопка вызова справки.
9. Пользовательское меню.
10. Панель навигации.
11. Кнопка переключения видимости раздела навигации.
12. Активный раздел навигации.
13. Неактивные разделы навигации.
14. Заголовок окна.

Панель навигации включает следующие разделы:

- **Обзор** (Overview).

  Раздел предоставляет сводную информацию о текущем состоянии платформы, основных показателях автоматизации, активности пользователей и выполнении задач. Здесь можно быстро оценить эффективность работы, выявить возможные проблемы и получить доступ к аналитическим данным по выполненным процессам.

  Подробное описание раздела **Обзор** (Overview) см. *ниже*.

- **Автоматизация процессов** (Automation Execution).

  В этом разделе осуществляется управление заданиями, расписаниями, мониторинг состояния автоматизации и просмотр истории запусков. Пользователь может создавать, запускать, отслеживать и анализировать автоматизированные процессы с помощью контроллера автоматизации.

Подробное описание раздела **Автоматизация процессов** (Automation Execution) см. в документе *Графический интерфейс*.

- **Обработка событий** (Automation Decisions).

  Раздел предназначен для настройки и использования Event-Driven Automation. Пользователь может создавать и управлять средами принятия решений и настраивать события для триггеров автоматических реагирований платформы.

  Подробное описание раздела **Обработка событий** (Automation Decisions) см. в документе *Графический интерфейс*.

- **Контент автоматизации** (Automation Content).

  Раздел отвечает за управление коллекциями, ролями и сценариями в Private Automation Hub. Здесь осуществляется публикация, просмотр, редактирование, удаление и организация контента, а также интеграция с внешними источниками.

  Подробное описание раздела **Контент автоматизации** (Automation Content) см. в документе *Графический интерфейс*.

- **Управление доступом** (Access Management).

  В этом разделе производится централизованное управление аутентификацией, правами пользователей, группами и ролями. Здесь доступны настройки внешних методов аутентификации, токенов и *RBAC*.

  Подробное описание раздела **Управление доступом** (Access Management) см. в документе *Графический интерфейс*.

- **Настройки** (Settings).

  Данный раздел предоставляет единый центр настройки платформы, включая параметры выполнения, интеграций, безопасности, лицензирования, сбор аналитики и другие системные настройки.

  Подробное описание раздела **Настройки** (Settings) см. в документе *Графический интерфейс*.

9.1.2 Обзор (Overview)

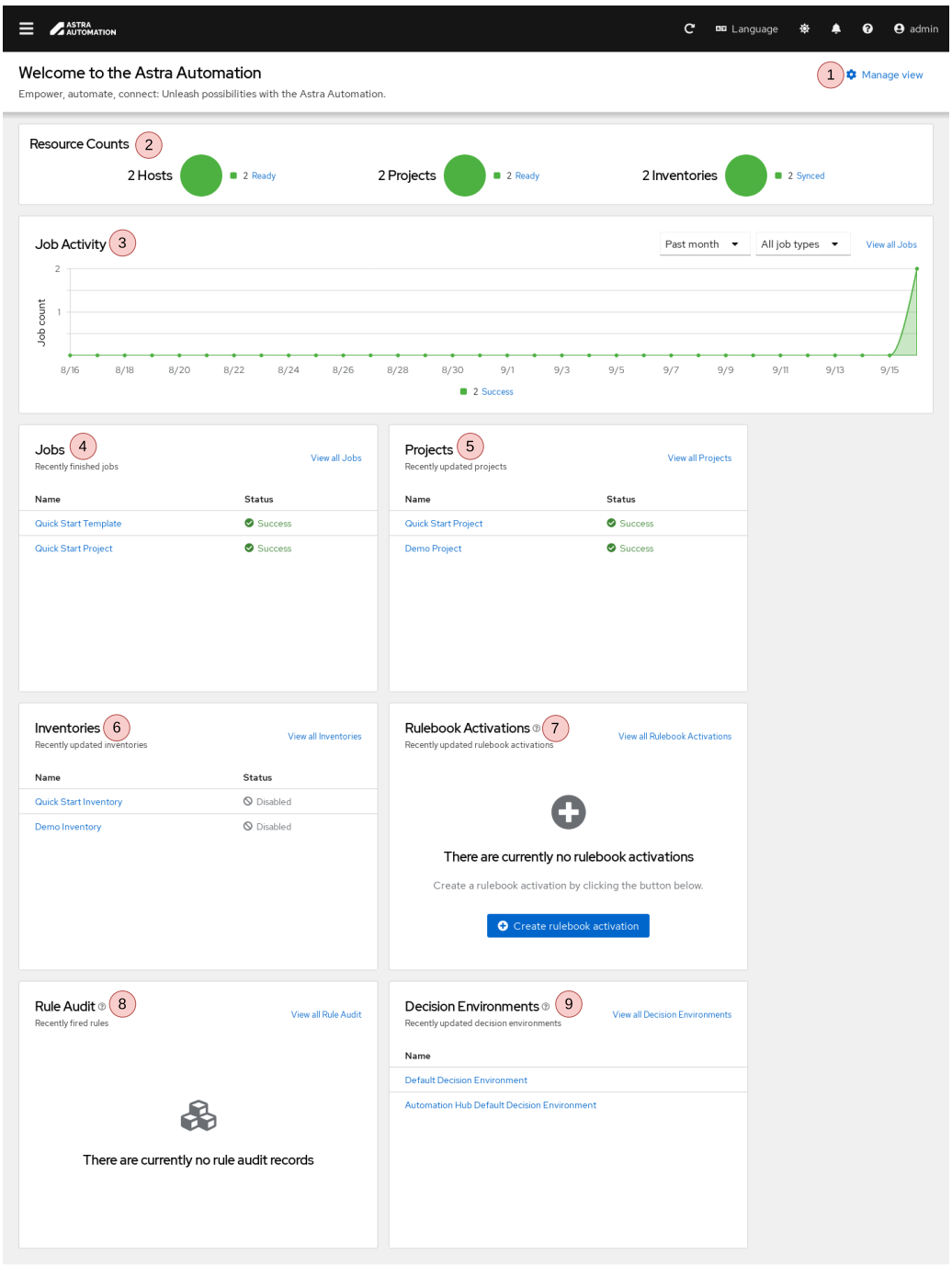
Раздел **Обзор** (Overview) является центральным элементом пользовательского интерфейса платформы Astra Automation, который позволяет выполнить следующие действия:

- оценить общее состояние платформы;
- проанализировать активность задач и ресурсов;
- быстро получить доступ к последним изменениям в проектах, инвентарях и правилах.

Примечание

В этом разделе нельзя вносить изменения или управлять ресурсами напрямую. Он служит исключительно для просмотра и анализа данных.

На схеме представлены основные компоненты пользовательского интерфейса:



Здесь:

1. Кнопка управления отображаемыми панелями.
2. Панель **Количество ресурсов** (Resource Counts) – текущее количество основных ресурсов платформы: управляемых узлов, проектов и инвентарных списков.
3. Панель **Активность заданий** (Job Activity) – график количества выполненных заданий.
4. Панель **Задания** (Jobs) – список последних завершенных заданий.
5. Панель **Проекты** (Projects) – список последних обновленных проектов.
6. Панель **Инвентарные списки** (Inventories) – список последних обновленных инвентарных списков.
7. Панель **Активации свобод правил** (Rulebook Activations) – список последних обновленных активаций сводов правил.
8. Панель **Аудит правил** (Rule Audit) – список последних сработавших правил.

9. Панель **Среды принятия решений** (Decision Environments) – список последних обновленных сред принятия решений.

9.2 API

API шлюза Astra Automation обеспечивает программный доступ к функциям платформы, которые используются для управления остальными компонентами. Platform gateway выступает в качестве центрального сервиса аутентификации и авторизации, обрабатывая запросы к API и предоставляя доступ к общим ресурсам платформы через унифицированный интерфейс `/api/gateway/v1/`.

API поддерживает различные методы аутентификации, включая Basic Authentication и OAuth 2.0 token-based authentication с настраиваемыми сроками действия токенов и областями применения (scope), что обеспечивает безопасный программный доступ для сторонних инструментов и автоматизированных сервисов.

Примечание

Подробное формальное описание HTTP API представлено в *спецификации*.

9.2.1 Общий синтаксис

В этом документе рассматривается синтаксис запросов к API и получаемых ответов.

Структура запроса

Запрос API состоит из следующих элементов:

```
<method> <URI>[<query>]
   [<headers>]
   [{<body>}]
```

- <method> – указывает, какого рода действие нужно выполнить с ресурсом:
  - GET – получение данных об объекте (ресурсе) или данных в виде списка объектов по указанному URI.
  - POST – создание нового объекта или выполнение команды контроллера.
  - PUT и PATCH – обновление указанного объекта.
  - DELETE – удаление объекта.
- <URI> – указывает на конкретный ресурс, с которым необходимо выполнить действие. При необходимости к URI добавляют параметры запроса (query string).
- <headers> – заголовки HTTP содержат дополнительные метаданные запроса, такие как:
  - тип контента;
  - параметры авторизации;
  - язык;
  - формат.
- <body> – тело запроса содержит данные, которые нужно передать на сервер.

Например, в результате следующего запроса будет создан пользователь с именем <your_username> и паролем <your_password>:

```
POST https://<address>/api/gateway/v1/users/ HTTP/1.1
Content-Type: application/json
Authorization: Bearer <your_token>

```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
   {
      "username": "<your_username>",
      "password": "<your_password>"
   }
```

Здесь:

- POST – указывает на то, что новый объект должен быть создан на сервере.
- `https://<address>/api/gateway/v1/users/` – указывает на конкретный ресурс, к которому обращается запрос. В данном случае запрос отправляется к ресурсу `users` в рамках API.
- `HTTP/1.1` – название и версия протокола HTTP, используемые для связи между клиентом и сервером.
- `Content-Type: application/json` – заголовок, указывающий на то, что данные отправляются в формате JSON.
- `Authorization: Bearer <your_token>` – заголовок, указывающий на использование bearer token для авторизации и содержащий сам токен.

Структура ответа

Ответ на HTTP-запрос состоит из следующих частей:

```
<status_line>
   [<headers>]
   [{<body>}]
```

- <status_line> – статусная строка. Содержит название и номер версии используемого протокола, код статуса и его краткое описание. Самыми распространенными кодами статуса являются:
  - 200 – запрос успешно выполнен.
  - 201 – в результате запроса был создан новый объект.
  - 202 – запрос получен, но еще не обработан.
  - 301 – URL запрошенного ресурса был изменен навсегда. Новый URL будет указан в ответе.
  - 302 – URL запрошенного ресурса был временно изменен.
  - 400 – запрос был сформирован с ошибками.
  - 403 – нет прав доступа к запрошенному ресурсу.
  - 404 – запрошенный ресурс не существует.
  - 500 – на сервере произошла ошибка, в результате которой он не может успешно обработать запрос.
- <headers> – заголовки, содержащие сведения об ответе. Могут включать информацию о типе содержимого, кешировании, дате и времени, сервере, длине тела ответа и другие сведения.
- <body> – тело ответа, содержащее данные, отправленные от сервера к клиенту в ответ на запрос. Может содержать данные различных типов, например, текст, JSON и так далее, в зависимости от запроса и целей обмена информацией.

Пример ответа на запрос о создании пользователя:

```
HTTP/1.1 201 Created
Content-Type: application/json

   {
   "id": 9,
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
  "url": "/api/gateway/v1/users/9/",
  "related": {
     "personal_tokens": "/api/gateway/v1/users/9/personal_tokens/",
     "authorized_tokens": "/api/gateway/v1/users/9/authorized_tokens/",
     "tokens": "/api/gateway/v1/users/9/tokens/",
     "activity_stream": "/api/gateway/v1/activitystream/?content_type=1&object_id=9",
     "created_by": "/api/gateway/v1/users/2/",
     "modified_by": "/api/gateway/v1/users/2/",
     "authenticators": "/api/gateway/v1/users/9/authenticators/"
  },
  "summary_fields": {
     "modified_by": {
        "id": 2,
        "username": "admin",
        "first_name": "",
        "last_name": ""
     },
     "created_by": {
        "id": 2,
        "username": "admin",
        "first_name": "",
        "last_name": ""
     },
     "resource": {
        "ansible_id": "af1437fe-0eed-4efe-a0f1-df38aad0f66f",
        "resource_type": "shared.user"
     }
  },
  "created": "2025-11-21T15:08:41.267776Z",
  "created_by": 2,
  "modified": "2025-11-21T15:08:41.267763Z",
  "modified_by": 2,
  "username": "<your_username>",
  "email": "",
  "first_name": "",
  "last_name": "",
  "last_login": null,
  "password": "$encrypted$",
  "is_superuser": false,
  "is_platform_auditor": false,
  "managed": false,
  "last_login_results": {},
  "authenticators": [],
  "authenticator_uid": ""
  }
```


Здесь:

- `id` – идентификатор созданного пользователя;
- `type` – тип пользователя;
- `url` – идентификатор ресурса;
- `related` – связанные ссылки на другие ресурсы, такие как команды, организации, проекты, роли и другие;
- `summary_fields` – сводные поля с информацией о возможностях пользователя, в данном случае: редактирование, удаление;
- `created` – дата и время создания пользователя;
- `modified` – дата и время последней модификации пользователя;
- `username` – название учетной записи пользователя;
- `first_name` – имя пользователя;

- last_name – фамилия пользователя;
- email – электронная почта пользователя;
- is_superuser – сведения о том, является ли пользователь администратором платформы;
- is_platform_auditor – сведения о том, является ли пользователь аудитором платформы;
- last_login_results – дата последнего входа пользователя;
- authenticators – данные аутентификации пользователя.

9.2.2 Методы авторизации

API Astra Automation поддерживает авторизацию следующими способами:

- авторизация сеанса (session authorization);
- базовая авторизация (basic authorization);
- авторизация через токен OAuth 2 (OAuth 2 token authorization).

Для доступа к API рекомендуется использовать авторизацию через токен OAuth 2.

Примечание

По умолчанию без авторизации доступны следующие точки доступа API:

- /api/;
- /api/gateway/v1/login/;
- /api/gateway/v1/ui_auth/;
- /api/gateway/v1/ping/;
- /api/controller/v2/;
- /api/controller/v2/ping/;
- /api/galaxy/.

Авторизация сеанса

Авторизация сеанса используется при входе в API или пользовательский интерфейс Astra Automation. С помощью утилиты curl можно увидеть активность, которая происходит при входе в Astra Automation:

1. Получите X-CSRFToken с помощью команды:

```
curl -k -c - https://<address>/api/geteway/v1/login/
```

Пример вывода:

```
192.168.56.11   FALSE   /       TRUE    0       csrftoken       ␣
↪n97Hk5MjncJ45qHYmPFoqx8dLy8MPQMtfOafEZZlO9oPTJIztoD8qXRVuJY1OVsQ
```

Здесь:

- FALSE – указывает, что cookie не требует безопасного соединения (HTTPS);
- / – URI, к которому относится установленный cookie;
- TRUE – указывает, что cookie доступен для всех поддоменов этого домена;
- 0 – срок действия cookie (в секундах) отсутствует, поэтому cookie действителен только в рамках текущей сессии;
- csrftoken – название cookie;

- n97Hk5MjncJ45qHYmPFoqx8dLy8MPQMtfOafEZZlO9oPTJIztoD8qXRVuJY1OVsQ – CSRF-токен.

2. Выполните запрос авторизации:

```
curl -X POST -H 'Content-Type: application/x-www-form-urlencoded' \
     --referer https://<address>/api/geteway/v1/login/ \
     -H 'X-CSRFToken: <your_CSRFToken>' \
     --data 'username=<username>&password=<password>' \
     --cookie 'csrftoken=<your_CSRFToken>' \
     https://<address>/api/geteway/v1/login/ -k -D - -o /dev/null
```

Здесь:

- <your_CSRFToken> – CSRF-токен, полученный ранее;
- <username> – название учетной записи пользователя;
- <password> – пароль.

Если авторизация прошла успешно, сервер возвращает заголовок Set-Cookie, в котором создает cookie с именем gateway_sessionid. Значение этого cookie является идентификатором сессии (<session_ID>).

Пример ответа:

```
server: envoy
date: Wed, 19 Nov 2025 13:59:11 GMT
content-type: text/html; charset=utf-8
content-length: 0
location: /api/gateway/v1/
expires: Wed, 19 Nov 2025 13:59:11 GMT
cache-control: max-age=0, no-cache, no-store, must-revalidate, private
vary: Cookie, Accept-Language
x-frame-options: DENY
content-language: en
x-content-type-options: nosniff
referrer-policy: same-origin
cross-origin-opener-policy: same-origin
set-cookie: csrftoken=9C9myzRBzbGZoTdEPdgtAElJJjBsddBW; expires=Wed, 18 Nov 2026␣
↪13:59:11 G
set-cookie: gateway_sessionid=zxtqte7xgpj7hsj9xbfui98pbg7sae52; expires=Wed, 19 Nov␣
↪2025 14
strict-transport-security: max-age=63072000
x-frame-options: DENY
x-envoy-upstream-service-time: 304
```

После успешной авторизации можно использовать cookie gateway_sessionid при обращении к API. Пример использования:

```
curl -X GET --cookie "gateway_sessionid=<session_ID>" https://<address>/api/
↪<component>/<api_endpoint> -k -L | jq
```

Здесь:

- <session_ID> – идентификатор сеанса, полученный ранее.
- <component> – компонент к которому необходимо обратиться, например, gateway. API поддерживает следующие компоненты:
  - шлюз – gateway;
  - Automation Controller – controller;
  - Private Automation Hub – galaxy;
  - Event-Driven Automation – eda.

- <api_endpoint> – точка доступа API (URI), к которой делается запрос, например, v1/users/. Полный список поддерживаемых точек доступа смотри в спецификации конкретного компонента.

Примечание

По умолчанию сеанс длится 900 секунд. Изменить длительность сеанса можно в *графическом интерфейсе* с помощью настройки **Время жизни cookie сессии** (Session cookie age).

Базовая авторизация

При использовании базовой авторизации состояние не сохраняется. Поэтому название учетной записи пользователя (username) и пароль (password) в формате Base64 должны отправляться вместе с каждым запросом через заголовок авторизации. Это применимо для локальных записей и для учетных записей LDAP.

Пример запроса:

```
curl -X GET --user '<username>:<password>' https://<address>/api/gateway/v1/users/ -k -L␣
↪| jq
```

В целях безопасности вы можете отключить базовую авторизацию в графическом интерфейсе. Для этого выполните следующие действия:

1. Перейдите в окно **Шлюз платформы** (Platform gateway) выбрав на панели навигации *Настройки* ▸ *Шлюз платформы* (*Settings* ▸ *Platform gateway*).
2. Нажмите кнопку *Изменить настройки шлюза платформы* (Edit platform gateway settings).
3. Установите в поле **Базовая аутентификация шлюза включена** (Gateway basic auth enabled) значение Неактивный (Disable).
4. Нажмите кнопку *Сохранить настройки шлюза платформы* (Save platform gateway settings).

Авторизация через токен OAuth 2

Авторизация через токен OAuth 2 обычно используется при программном взаимодействии с API Astra Automation. Как и при базовой авторизации, токен OAuth 2 предоставляется с каждым запросом API через заголовок авторизации. Токены имеют настраиваемый срок действия и при необходимости могут быть отозваны администратором для одного пользователя или для всей системы. По умолчанию внешним пользователям, например, созданным с помощью единого входа, не разрешено создавать токены OAuth 2 в целях безопасности.

Чтобы разрешить внешним пользователям создавать токены OAuth 2, выполните следующие действия в графическом интерфейсе контроллера:

1. Перейдите в окно **Шлюз платформы** (Platform gateway) выбрав на панели навигации *Настройки* ▸ *Шлюз платформы* (*Settings* ▸ *Platform gateway*).
2. Нажмите кнопку *Изменить настройки шлюза платформы* (Edit platform gateway settings).
3. Установите в поле **Разрешить внешним пользователям создавать OAuth2-токены** (Allow external users to create OAuth2 tokens) значение Включенный (Enable).
4. Нажмите кнопку *Сохранить настройки шлюза платформы* (Save platform gateway settings).

Создать токен доступа OAuth 2 можно с помощью следующих средств:

- графический интерфейс;
- запрос API.

Создание токена в графическом интерфейсе

Пошаговые инструкции по управлению токенами через графический интерфейс доступны в секции *Управление токенами*.

Создание токена с помощью запроса

Создайте токен с помощью команды:

```
curl -u <username>:<password> -k -X POST https://<address>/api/gateway/v1/tokens/ | jq
```

Пример вывода команды:


```
{
   "id": 2,
   "url": "/api/gateway/v1/tokens/2/",
   "related": {
      "activity_stream": "/api/gateway/v1/activitystream/?content_type=38&object_id=2",
      "created_by": "/api/gateway/v1/users/2/",
      "modified_by": "/api/gateway/v1/users/2/",
      "user": "/api/gateway/v1/users/2/"
   },
   "summary_fields": {
      "modified_by": {
         "id": 2,
         "username": "admin",
         "first_name": "",
         "last_name": ""
      },
      "created_by": {
         "id": 2,
         "username": "admin",
         "first_name": "",
         "last_name": ""
      },
      "user": {
         "id": 2,
         "username": "admin",
         "first_name": "",
         "last_name": ""
      }
   },
   "created": "2025-11-21T13:28:35.383157Z",
   "created_by": 2,
   "modified": "2025-11-21T13:28:35.413663Z",
   "modified_by": 2,
   "expires": "3025-03-24T13:28:35.378890Z",
   "user": 2,
   "application": null,
   "description": "",
   "last_used": null,
   "scope": "write",
   "token": "<your_token>",
   "refresh_token": null
}
```


Здесь <your_token> – токен авторизации, который теперь можно использовать для выполнения запроса, например:

```
curl -k -X GET \
   -H "Content-Type: application/json" \
   -H "Authorization: Bearer <your_token>" \
   https://<address>/api/controller/v2/hosts/ | jq
```

9.2.3 Клиенты API

Для взаимодействия с контроллером по API можно использовать следующие типы клиентов:

- браузер;
- утилиту curl[95];
- специальные приложения, часто называемые REST API clients[96];
- расширения для браузера (например, Postman[97]);
- программу на любом языке программирования (например, Python[98]).

Браузер

Примечание

Доступ к API с помощью браузера возможен только после прохождения авторизации в графическом интерфейсе контроллера.

В адресной строке браузера введите `https://<address>/api/`, где <address> – *FQDN* или IP-адрес контроллера. При выборе любой точки доступа API (API endpoint) отображается страница с результатом GET-запроса к ней. Например, по адресу `https://<address>/api/gateway/v1/` вы увидите результат запроса на получение списка всех доступных точек доступа шлюза. А по адресу `https://<address>/api/gateway/v1/ping/` вы увидите результат выполнения команды `GET /api/gateway/v1/ping/`, которая используется для проверки доступности сервера через API.

Утилита curl

Чтобы улучшить формат вывода JSON, в дополнение к утилите `curl` установите утилиту `jq`, с помощью команды:

```
sudo apt install jq
```

Для составления curl-запроса используйте шаблон:

```
curl -X GET --user <username>:<password> https://<address>/api/<component>/<api_endpoint>
↪ -k -s | jq .
```

Здесь:

- `-X GET` – метод HTTP, который вы хотите использовать в своем запросе. GET используется для запроса данных с сервера.
- `--user <username>:<password>` – ключ, в котором необходимо указать название учетной записи пользователя и пароль для аутентификации на сервере.
- `https://<address>/api/<component>/<api_endpoint>` – URI, на который отправляется запрос.
- <address> – FQDN или IP-адрес контроллера.
- <api_endpoint> – точка доступа API (URI), к которой делается запрос.
- `-k` или `--insecure` – флаг, позволяющий утилите `curl` игнорировать проверки сертификата SSL. Это может быть полезно для тестов или в случаях, когда используются самоподписанные сертификаты.
- `-s` или `--silent` – флаг, отключающий индикаторы прогресса и сообщения об ошибках.

[95] https://curl.se/
[96] https://hevodata.com/learn/rest-clients/
[97] https://www.postman.com/
[98] https://www.python.org/

- `jq` – утилита для фильтрации и преобразования данных JSON.

Примеры использования curl

Обращение к корневой точке доступа API:

```
curl -X GET -k https://<address>/api/ | jq .
```

Пример вывода команды:

```
{
   "description": "AAP gateway REST API",
   "apis": {
      "gateway": "/api/gateway/",
      "controller": "/api/controller/",
      "eda": "/api/eda/",
      "galaxy": "/api/galaxy/"
   }
}
```

Получение списка всех точек доступа шлюза:

```
curl -X GET -k https://<address>/api/gateway/v1/ | jq .
```

Пример вывода команды:

```
{
  "activitystream": "/api/gateway/v1/activitystream/",
  "applications": "/api/gateway/v1/applications/",
  "authenticator_maps": "/api/gateway/v1/authenticator_maps/",
  "authenticator_plugins": "/api/gateway/v1/authenticator_plugins/",
  "authenticator_users": "/api/gateway/v1/authenticator_users/",
  "authenticators": "/api/gateway/v1/authenticators/",
  "http_ports": "/api/gateway/v1/http_ports/",
  "legacy_auth": "/api/gateway/v1/legacy_auth/",
  "me": "/api/gateway/v1/me/",
  "organizations": "/api/gateway/v1/organizations/",
  "ping": "/api/gateway/v1/ping/",
  "role_definitions": "/api/gateway/v1/role_definitions/",
  "role_team_assignments": "/api/gateway/v1/role_team_assignments/",
  "role_user_assignments": "/api/gateway/v1/role_user_assignments/",
  "routes": "/api/gateway/v1/routes/",
  "service_clusters": "/api/gateway/v1/service_clusters/",
  "service_keys": "/api/gateway/v1/service_keys/",
  "service_nodes": "/api/gateway/v1/service_nodes/",
  "services": "/api/gateway/v1/services/",
  "session": "/api/gateway/v1/session/",
  "settings": "/api/gateway/v1/settings/",
  "status": "/api/gateway/v1/status/",
  "teams": "/api/gateway/v1/teams/",
  "tokens": "/api/gateway/v1/tokens/",
  "trigger_definition": "/api/gateway/v1/trigger_definition/",
  "ui_auth": "/api/gateway/v1/ui_auth/",
  "users": "/api/gateway/v1/users/"
}
```

Получение списка шаблонов заданий:

```
curl -X GET --user <username>:<password> -k https://<address>/api/controller/v2/job_
↪templates/ | jq .
```

Пример вывода команды:


```
{
  "count": 3,
  "next": null,
  "previous": null,
  "results": [
    {
      "id": 6,
      "type": "job_template",
      "url": "/api/controller/v2/job_templates/6/",
      "related": {
        "created_by": "/api/controller/v2/users/1/",
        "modified_by": "/api/controller/v2/users/1/",
        "labels": "/api/controller/v2/job_templates/6/labels/",
        "inventory": "/api/controller/v2/inventories/1/",
        "project": "/api/controller/v2/projects/5/",
        "organization": "/api/controller/v2/organizations/1/",
        "credentials": "/api/controller/v2/job_templates/6/credentials/",
        "last_job": "/api/controller/v2/jobs/5/",
        "jobs": "/api/controller/v2/job_templates/6/jobs/",
        "schedules": "/api/controller/v2/job_templates/6/schedules/",
        "activity_stream": "/api/controller/v2/job_templates/6/activity_stream/",
        "launch": "/api/controller/v2/job_templates/6/launch/",
        "webhook_key": "/api/controller/v2/job_templates/6/webhook_key/",
        "webhook_receiver": "",
        "notification_templates_started": "/api/controller/v2/job_templates/6/
↪notification_templates_started/",
        "notification_templates_success": "/api/controller/v2/job_templates/6/
↪notification_templates_success/",
        "notification_templates_error": "/api/controller/v2/job_templates/6/notification_
↪templates_error/",
        "access_list": "/api/controller/v2/job_templates/6/access_list/",
        "survey_spec": "/api/controller/v2/job_templates/6/survey_spec/",
        "object_roles": "/api/controller/v2/job_templates/6/object_roles/",
        "instance_groups": "/api/controller/v2/job_templates/6/instance_groups/",
        "slice_workflow_jobs": "/api/controller/v2/job_templates/6/slice_workflow_jobs/",
        "copy": "/api/controller/v2/job_templates/6/copy/"
      },
      "summary_fields": {
        "organization": {
          "id": 1,
          "name": "Default",
          "description": "The default organization for Astra Automation"
        },
        "inventory": {
          "id": 1,
          "name": "Demo Inventory",
          "description": "",
          "has_active_failures": false,
          "total_hosts": 2,
          "hosts_with_active_failures": 0,
          "total_groups": 1,
          "has_inventory_sources": false,
          "total_inventory_sources": 0,
          "inventory_sources_with_failures": 0,
          "organization_id": 1,
          "kind": ""
        },
        "project": {
          "id": 5,
          "name": "Demo Project",
          "description": "",
          "status": "successful",
          "scm_type": "git",
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
        "allow_override": false
      },
      "last_job": {
        "id": 5,
        "name": "Demo Job Template",
        "description": "",
        "finished": "2025-11-16T13:39:07.609606Z",
        "status": "successful",
        "failed": false
      },
      "last_update": {
        "id": 5,
        "name": "Demo Job Template",
        "description": "",
        "status": "successful",
        "failed": false
      },
      "created_by": {
        "id": 1,
        "username": "admin",
        "first_name": "",
        "last_name": ""
      },
      "modified_by": {
        "id": 1,
        "username": "admin",
        "first_name": "",
        "last_name": ""
      },
      "object_roles": {
        "admin_role": {
          "description": "Can manage all aspects of the job template",
          "name": "Admin",
          "id": 37
        },
        "execute_role": {
          "description": "May run the job template",
          "name": "Execute",
          "id": 38
        },
        "read_role": {
          "description": "May view settings for the job template",
          "name": "Read",
          "id": 39
        }
      },
      "user_capabilities": {
        "edit": true,
        "delete": true,
        "start": true,
        "schedule": true,
        "copy": true
      },
      "labels": {
        "count": 0,
        "results": []
      },
      "recent_jobs": [
        {
          "id": 5,
          "status": "successful",
          "finished": "2025-11-16T13:39:07.609606Z",
          "canceled_on": null,
          "type": "job"
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
        },
        {
          "id": 1,
          "status": "successful",
          "finished": "2025-11-13T10:39:26.478610Z",
          "canceled_on": null,
          "type": "job"
        }
      ],
      "credentials": [
        {
          "id": 1,
          "name": "Demo Credential",
          "description": "",
          "kind": "ssh",
          "cloud": false
        }
      ]
    },
    "created": "2025-11-12T10:21:57.002351Z",
    "modified": "2025-11-12T10:21:57.002361Z",
    "name": "Demo Job Template",
    "description": "",
    "job_type": "run",
    "inventory": 1,
    "project": 5,
    "playbook": "hello_world.yml",
    "scm_branch": "",
    "forks": 0,
    "limit": "",
    "verbosity": 0,
    "extra_vars": "",
    "job_tags": "",
    "force_handlers": false,
    "skip_tags": "",
    "start_at_task": "",
    "timeout": 0,
    "use_fact_cache": false,
    "organization": 1,
    "last_job_run": "2025-11-16T13:39:07.609606Z",
    "last_job_failed": false,
    "next_job_run": null,
    "status": "successful",
    "execution_environment": null,
    "host_config_key": "",
    "ask_scm_branch_on_launch": false,
    "ask_diff_mode_on_launch": false,
    "ask_variables_on_launch": false,
    "ask_limit_on_launch": false,
    "ask_tags_on_launch": false,
    "ask_skip_tags_on_launch": false,
    "ask_job_type_on_launch": false,
    "ask_verbosity_on_launch": false,
    "ask_inventory_on_launch": false,
    "ask_credential_on_launch": false,
    "ask_execution_environment_on_launch": false,
    "ask_labels_on_launch": false,
    "ask_forks_on_launch": false,
    "ask_job_slice_count_on_launch": false,
    "ask_timeout_on_launch": false,
    "ask_instance_groups_on_launch": false,
    "survey_enabled": false,
    "become_enabled": false,
    "diff_mode": false,
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
      "allow_simultaneous": false,
      "custom_virtualenv": null,
      "job_slice_count": 1,
      "webhook_service": "",
      "webhook_credential": null,
      "prevent_instance_group_fallback": false
    },
    {
      "id": 8,
      "type": "job_template",
      "url": "/api/controller/v2/job_templates/8/",
      "related": {
        "created_by": "/api/controller/v2/users/1/",
        "modified_by": "/api/controller/v2/users/1/",
        "labels": "/api/controller/v2/job_templates/8/labels/",
        "inventory": "/api/controller/v2/inventories/1/",
        "project": "/api/controller/v2/projects/7/",
        "organization": "/api/controller/v2/organizations/1/",
        "credentials": "/api/controller/v2/job_templates/8/credentials/",
        "last_job": "/api/controller/v2/jobs/7/",
        "execution_environment": "/api/controller/v2/execution_environments/2/",
        "jobs": "/api/controller/v2/job_templates/8/jobs/",
        "schedules": "/api/controller/v2/job_templates/8/schedules/",
        "activity_stream": "/api/controller/v2/job_templates/8/activity_stream/",
        "launch": "/api/controller/v2/job_templates/8/launch/",
        "webhook_key": "/api/controller/v2/job_templates/8/webhook_key/",
        "webhook_receiver": "",
        "notification_templates_started": "/api/controller/v2/job_templates/8/
↪notification_templates_started/",
        "notification_templates_success": "/api/controller/v2/job_templates/8/
↪notification_templates_success/",
        "notification_templates_error": "/api/controller/v2/job_templates/8/notification_
↪templates_error/",
        "access_list": "/api/controller/v2/job_templates/8/access_list/",
        "survey_spec": "/api/controller/v2/job_templates/8/survey_spec/",
        "object_roles": "/api/controller/v2/job_templates/8/object_roles/",
        "instance_groups": "/api/controller/v2/job_templates/8/instance_groups/",
        "slice_workflow_jobs": "/api/controller/v2/job_templates/8/slice_workflow_jobs/",
        "copy": "/api/controller/v2/job_templates/8/copy/"
      },
      "summary_fields": {
        "organization": {
          "id": 1,
          "name": "Default",
          "description": "The default organization for Astra Automation"
        },
        "inventory": {
          "id": 1,
          "name": "Demo Inventory",
          "description": "",
          "has_active_failures": false,
          "total_hosts": 2,
          "hosts_with_active_failures": 0,
          "total_groups": 1,
          "has_inventory_sources": false,
          "total_inventory_sources": 0,
          "inventory_sources_with_failures": 0,
          "organization_id": 1,
          "kind": ""
        },
        "execution_environment": {
          "id": 2,
          "name": "Default execution environment",
          "description": "",
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
        "image": "hub.astra-automation.ru/aa-1.2/aa-full-ee:latest"
      },
      "project": {
        "id": 7,
        "name": "NGINX",
        "description": "",
        "status": "successful",
        "scm_type": "git",
        "allow_override": false
      },
      "last_job": {
        "id": 7,
        "name": "Project example",
        "description": "",
        "finished": "2025-11-17T06:13:48.687932Z",
        "status": "successful",
        "failed": false
      },
      "last_update": {
        "id": 7,
        "name": "Project example",
        "description": "",
        "status": "successful",
        "failed": false
      },
      "created_by": {
        "id": 1,
        "username": "admin",
        "first_name": "",
        "last_name": ""
      },
      "modified_by": {
        "id": 1,
        "username": "admin",
        "first_name": "",
        "last_name": ""
      },
      "object_roles": {
        "admin_role": {
          "description": "Can manage all aspects of the job template",
          "name": "Admin",
          "id": 61
        },
        "execute_role": {
          "description": "May run the job template",
          "name": "Execute",
          "id": 62
        },
        "read_role": {
          "description": "May view settings for the job template",
          "name": "Read",
          "id": 63
        }
      },
      "user_capabilities": {
        "edit": true,
        "delete": true,
        "start": true,
        "schedule": true,
        "copy": true
      },
      "labels": {
        "count": 0,
        "results": []
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
      },
      "recent_jobs": [
        {
          "id": 7,
          "status": "successful",
          "finished": "2025-11-17T06:13:48.687932Z",
          "canceled_on": null,
          "type": "job"
        }
      ],
      "credentials": []
    },
    "created": "2025-11-17T06:12:24.272767Z",
    "modified": "2025-11-17T06:12:24.272789Z",
    "name": "Project example",
    "description": "",
    "job_type": "run",
    "inventory": 1,
    "project": 7,
    "playbook": "example.yaml",
    "scm_branch": "",
    "forks": 0,
    "limit": "",
    "verbosity": 0,
    "extra_vars": "",
    "job_tags": "",
    "force_handlers": false,
    "skip_tags": "",
    "start_at_task": "",
    "timeout": 0,
    "use_fact_cache": false,
    "organization": 1,
    "last_job_run": "2025-11-17T06:13:48.687932Z",
    "last_job_failed": false,
    "next_job_run": null,
    "status": "successful",
    "execution_environment": 2,
    "host_config_key": "",
    "ask_scm_branch_on_launch": false,
    "ask_diff_mode_on_launch": false,
    "ask_variables_on_launch": false,
    "ask_limit_on_launch": false,
    "ask_tags_on_launch": false,
    "ask_skip_tags_on_launch": false,
    "ask_job_type_on_launch": false,
    "ask_verbosity_on_launch": false,
    "ask_inventory_on_launch": false,
    "ask_credential_on_launch": false,
    "ask_execution_environment_on_launch": false,
    "ask_labels_on_launch": false,
    "ask_forks_on_launch": false,
    "ask_job_slice_count_on_launch": false,
    "ask_timeout_on_launch": false,
    "ask_instance_groups_on_launch": false,
    "survey_enabled": false,
    "become_enabled": false,
    "diff_mode": false,
    "allow_simultaneous": false,
    "custom_virtualenv": null,
    "job_slice_count": 1,
    "webhook_service": "",
    "webhook_credential": null,
    "prevent_instance_group_fallback": false
  },
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
    {
      "id": 9,
      "type": "job_template",
      "url": "/api/controller/v2/job_templates/9/",
      "related": {
        "created_by": "/api/controller/v2/users/1/",
        "modified_by": "/api/controller/v2/users/1/",
        "labels": "/api/controller/v2/job_templates/9/labels/",
        "inventory": "/api/controller/v2/inventories/1/",
        "project": "/api/controller/v2/projects/7/",
        "organization": "/api/controller/v2/organizations/1/",
        "credentials": "/api/controller/v2/job_templates/9/credentials/",
        "last_job": "/api/controller/v2/jobs/8/",
        "execution_environment": "/api/controller/v2/execution_environments/2/",
        "jobs": "/api/controller/v2/job_templates/9/jobs/",
        "schedules": "/api/controller/v2/job_templates/9/schedules/",
        "activity_stream": "/api/controller/v2/job_templates/9/activity_stream/",
        "launch": "/api/controller/v2/job_templates/9/launch/",
        "webhook_key": "/api/controller/v2/job_templates/9/webhook_key/",
        "webhook_receiver": "",
        "notification_templates_started": "/api/controller/v2/job_templates/9/
↪notification_templates_started/",
        "notification_templates_success": "/api/controller/v2/job_templates/9/
↪notification_templates_success/",
        "notification_templates_error": "/api/controller/v2/job_templates/9/notification_
↪templates_error/",
        "access_list": "/api/controller/v2/job_templates/9/access_list/",
        "survey_spec": "/api/controller/v2/job_templates/9/survey_spec/",
        "object_roles": "/api/controller/v2/job_templates/9/object_roles/",
        "instance_groups": "/api/controller/v2/job_templates/9/instance_groups/",
        "slice_workflow_jobs": "/api/controller/v2/job_templates/9/slice_workflow_jobs/",
        "copy": "/api/controller/v2/job_templates/9/copy/"
      },
      "summary_fields": {
        "organization": {
          "id": 1,
          "name": "Default",
          "description": "The default organization for Astra Automation"
        },
        "inventory": {
          "id": 1,
          "name": "Demo Inventory",
          "description": "",
          "has_active_failures": false,
          "total_hosts": 2,
          "hosts_with_active_failures": 0,
          "total_groups": 1,
          "has_inventory_sources": false,
          "total_inventory_sources": 0,
          "inventory_sources_with_failures": 0,
          "organization_id": 1,
          "kind": ""
        },
        "execution_environment": {
          "id": 2,
          "name": "Default execution environment",
          "description": "",
          "image": "hub.astra-automation.ru/aa-1.2/aa-full-ee:latest"
        },
        "project": {
          "id": 7,
          "name": "NGINX",
          "description": "",
          "status": "successful",
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
        "scm_type": "git",
        "allow_override": false
      },
      "last_job": {
        "id": 8,
        "name": "Web project example template",
        "description": "",
        "finished": "2025-11-17T07:12:10.043589Z",
        "status": "successful",
        "failed": false
      },
      "last_update": {
        "id": 8,
        "name": "Web project example template",
        "description": "",
        "status": "successful",
        "failed": false
      },
      "created_by": {
        "id": 1,
        "username": "admin",
        "first_name": "",
        "last_name": ""
      },
      "modified_by": {
        "id": 1,
        "username": "admin",
        "first_name": "",
        "last_name": ""
      },
      "object_roles": {
        "admin_role": {
          "description": "Can manage all aspects of the job template",
          "name": "Admin",
          "id": 64
        },
        "execute_role": {
          "description": "May run the job template",
          "name": "Execute",
          "id": 65
        },
        "read_role": {
          "description": "May view settings for the job template",
          "name": "Read",
          "id": 66
        }
      },
      "user_capabilities": {
        "edit": true,
        "delete": true,
        "start": true,
        "schedule": true,
        "copy": true
      },
      "labels": {
        "count": 0,
        "results": []
      },
      "recent_jobs": [
        {
          "id": 8,
          "status": "successful",
          "finished": "2025-11-17T07:12:10.043589Z",
          "canceled_on": null,
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
            "type": "job"
          }
        ],
        "credentials": []
      },
      "created": "2025-11-17T07:10:34.194253Z",
      "modified": "2025-11-17T07:10:34.194283Z",
      "name": "Web project example template",
      "description": "",
      "job_type": "run",
      "inventory": 1,
      "project": 7,
      "playbook": "example.yaml",
      "scm_branch": "",
      "forks": 0,
      "limit": "",
      "verbosity": 0,
      "extra_vars": "",
      "job_tags": "",
      "force_handlers": false,
      "skip_tags": "",
      "start_at_task": "",
      "timeout": 0,
      "use_fact_cache": false,
      "organization": 1,
      "last_job_run": "2025-11-17T07:12:10.043589Z",
      "last_job_failed": false,
      "next_job_run": null,
      "status": "successful",
      "execution_environment": 2,
      "host_config_key": "",
      "ask_scm_branch_on_launch": false,
      "ask_diff_mode_on_launch": false,
      "ask_variables_on_launch": false,
      "ask_limit_on_launch": false,
      "ask_tags_on_launch": false,
      "ask_skip_tags_on_launch": false,
      "ask_job_type_on_launch": false,
      "ask_verbosity_on_launch": false,
      "ask_inventory_on_launch": false,
      "ask_credential_on_launch": false,
      "ask_execution_environment_on_launch": false,
      "ask_labels_on_launch": false,
      "ask_forks_on_launch": false,
      "ask_job_slice_count_on_launch": false,
      "ask_timeout_on_launch": false,
      "ask_instance_groups_on_launch": false,
      "survey_enabled": false,
      "become_enabled": false,
      "diff_mode": false,
      "allow_simultaneous": false,
      "custom_virtualenv": null,
      "job_slice_count": 1,
      "webhook_service": "",
      "webhook_credential": null,
      "prevent_instance_group_fallback": false
    }
  ]
}
```


Запуск задания из шаблона:

```
curl -X POST --user <username>:<password> -k https://<address>/api/controller/v2/job_
↪templates/6/launch/ | jq .
```

Пример вывода команды:


```
{
  "job": 12,
  "ignored_fields": {},
  "id": 12,
  "type": "job",
  "url": "/api/controller/v2/jobs/12/",
  "related": {
    "created_by": "/api/controller/v2/users/1/",
    "modified_by": "/api/controller/v2/users/1/",
    "labels": "/api/controller/v2/jobs/12/labels/",
    "inventory": "/api/controller/v2/inventories/1/",
    "project": "/api/controller/v2/projects/5/",
    "organization": "/api/controller/v2/organizations/1/",
    "credentials": "/api/controller/v2/jobs/12/credentials/",
    "unified_job_template": "/api/controller/v2/job_templates/6/",
    "stdout": "/api/controller/v2/jobs/12/stdout/",
    "job_events": "/api/controller/v2/jobs/12/job_events/",
    "job_host_summaries": "/api/controller/v2/jobs/12/job_host_summaries/",
    "activity_stream": "/api/controller/v2/jobs/12/activity_stream/",
    "notifications": "/api/controller/v2/jobs/12/notifications/",
    "create_schedule": "/api/controller/v2/jobs/12/create_schedule/",
    "job_template": "/api/controller/v2/job_templates/6/",
    "cancel": "/api/controller/v2/jobs/12/cancel/",
    "relaunch": "/api/controller/v2/jobs/12/relaunch/"
  },
  "summary_fields": {
    "organization": {
      "id": 1,
      "name": "Default",
      "description": "The default organization for Astra Automation"
    },
    "inventory": {
      "id": 1,
      "name": "Demo Inventory",
      "description": "",
      "has_active_failures": true,
      "total_hosts": 2,
      "hosts_with_active_failures": 1,
      "total_groups": 1,
      "has_inventory_sources": false,
      "total_inventory_sources": 0,
      "inventory_sources_with_failures": 0,
      "organization_id": 1,
      "kind": ""
    },
    "project": {
      "id": 5,
      "name": "Demo Project",
      "description": "",
      "status": "successful",
      "scm_type": "git",
      "allow_override": false
    },
    "job_template": {
      "id": 6,
      "name": "Demo Job Template",
      "description": ""
    },
    "unified_job_template": {
      "id": 6,
      "name": "Demo Job Template",
      "description": "",
      "unified_job_type": "job"
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
    },
    "created_by": {
      "id": 1,
      "username": "admin",
      "first_name": "",
      "last_name": ""
    },
    "modified_by": {
      "id": 1,
      "username": "admin",
      "first_name": "",
      "last_name": ""
    },
    "user_capabilities": {
      "delete": true,
      "start": true
    },
    "labels": {
      "count": 0,
      "results": []
    },
    "credentials": [
      {
        "id": 1,
        "name": "Demo Credential",
        "description": "",
        "kind": "ssh",
        "cloud": false
      }
    ]
  },
  "created": "2025-11-21T14:30:06.495914Z",
  "modified": "2025-11-21T14:30:06.747492Z",
  "name": "Demo Job Template",
  "description": "",
  "job_type": "run",
  "inventory": 1,
  "project": 5,
  "playbook": "hello_world.yml",
  "scm_branch": "",
  "forks": 0,
  "limit": "",
  "verbosity": 0,
  "extra_vars": "{}",
  "job_tags": "",
  "force_handlers": false,
  "skip_tags": "",
  "start_at_task": "",
  "timeout": 0,
  "use_fact_cache": false,
  "organization": 1,
  "unified_job_template": 6,
  "launch_type": "manual",
  "status": "pending",
  "execution_environment": null,
  "failed": false,
  "started": null,
  "finished": null,
  "canceled_on": null,
  "elapsed": 0,
  "job_args": "",
  "job_cwd": "",
  "job_env": {},
  "job_explanation": "",
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
  "execution_node": "",
  "controller_node": "",
  "result_traceback": "",
  "event_processing_finished": false,
  "launched_by": {
    "id": 1,
    "name": "admin",
    "type": "user",
    "url": "/api/controller/v2/users/1/"
  },
  "work_unit_id": null,
  "job_template": 6,
  "passwords_needed_to_start": [],
  "allow_simultaneous": false,
  "artifacts": {},
  "scm_revision": "",
  "instance_group": null,
  "diff_mode": false,
  "job_slice_number": 0,
  "job_slice_count": 1,
  "webhook_service": "",
  "webhook_credential": null,
  "webhook_guid": ""
}
```


Получение токена авторизации:

```
curl -k -X POST \
https://<address>/api/gateway/v1/tokens/ \
-H 'Content-Type: application/json' \
-d '{
      "description": "Token description",
      "application": null,
      "user": "admin",
      "scope": "write"
   }' \
-u '<username>:<password>' | jq .
```

Пример вывода команды:


```
{
  "id": 3,
  "url": "/api/gateway/v1/tokens/3/",
  "related": {
    "activity_stream": "/api/gateway/v1/activitystream/?content_type=38&object_id=3",
    "created_by": "/api/gateway/v1/users/2/",
    "modified_by": "/api/gateway/v1/users/2/",
    "user": "/api/gateway/v1/users/2/"
  },
  "summary_fields": {
    "modified_by": {
      "id": 2,
      "username": "admin",
      "first_name": "",
      "last_name": ""
    },
    "created_by": {
      "id": 2,
      "username": "admin",
      "first_name": "",
      "last_name": ""
    },
    "user": {
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
      "id": 2,
      "username": "admin",
      "first_name": "",
      "last_name": ""
    }
  },
  "created": "2025-11-21T14:41:45.209414Z",
  "created_by": 2,
  "modified": "2025-11-21T14:41:45.247211Z",
  "modified_by": 2,
  "expires": "3025-03-24T14:41:45.201955Z",
  "user": 2,
  "application": null,
  "description": "Token description",
  "last_used": null,
  "scope": "write",
  "token": "<your_token>",
  "refresh_token": null
}
```


Здесь <your_token> – токен авторизации.

Проверка статуса задания, запущенного ранее:

```
curl -X GET \
https://<address>/api/controller/v2/jobs/<jobs_id>/ \
-H 'Authorization: Bearer <your_token>' \
-k -s | jq .
```

Здесь <jobs_id> – идентификатор задания, запущенного ранее.

Пример вывода команды:


```
{
  "id": 12,
  "type": "job",
  "url": "/api/controller/v2/jobs/12/",
  "related": {
    "created_by": "/api/controller/v2/users/1/",
    "labels": "/api/controller/v2/jobs/12/labels/",
    "inventory": "/api/controller/v2/inventories/1/",
    "project": "/api/controller/v2/projects/5/",
    "organization": "/api/controller/v2/organizations/1/",
    "credentials": "/api/controller/v2/jobs/12/credentials/",
    "unified_job_template": "/api/controller/v2/job_templates/6/",
    "stdout": "/api/controller/v2/jobs/12/stdout/",
    "execution_environment": "/api/controller/v2/execution_environments/2/",
    "job_events": "/api/controller/v2/jobs/12/job_events/",
    "job_host_summaries": "/api/controller/v2/jobs/12/job_host_summaries/",
    "activity_stream": "/api/controller/v2/jobs/12/activity_stream/",
    "notifications": "/api/controller/v2/jobs/12/notifications/",
    "create_schedule": "/api/controller/v2/jobs/12/create_schedule/",
    "job_template": "/api/controller/v2/job_templates/6/",
    "cancel": "/api/controller/v2/jobs/12/cancel/",
    "relaunch": "/api/controller/v2/jobs/12/relaunch/"
  },
  "summary_fields": {
    "organization": {
      "id": 1,
      "name": "Default",
      "description": "The default organization for Astra Automation"
    },
    "inventory": {
      "id": 1,
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
      "name": "Demo Inventory",
      "description": "",
      "has_active_failures": true,
      "total_hosts": 2,
      "hosts_with_active_failures": 1,
      "total_groups": 1,
      "has_inventory_sources": false,
      "total_inventory_sources": 0,
      "inventory_sources_with_failures": 0,
      "organization_id": 1,
      "kind": ""
    },
    "execution_environment": {
      "id": 2,
      "name": "Default execution environment",
      "description": "",
      "image": "hub.astra-automation.ru/aa-1.2/aa-full-ee:latest"
    },
    "project": {
      "id": 5,
      "name": "Demo Project",
      "description": "",
      "status": "successful",
      "scm_type": "git",
      "allow_override": false
    },
    "job_template": {
      "id": 6,
      "name": "Demo Job Template",
      "description": ""
    },
    "unified_job_template": {
      "id": 6,
      "name": "Demo Job Template",
      "description": "",
      "unified_job_type": "job"
    },
    "instance_group": {
      "id": 2,
      "name": "default",
      "is_container_group": false
    },
    "created_by": {
      "id": 1,
      "username": "admin",
      "first_name": "",
      "last_name": ""
    },
    "user_capabilities": {
      "delete": true,
      "start": true
    },
    "labels": {
      "count": 0,
      "results": []
    },
    "credentials": [
      {
        "id": 1,
        "name": "Demo Credential",
        "description": "",
        "kind": "ssh",
        "cloud": false
      }
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
    ]
  },
  "created": "2025-11-21T14:30:06.495914Z",
  "modified": "2025-11-21T14:30:07.027587Z",
  "name": "Demo Job Template",
  "description": "",
  "job_type": "run",
  "inventory": 1,
  "project": 5,
  "playbook": "hello_world.yml",
  "scm_branch": "",
  "forks": 0,
  "limit": "",
  "verbosity": 0,
  "extra_vars": "{}",
  "job_tags": "",
  "force_handlers": false,
  "skip_tags": "",
  "start_at_task": "",
  "timeout": 0,
  "use_fact_cache": false,
  "organization": 1,
  "unified_job_template": 6,
  "launch_type": "manual",
  "status": "failed",
  "execution_environment": 2,
  "failed": true,
  "started": "2025-11-21T14:30:07.202079Z",
  "finished": "2025-11-21T14:30:12.733562Z",
  "canceled_on": null,
  "elapsed": 5.531,
  "job_args": "[\"podman\", \"run\", \"--rm\", \"--tty\", \"--interactive\", \"--workdir\
↪", \"/runner/project\", \"-v\", \"/tmp/awx_12_8rgq8tz7/:/runner/:Z\", \"-v\", \"/etc/
↪ssl/certs/:/etc/ssl/certs/:O\", \"-v\", \"/usr/local/share/ca-certificates/:/usr/local/
↪share/ca-certificates/:O\", \"--env-file\", \"/tmp/awx_12_8rgq8tz7/artifacts/12/env.
↪list\", \"--quiet\", \"--name\", \"ansible_runner_12\", \"--user=root\", \"--network\",
↪ \"slirp4netns:enable_ipv6=true\", \"hub.astra-automation.ru/aa-1.2/aa-full-ee:latest\
↪", \"ansible-playbook\", \"-u\", \"admin\", \"-i\", \"/runner/inventory\", \"-e\", \"@/
↪runner/env/extravars\", \"hello_world.yml\"]",
  "job_cwd": "/runner/project",
  "job_env": {
    "JOB_ID": "12",
    "AWX_HOST": "https://10.205.212.89",
    "INVENTORY_ID": "1",
    "MAX_EVENT_RES": "700000",
    "PROJECT_REVISION": "347e44fea036c94d5f60e544de006453ee5c71ad",
    "ANSIBLE_ROLES_PATH": "/runner/requirements_roles:~/.ansible/roles:/usr/share/
↪ansible/roles:/etc/ansible/roles",
    "RUNNER_OMIT_EVENTS": "False",
    "ANSIBLE_FORCE_COLOR": "True",
    "AWX_PRIVATE_DATA_DIR": "/tmp/awx_12_8rgq8tz7",
    "ANSIBLE_UNSAFE_WRITES": "1",
    "AWX_ISOLATED_DATA_DIR": "/runner/artifacts/12",
    "ANSIBLE_STDOUT_CALLBACK": "awx_display",
    "ANSIBLE_CALLBACK_PLUGINS": "/runner/artifacts/12/callback",
    "ANSIBLE_COLLECTIONS_PATH": "/runner/requirements_collections:~/.ansible/
↪collections:/usr/share/ansible/collections",
    "ANSIBLE_HOST_KEY_CHECKING": "False",
    "RUNNER_ONLY_FAILED_EVENTS": "False",
    "ANSIBLE_RETRY_FILES_ENABLED": "False",
    "ANSIBLE_SSH_CONTROL_PATH_DIR": "/runner/cp",
    "ANSIBLE_STANDARD_SETTINGS_FILES": "[PosixPath('/etc/ansible-automation-platform/
↪settings.yaml'), PosixPath('/etc/ansible-automation-platform/flags.yaml'), PosixPath('/
↪etc/ansible-automation-platform/.secrets.yaml'), PosixPath('/etc/ansible-automation-
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
↪platform/awx/settings.yaml'), PosixPath('/etc/ansible-automation-platform/awx/flags.
↪yaml'), PosixPath('/etc/ansible-automation-platform/awx/.secrets.yaml')]",
    "ANSIBLE_INVENTORY_UNPARSED_FAILED": "True",
    "ANSIBLE_PARAMIKO_RECORD_HOST_KEYS": "False"
  },
  "job_explanation": "",
  "execution_node": "10.205.212.60",
  "controller_node": "10.205.212.90",
  "result_traceback": "",
  "event_processing_finished": true,
  "launched_by": {
    "id": 1,
    "name": "admin",
    "type": "user",
    "url": "/api/controller/v2/users/1/"
  },
  "work_unit_id": "10205212902HUMxyGa",
  "job_template": 6,
  "passwords_needed_to_start": [],
  "allow_simultaneous": false,
  "artifacts": {},
  "scm_revision": "347e44fea036c94d5f60e544de006453ee5c71ad",
  "instance_group": 2,
  "diff_mode": false,
  "job_slice_number": 0,
  "job_slice_count": 1,
  "webhook_service": "",
  "webhook_credential": null,
  "webhook_guid": "",
  "host_status_counts": {
    "ok": 1,
    "dark": 1
  },
  "playbook_counts": {
    "play_count": 1,
    "task_count": 2
  },
  "custom_virtualenv": null
}
```


Создание проекта:

```
curl -X POST \
https://<address>/api/controller/v2/projects/ \
-H 'Authorization: Bearer <your_token>' \
-H 'Content-Type: application/json' \
-d '{
   "name": "New Project",
   "organization": 1,
   "scm_type": "git",
   "scm_url": "https://github.com/ansible/ansible-tower-samples"
}' \
-k -s | jq .
```

Пример вывода команды:


```
{
  "id": 10,
  "type": "project",
  "url": "/api/controller/v2/projects/10/",
  "related": {
    "named_url": "/api/controller/v2/projects/New Project++Default/",
    "created_by": "/api/controller/v2/users/1/",
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
    "modified_by": "/api/controller/v2/users/1/",
    "current_job": "/api/controller/v2/project_updates/13/",
    "teams": "/api/controller/v2/projects/10/teams/",
    "playbooks": "/api/controller/v2/projects/10/playbooks/",
    "inventory_files": "/api/controller/v2/projects/10/inventories/",
    "update": "/api/controller/v2/projects/10/update/",
    "project_updates": "/api/controller/v2/projects/10/project_updates/",
    "scm_inventory_sources": "/api/controller/v2/projects/10/scm_inventory_sources/",
    "schedules": "/api/controller/v2/projects/10/schedules/",
    "activity_stream": "/api/controller/v2/projects/10/activity_stream/",
    "notification_templates_started": "/api/controller/v2/projects/10/notification_
↪templates_started/",
    "notification_templates_success": "/api/controller/v2/projects/10/notification_
↪templates_success/",
    "notification_templates_error": "/api/controller/v2/projects/10/notification_
↪templates_error/",
    "access_list": "/api/controller/v2/projects/10/access_list/",
    "object_roles": "/api/controller/v2/projects/10/object_roles/",
    "copy": "/api/controller/v2/projects/10/copy/",
    "organization": "/api/controller/v2/organizations/1/",
    "current_update": "/api/controller/v2/project_updates/13/"
  },
  "summary_fields": {
    "organization": {
      "id": 1,
      "name": "Default",
      "description": "The default organization for Astra Automation"
    },
    "current_update": {
      "id": 13,
      "name": "New Project",
      "description": "",
      "status": "pending",
      "failed": false
    },
    "current_job": {
      "id": 13,
      "name": "New Project",
      "description": "",
      "status": "pending",
      "failed": false
    },
    "created_by": {
      "id": 1,
      "username": "admin",
      "first_name": "",
      "last_name": ""
    },
    "modified_by": {
      "id": 1,
      "username": "admin",
      "first_name": "",
      "last_name": ""
    },
    "object_roles": {
      "admin_role": {
        "description": "Can manage all aspects of the project",
        "name": "Admin",
        "id": 99
      },
      "use_role": {
        "description": "Can use the project in a job template",
        "name": "Use",
        "id": 100
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
      },
      "update_role": {
        "description": "May update the project",
        "name": "Update",
        "id": 101
      },
      "read_role": {
        "description": "May view settings for the project",
        "name": "Read",
        "id": 102
      }
    },
    "user_capabilities": {
      "edit": true,
      "delete": true,
      "start": true,
      "schedule": true,
      "copy": true
    }
  },
  "created": "2025-11-21T14:49:52.871576Z",
  "modified": "2025-11-21T14:49:52.871603Z",
  "name": "New Project",
  "description": "",
  "local_path": "_10__new_project",
  "scm_type": "git",
  "scm_url": "https://github.com/ansible/ansible-tower-samples",
  "scm_branch": "",
  "scm_refspec": "",
  "scm_clean": false,
  "scm_track_submodules": false,
  "scm_delete_on_update": false,
  "credential": null,
  "timeout": 0,
  "scm_revision": "",
  "last_job_run": null,
  "last_job_failed": false,
  "next_job_run": null,
  "status": "pending",
  "organization": 1,
  "scm_update_on_launch": false,
  "scm_update_cache_timeout": 0,
  "allow_override": false,
  "custom_virtualenv": null,
  "default_environment": null,
  "signature_validation_credential": null,
  "last_update_failed": false,
  "last_updated": null
}
```


Изменение проекта, созданного ранее:

```
curl -X PATCH \
https://<address>/api/controller/v2/projects/<project_id>/ \
-H 'Authorization: Bearer <your_token>' \
-H 'Content-Type: application/json' \
-d '{
   "name": "Updated Project",
   "scm_url": "https://github.com/ansible/ansible-examples"
}' \
-k -s | jq .
```

Здесь <project_id> – идентификатор проекта, созданного ранее.

Пример вывода команды:


```
{
  "id": 10,
  "type": "project",
  "url": "/api/controller/v2/projects/10/",
  "related": {
    "named_url": "/api/controller/v2/projects/Updated Project++Default/",
    "created_by": "/api/controller/v2/users/1/",
    "modified_by": "/api/controller/v2/users/1/",
    "current_job": "/api/controller/v2/project_updates/14/",
    "last_job": "/api/controller/v2/project_updates/13/",
    "teams": "/api/controller/v2/projects/10/teams/",
    "playbooks": "/api/controller/v2/projects/10/playbooks/",
    "inventory_files": "/api/controller/v2/projects/10/inventories/",
    "update": "/api/controller/v2/projects/10/update/",
    "project_updates": "/api/controller/v2/projects/10/project_updates/",
    "scm_inventory_sources": "/api/controller/v2/projects/10/scm_inventory_sources/",
    "schedules": "/api/controller/v2/projects/10/schedules/",
    "activity_stream": "/api/controller/v2/projects/10/activity_stream/",
    "notification_templates_started": "/api/controller/v2/projects/10/notification_
↪templates_started/",
    "notification_templates_success": "/api/controller/v2/projects/10/notification_
↪templates_success/",
    "notification_templates_error": "/api/controller/v2/projects/10/notification_
↪templates_error/",
    "access_list": "/api/controller/v2/projects/10/access_list/",
    "object_roles": "/api/controller/v2/projects/10/object_roles/",
    "copy": "/api/controller/v2/projects/10/copy/",
    "organization": "/api/controller/v2/organizations/1/",
    "current_update": "/api/controller/v2/project_updates/14/",
    "last_update": "/api/controller/v2/project_updates/13/"
  },
  "summary_fields": {
    "organization": {
      "id": 1,
      "name": "Default",
      "description": "The default organization for Astra Automation"
    },
    "last_job": {
      "id": 13,
      "name": "New Project",
      "description": "",
      "finished": "2025-11-21T14:50:01.025151Z",
      "status": "successful",
      "failed": false
    },
    "last_update": {
      "id": 13,
      "name": "New Project",
      "description": "",
      "status": "successful",
      "failed": false
    },
    "current_update": {
      "id": 14,
      "name": "Updated Project",
      "description": "",
      "status": "pending",
      "failed": false
    },
    "current_job": {
      "id": 14,
      "name": "Updated Project",
      "description": "",
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
      "status": "pending",
      "failed": false
    },
    "created_by": {
      "id": 1,
      "username": "admin",
      "first_name": "",
      "last_name": ""
    },
    "modified_by": {
      "id": 1,
      "username": "admin",
      "first_name": "",
      "last_name": ""
    },
    "object_roles": {
      "admin_role": {
        "description": "Can manage all aspects of the project",
        "name": "Admin",
        "id": 99
      },
      "use_role": {
        "description": "Can use the project in a job template",
        "name": "Use",
        "id": 100
      },
      "update_role": {
        "description": "May update the project",
        "name": "Update",
        "id": 101
      },
      "read_role": {
        "description": "May view settings for the project",
        "name": "Read",
        "id": 102
      }
    },
    "user_capabilities": {
      "edit": true,
      "delete": true,
      "start": true,
      "schedule": true,
      "copy": true
    },
    "resolved_environment": {
      "id": 2,
      "name": "Default execution environment",
      "description": "",
      "image": "hub.astra-automation.ru/aa-1.2/aa-full-ee:latest"
    }
  },
  "created": "2025-11-21T14:49:52.871576Z",
  "modified": "2025-11-21T14:55:12.985646Z",
  "name": "Updated Project",
  "description": "",
  "local_path": "_10__new_project",
  "scm_type": "git",
  "scm_url": "https://github.com/ansible/ansible-examples",
  "scm_branch": "",
  "scm_refspec": "",
  "scm_clean": false,
  "scm_track_submodules": false,
  "scm_delete_on_update": false,
  "credential": null,
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
  "timeout": 0,
  "scm_revision": "347e44fea036c94d5f60e544de006453ee5c71ad",
  "last_job_run": "2025-11-21T14:50:01.025151Z",
  "last_job_failed": false,
  "next_job_run": null,
  "status": "pending",
  "organization": 1,
  "scm_update_on_launch": false,
  "scm_update_cache_timeout": 0,
  "allow_override": false,
  "custom_virtualenv": null,
  "default_environment": null,
  "signature_validation_credential": null,
  "last_update_failed": false,
  "last_updated": "2025-11-21T14:50:01.025151Z"
}
```


Удаление проекта:

```
curl -k -X DELETE \
https://<address>/api/controller/v2/projects/<project_id>/ \
-H 'Authorization: Bearer <your-token>'
```

Расширения для браузера

Применение расширения для браузера Google Chrome выглядит следующим образом:

1. Откройте Chrome Web Store и установите расширение Postman на ваш браузер.
2. Запустите Postman из списка расширений в вашем браузере.
3. Создайте новый запрос.
4. Введите URI объекта, который вы хотите запросить, и выберите нужный метод HTTP (GET, POST, PUT, PATCH, DELETE).
5. Отправьте запрос.
6. После отправки запроса вы увидите ответ от сервера в формате JSON, XML или другом формате. Вы также можете просмотреть статус-код ответа и другие детали.

Программа

Рассмотрим в качестве примера скрипт Python, который выполняет следующие действия в контроллере:

1. Создает новый проект.
2. Получает информацию о всех проектах.
3. Получает информацию о созданном проекте.
4. Обновляет проект из источника.
5. Проверяет статус обновления проекта.
6. Изменяет проект.
7. Получает информацию о созданном проекте после изменения.
8. Удаляет проект.
9. Получает информацию о всех проектах.
10. Получает конфигурацию Automation Controller.

После команд, запускающих какое-либо действие на сервере, добавлена тридцатисекундная пауза для наглядности выполнения команд.

Для запуска скрипта необходимо:

- Использовать Python не ниже версии 3.8.
- Установить пакет requests.

```
"""Demo scenario"""

import json
import time
import requests

# Define base URL and headers
gateway_url = "https://<address>/api/gateway/v1/"
controller_url = "https://<address>/api/controller/v2/"
headers = {"Authorization": "Bearer <your_token>", "Content-type": "application/json"}


def button_press():
    """Prompt the user to press Enter to continue."""
    input("\nНажмите Enter чтобы продолжить...\n")


def print_json(response):
    """
    Print the response JSON returned by a REST API request.

    Args:
        response: The response object returned by the REST API request
    """
    if response.text:
        print(json.dumps(response.json(), indent=4))
    else:
        print("Status code:", response.status_code)


# Define the project data
project_data = {
    "name": "New Project",
    "organization": 1,
    "scm_type": "git",
    "scm_url": "https://github.com/ansible/ansible-tower-samples",
}

# Post new project
response_post_new_project = requests.post(
    f"{controller_url}projects/",
    headers=headers,
    data=json.dumps(project_data),
    verify=False,
    timeout=10,
)
print("\n--- Создание проекта ---")
print_json(response_post_new_project)
project_id = response_post_new_project.json()["id"]
time.sleep(30)
print("\n --- Проект создан ---")
button_press()

# Get all projects
response_get_all_projects = requests.get(
    f"{controller_url}projects/", headers=headers, verify=False, timeout=10
)
print("\n--- Информация о проектах ---")
print_json(response_get_all_projects)
print("\n --- Выведена информация по всем проектам ---")
button_press()
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
# Get project details
response_get_project_detail = requests.get(
    f"{controller_url}projects/{project_id}/", headers=headers, verify=False, timeout=10
)
print("\n--- Получение деталей проекта ---")
print_json(response_get_project_detail)
print("\n --- Выведена информация о созданном проекте ---")
button_press()

# Update project from the version control source
response_update_project = requests.post(
    f"{controller_url}projects/{project_id}/update/",
    headers=headers,
    verify=False,
    timeout=10,
)
print("\n--- Обновление проекта из источника контроля версий ---")
print_json(response_update_project)
time.sleep(30)
print("\n --- Проект обновлен ---")
button_press()

# Check update status information
response_project_update_status = requests.get(
    f"{controller_url}projects/{project_id}/project_updates/",
    headers=headers,
    verify=False,
    timeout=10,
)
print("\n--- Проверка информации о статусе обновления ---")
print_json(response_project_update_status)
print("\n --- Выведена информация о статусе обновления ---")
button_press()

# Change the project
patch_data = {
    "name": "Updated Project",
    "scm_url": "https://github.com/ansible/ansible-examples",
}
response_patch_project = requests.patch(
    f"{controller_url}projects/{project_id}/",
    headers=headers,
    data=json.dumps(patch_data),
    verify=False,
    timeout=10,
)
print("\n--- Изменение проекта ---")
print_json(response_patch_project)
time.sleep(30)
print("\n --- Проект изменен ---")
button_press()

# Get project details after the change
response_get_project_detail_after_change = requests.get(
    f"{controller_url}projects/{project_id}/", headers=headers, verify=False, timeout=10
)
print("\n--- Получение деталей проекта (после обновления) ---")
print_json(response_get_project_detail_after_change)
print("\n --- Выведена информация о созданном проекте после изменения ---")
button_press()

# Delete the project
response_delete_project = requests.delete(
    f"{controller_url}projects/{project_id}/", headers=headers, verify=False, timeout=10
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
)
print("\n--- Удаление проекта ---")
print_json(response_delete_project)
print("\n --- Проект удален ---")
button_press()

# List all projects after deletion
response_get_all_projects_after_deletion = requests.get(
    f"{controller_url}projects/", headers=headers, verify=False, timeout=10
)
print("\n--- Список проектов (после удаления) ---")
print_json(response_get_all_projects_after_deletion)
print("\n --- Выведена информация по всем проектам после удаления ---")
button_press()

# Get Automation Controller Configuration
response_get_configuration = requests.get(
    f"{controller_url}config/", headers=headers, verify=False, timeout=10
)
print("\n--- Получение конфигурации Automation Controller ---")
print_json(response_get_configuration)
print("\n --- Выведена информация по серверу Automation Controller ---")
```

В демонстрационном скрипте замените значения <address> и <your_token> на свои.

Пример вывода:

Создание проекта:

```
{
  "id": 11,
  "type": "project",
  "url": "/api/controller/v2/projects/11/",
  "related": {
    "named_url": "/api/controller/v2/projects/New Project++Default/",
    "created_by": "/api/controller/v2/users/1/",
    "modified_by": "/api/controller/v2/users/1/",
    "current_job": "/api/controller/v2/project_updates/16/",
    "teams": "/api/controller/v2/projects/11/teams/",
    "playbooks": "/api/controller/v2/projects/11/playbooks/",
    "inventory_files": "/api/controller/v2/projects/11/inventories/",
    "update": "/api/controller/v2/projects/11/update/",
    "project_updates": "/api/controller/v2/projects/11/project_updates/",
    "scm_inventory_sources": "/api/controller/v2/projects/11/scm_inventory_sources/",
    "schedules": "/api/controller/v2/projects/11/schedules/",
    "activity_stream": "/api/controller/v2/projects/11/activity_stream/",
    "notification_templates_started": "/api/controller/v2/projects/11/notification_
↪templates_started/",
    "notification_templates_success": "/api/controller/v2/projects/11/notification_
↪templates_success/",
    "notification_templates_error": "/api/controller/v2/projects/11/notification_
↪templates_error/",
    "access_list": "/api/controller/v2/projects/11/access_list/",
    "object_roles": "/api/controller/v2/projects/11/object_roles/",
    "copy": "/api/controller/v2/projects/11/copy/",
    "organization": "/api/controller/v2/organizations/1/",
    "current_update": "/api/controller/v2/project_updates/16/"
  },
  "summary_fields": {
    "organization": {
      "id": 1,
      "name": "Default",
      "description": "The default organization for Astra Automation"
    },
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
    "current_update": {
      "id": 16,
      "name": "New Project",
      "description": "",
      "status": "pending",
      "failed": false
    },
    "current_job": {
      "id": 16,
      "name": "New Project",
      "description": "",
      "status": "pending",
      "failed": false
    },
    "created_by": {
      "id": 1,
      "username": "admin",
      "first_name": "",
      "last_name": ""
    },
    "modified_by": {
      "id": 1,
      "username": "admin",
      "first_name": "",
      "last_name": ""
    },
    "object_roles": {
      "admin_role": {
        "description": "Can manage all aspects of the project",
        "name": "Admin",
        "id": 106
      },
      "use_role": {
        "description": "Can use the project in a job template",
        "name": "Use",
        "id": 107
      },
      "update_role": {
        "description": "May update the project",
        "name": "Update",
        "id": 108
      },
      "read_role": {
        "description": "May view settings for the project",
        "name": "Read",
        "id": 109
      }
    },
    "user_capabilities": {
      "edit": true,
      "delete": true,
      "start": true,
      "schedule": true,
      "copy": true
    }
  },
  "created": "2025-11-25T08:21:51.600418Z",
  "modified": "2025-11-25T08:21:51.600456Z",
  "name": "New Project",
  "description": "",
  "local_path": "_11__new_project",
  "scm_type": "git",
  "scm_url": "https://github.com/ansible/ansible-tower-samples",
  "scm_branch": "",
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
  "scm_refspec": "",
  "scm_clean": false,
  "scm_track_submodules": false,
  "scm_delete_on_update": false,
  "credential": null,
  "timeout": 0,
  "scm_revision": "",
  "last_job_run": null,
  "last_job_failed": false,
  "next_job_run": null,
  "status": "pending",
  "organization": 1,
  "scm_update_on_launch": false,
  "scm_update_cache_timeout": 0,
  "allow_override": false,
  "custom_virtualenv": null,
  "default_environment": null,
  "signature_validation_credential": null,
  "last_update_failed": false,
  "last_updated": null
}
```


Получение информации о проектах:


```
{
  "count": 3,
  "next": null,
  "previous": null,
  "results": [
    {
      "id": 5,
      "type": "project",
      "url": "/api/controller/v2/projects/5/",
      "related": {
        "created_by": "/api/controller/v2/users/1/",
        "modified_by": "/api/controller/v2/users/1/",
        "teams": "/api/controller/v2/projects/5/teams/",
        "playbooks": "/api/controller/v2/projects/5/playbooks/",
        "inventory_files": "/api/controller/v2/projects/5/inventories/",
        "update": "/api/controller/v2/projects/5/update/",
        "project_updates": "/api/controller/v2/projects/5/project_updates/",
        "scm_inventory_sources": "/api/controller/v2/projects/5/scm_inventory_sources/",
        "schedules": "/api/controller/v2/projects/5/schedules/",
        "activity_stream": "/api/controller/v2/projects/5/activity_stream/",
        "notification_templates_started": "/api/controller/v2/projects/5/notification_
↪templates_started/",
        "notification_templates_success": "/api/controller/v2/projects/5/notification_
↪templates_success/",
        "notification_templates_error": "/api/controller/v2/projects/5/notification_
↪templates_error/",
        "access_list": "/api/controller/v2/projects/5/access_list/",
        "object_roles": "/api/controller/v2/projects/5/object_roles/",
        "copy": "/api/controller/v2/projects/5/copy/",
        "organization": "/api/controller/v2/organizations/1/"
      },
      "summary_fields": {
        "organization": {
          "id": 1,
          "name": "Default",
          "description": "The default organization for Astra Automation"
        },
        "created_by": {
          "id": 1,
          "username": "admin",
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
        "first_name": "",
        "last_name": ""
      },
      "modified_by": {
        "id": 1,
        "username": "admin",
        "first_name": "",
        "last_name": ""
      },
      "object_roles": {
        "admin_role": {
          "description": "Can manage all aspects of the project",
          "name": "Admin",
          "id": 22
        },
        "use_role": {
          "description": "Can use the project in a job template",
          "name": "Use",
          "id": 23
        },
        "update_role": {
          "description": "May update the project",
          "name": "Update",
          "id": 24
        },
        "read_role": {
          "description": "May view settings for the project",
          "name": "Read",
          "id": 25
        }
      },
      "user_capabilities": {
        "edit": true,
        "delete": true,
        "start": true,
        "schedule": true,
        "copy": true
      }
    },
    "created": "2025-11-12T10:21:56.220632Z",
    "modified": "2025-11-12T10:21:56.220653Z",
    "name": "Demo Project",
    "description": "",
    "local_path": "_5__demo_project",
    "scm_type": "git",
    "scm_url": "https://github.com/ansible/ansible-tower-samples",
    "scm_branch": "",
    "scm_refspec": "",
    "scm_clean": false,
    "scm_track_submodules": false,
    "scm_delete_on_update": false,
    "credential": null,
    "timeout": 0,
    "scm_revision": "347e44fea036c94d5f60e544de006453ee5c71ad",
    "last_job_run": null,
    "last_job_failed": false,
    "next_job_run": null,
    "status": "successful",
    "organization": 1,
    "scm_update_on_launch": false,
    "scm_update_cache_timeout": 0,
    "allow_override": false,
    "custom_virtualenv": null,
    "default_environment": null,
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
      "signature_validation_credential": null,
      "last_update_failed": false,
      "last_updated": null
    },
    {
      "id": 7,
      "type": "project",
      "url": "/api/controller/v2/projects/7/",
      "related": {
        "created_by": "/api/controller/v2/users/1/",
        "modified_by": "/api/controller/v2/users/1/",
        "credential": "/api/controller/v2/credentials/4/",
        "last_job": "/api/controller/v2/project_updates/6/",
        "teams": "/api/controller/v2/projects/7/teams/",
        "playbooks": "/api/controller/v2/projects/7/playbooks/",
        "inventory_files": "/api/controller/v2/projects/7/inventories/",
        "update": "/api/controller/v2/projects/7/update/",
        "project_updates": "/api/controller/v2/projects/7/project_updates/",
        "scm_inventory_sources": "/api/controller/v2/projects/7/scm_inventory_sources/",
        "schedules": "/api/controller/v2/projects/7/schedules/",
        "activity_stream": "/api/controller/v2/projects/7/activity_stream/",
        "notification_templates_started": "/api/controller/v2/projects/7/notification_
↪templates_started/",
        "notification_templates_success": "/api/controller/v2/projects/7/notification_
↪templates_success/",
        "notification_templates_error": "/api/controller/v2/projects/7/notification_
↪templates_error/",
        "access_list": "/api/controller/v2/projects/7/access_list/",
        "object_roles": "/api/controller/v2/projects/7/object_roles/",
        "copy": "/api/controller/v2/projects/7/copy/",
        "organization": "/api/controller/v2/organizations/1/",
        "default_environment": "/api/controller/v2/execution_environments/2/",
        "last_update": "/api/controller/v2/project_updates/6/"
      },
      "summary_fields": {
        "organization": {
          "id": 1,
          "name": "Default",
          "description": "The default organization for Astra Automation"
        },
        "default_environment": {
          "id": 2,
          "name": "Default execution environment",
          "description": "",
          "image": "hub.astra-automation.ru/aa-1.2/aa-full-ee:latest"
        },
        "credential": {
          "id": 4,
          "name": "SCM access",
          "description": "",
          "kind": "scm",
          "cloud": false,
          "kubernetes": false,
          "credential_type_id": 8
        },
        "last_job": {
          "id": 6,
          "name": "NGINX",
          "description": "",
          "finished": "2025-11-17T06:03:37.360186Z",
          "status": "successful",
          "failed": false
        },
        "last_update": {
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
        "id": 6,
        "name": "NGINX",
        "description": "",
        "status": "successful",
        "failed": false
      },
      "created_by": {
        "id": 1,
        "username": "admin",
        "first_name": "",
        "last_name": ""
      },
      "modified_by": {
        "id": 1,
        "username": "admin",
        "first_name": "",
        "last_name": ""
      },
      "object_roles": {
        "admin_role": {
          "description": "Can manage all aspects of the project",
          "name": "Admin",
          "id": 57
        },
        "use_role": {
          "description": "Can use the project in a job template",
          "name": "Use",
          "id": 58
        },
        "update_role": {
          "description": "May update the project",
          "name": "Update",
          "id": 59
        },
        "read_role": {
          "description": "May view settings for the project",
          "name": "Read",
          "id": 60
        }
      },
      "user_capabilities": {
        "edit": true,
        "delete": true,
        "start": true,
        "schedule": true,
        "copy": true
      }
    },
    "created": "2025-11-17T06:03:28.735832Z",
    "modified": "2025-11-17T06:03:28.735856Z",
    "name": "NGINX",
    "description": "",
    "local_path": "_7__nginx",
    "scm_type": "git",
    "scm_url": "https://gitflic.ru/project/dgaripov/ac-project.git",
    "scm_branch": "",
    "scm_refspec": "",
    "scm_clean": false,
    "scm_track_submodules": false,
    "scm_delete_on_update": false,
    "credential": 4,
    "timeout": 0,
    "scm_revision": "88c68e7f0a4c21e1d559d697ae85c9e7a6b31d3a",
    "last_job_run": "2025-11-17T06:03:37.360186Z",
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
      "last_job_failed": false,
      "next_job_run": null,
      "status": "successful",
      "organization": 1,
      "scm_update_on_launch": false,
      "scm_update_cache_timeout": 0,
      "allow_override": false,
      "custom_virtualenv": null,
      "default_environment": 2,
      "signature_validation_credential": null,
      "last_update_failed": false,
      "last_updated": "2025-11-17T06:03:37.360186Z"
    },
    {
      "id": 11,
      "type": "project",
      "url": "/api/controller/v2/projects/11/",
      "related": {
        "created_by": "/api/controller/v2/users/1/",
        "modified_by": "/api/controller/v2/users/1/",
        "last_job": "/api/controller/v2/project_updates/16/",
        "teams": "/api/controller/v2/projects/11/teams/",
        "playbooks": "/api/controller/v2/projects/11/playbooks/",
        "inventory_files": "/api/controller/v2/projects/11/inventories/",
        "update": "/api/controller/v2/projects/11/update/",
        "project_updates": "/api/controller/v2/projects/11/project_updates/",
        "scm_inventory_sources": "/api/controller/v2/projects/11/scm_inventory_sources/",
        "schedules": "/api/controller/v2/projects/11/schedules/",
        "activity_stream": "/api/controller/v2/projects/11/activity_stream/",
        "notification_templates_started": "/api/controller/v2/projects/11/notification_
↪templates_started/",
        "notification_templates_success": "/api/controller/v2/projects/11/notification_
↪templates_success/",
        "notification_templates_error": "/api/controller/v2/projects/11/notification_
↪templates_error/",
        "access_list": "/api/controller/v2/projects/11/access_list/",
        "object_roles": "/api/controller/v2/projects/11/object_roles/",
        "copy": "/api/controller/v2/projects/11/copy/",
        "organization": "/api/controller/v2/organizations/1/",
        "last_update": "/api/controller/v2/project_updates/16/"
      },
      "summary_fields": {
        "organization": {
          "id": 1,
          "name": "Default",
          "description": "The default organization for Astra Automation"
        },
        "last_job": {
          "id": 16,
          "name": "New Project",
          "description": "",
          "finished": "2025-11-25T08:21:59.748433Z",
          "status": "successful",
          "failed": false
        },
        "last_update": {
          "id": 16,
          "name": "New Project",
          "description": "",
          "status": "successful",
          "failed": false
        },
        "created_by": {
          "id": 1,
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
        "username": "admin",
        "first_name": "",
        "last_name": ""
      },
      "modified_by": {
        "id": 1,
        "username": "admin",
        "first_name": "",
        "last_name": ""
      },
      "object_roles": {
        "admin_role": {
          "description": "Can manage all aspects of the project",
          "name": "Admin",
          "id": 106
        },
        "use_role": {
          "description": "Can use the project in a job template",
          "name": "Use",
          "id": 107
        },
        "update_role": {
          "description": "May update the project",
          "name": "Update",
          "id": 108
        },
        "read_role": {
          "description": "May view settings for the project",
          "name": "Read",
          "id": 109
        }
      },
      "user_capabilities": {
        "edit": true,
        "delete": true,
        "start": true,
        "schedule": true,
        "copy": true
      }
    },
    "created": "2025-11-25T08:21:51.600418Z",
    "modified": "2025-11-25T08:21:51.600456Z",
    "name": "New Project",
    "description": "",
    "local_path": "_11__new_project",
    "scm_type": "git",
    "scm_url": "https://github.com/ansible/ansible-tower-samples",
    "scm_branch": "",
    "scm_refspec": "",
    "scm_clean": false,
    "scm_track_submodules": false,
    "scm_delete_on_update": false,
    "credential": null,
    "timeout": 0,
    "scm_revision": "347e44fea036c94d5f60e544de006453ee5c71ad",
    "last_job_run": "2025-11-25T08:21:59.748433Z",
    "last_job_failed": false,
    "next_job_run": null,
    "status": "successful",
    "organization": 1,
    "scm_update_on_launch": false,
    "scm_update_cache_timeout": 0,
    "allow_override": false,
    "custom_virtualenv": null,
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
      "default_environment": null,
      "signature_validation_credential": null,
      "last_update_failed": false,
      "last_updated": "2025-11-25T08:21:59.748433Z"
    }
  ]
}
```


Получение деталей проекта:


```
{
  "id": 11,
  "type": "project",
  "url": "/api/controller/v2/projects/11/",
  "related": {
    "named_url": "/api/controller/v2/projects/New Project++Default/",
    "created_by": "/api/controller/v2/users/1/",
    "modified_by": "/api/controller/v2/users/1/",
    "last_job": "/api/controller/v2/project_updates/16/",
    "teams": "/api/controller/v2/projects/11/teams/",
    "playbooks": "/api/controller/v2/projects/11/playbooks/",
    "inventory_files": "/api/controller/v2/projects/11/inventories/",
    "update": "/api/controller/v2/projects/11/update/",
    "project_updates": "/api/controller/v2/projects/11/project_updates/",
    "scm_inventory_sources": "/api/controller/v2/projects/11/scm_inventory_sources/",
    "schedules": "/api/controller/v2/projects/11/schedules/",
    "activity_stream": "/api/controller/v2/projects/11/activity_stream/",
    "notification_templates_started": "/api/controller/v2/projects/11/notification_
↪templates_started/",
    "notification_templates_success": "/api/controller/v2/projects/11/notification_
↪templates_success/",
    "notification_templates_error": "/api/controller/v2/projects/11/notification_
↪templates_error/",
    "access_list": "/api/controller/v2/projects/11/access_list/",
    "object_roles": "/api/controller/v2/projects/11/object_roles/",
    "copy": "/api/controller/v2/projects/11/copy/",
    "organization": "/api/controller/v2/organizations/1/",
    "last_update": "/api/controller/v2/project_updates/16/"
  },
  "summary_fields": {
    "organization": {
      "id": 1,
      "name": "Default",
      "description": "The default organization for Astra Automation"
    },
    "last_job": {
      "id": 16,
      "name": "New Project",
      "description": "",
      "finished": "2025-11-25T08:21:59.748433Z",
      "status": "successful",
      "failed": false
    },
    "last_update": {
      "id": 16,
      "name": "New Project",
      "description": "",
      "status": "successful",
      "failed": false
    },
    "created_by": {
      "id": 1,
      "username": "admin",
      "first_name": "",
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
      "last_name": ""
    },
    "modified_by": {
      "id": 1,
      "username": "admin",
      "first_name": "",
      "last_name": ""
    },
    "object_roles": {
      "admin_role": {
        "description": "Can manage all aspects of the project",
        "name": "Admin",
        "id": 106
      },
      "use_role": {
        "description": "Can use the project in a job template",
        "name": "Use",
        "id": 107
      },
      "update_role": {
        "description": "May update the project",
        "name": "Update",
        "id": 108
      },
      "read_role": {
        "description": "May view settings for the project",
        "name": "Read",
        "id": 109
      }
    },
    "user_capabilities": {
      "edit": true,
      "delete": true,
      "start": true,
      "schedule": true,
      "copy": true
    },
    "resolved_environment": {
      "id": 2,
      "name": "Default execution environment",
      "description": "",
      "image": "hub.astra-automation.ru/aa-1.2/aa-full-ee:latest"
    }
  },
  "created": "2025-11-25T08:21:51.600418Z",
  "modified": "2025-11-25T08:21:51.600456Z",
  "name": "New Project",
  "description": "",
  "local_path": "_11__new_project",
  "scm_type": "git",
  "scm_url": "https://github.com/ansible/ansible-tower-samples",
  "scm_branch": "",
  "scm_refspec": "",
  "scm_clean": false,
  "scm_track_submodules": false,
  "scm_delete_on_update": false,
  "credential": null,
  "timeout": 0,
  "scm_revision": "347e44fea036c94d5f60e544de006453ee5c71ad",
  "last_job_run": "2025-11-25T08:21:59.748433Z",
  "last_job_failed": false,
  "next_job_run": null,
  "status": "successful",
  "organization": 1,
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
  "scm_update_on_launch": false,
  "scm_update_cache_timeout": 0,
  "allow_override": false,
  "custom_virtualenv": null,
  "default_environment": null,
  "signature_validation_credential": null,
  "last_update_failed": false,
  "last_updated": "2025-11-25T08:21:59.748433Z"
}
```


Обновление проекта из источника контроля версий:


```
{
  "project_update": 17,
  "id": 17,
  "type": "project_update",
  "url": "/api/controller/v2/project_updates/17/",
  "related": {
    "created_by": "/api/controller/v2/users/1/",
    "modified_by": "/api/controller/v2/users/1/",
    "unified_job_template": "/api/controller/v2/projects/11/",
    "stdout": "/api/controller/v2/project_updates/17/stdout/",
    "project": "/api/controller/v2/projects/11/",
    "cancel": "/api/controller/v2/project_updates/17/cancel/",
    "scm_inventory_updates": "/api/controller/v2/project_updates/17/scm_inventory_
↪updates/",
    "notifications": "/api/controller/v2/project_updates/17/notifications/",
    "events": "/api/controller/v2/project_updates/17/events/"
  },
  "summary_fields": {
    "organization": {
      "id": 1,
      "name": "Default",
      "description": "The default organization for Astra Automation"
    },
    "project": {
      "id": 11,
      "name": "New Project",
      "description": "",
      "status": "pending",
      "scm_type": "git",
      "allow_override": false
    },
    "unified_job_template": {
      "id": 11,
      "name": "New Project",
      "description": "",
      "unified_job_type": "project_update"
    },
    "created_by": {
      "id": 1,
      "username": "admin",
      "first_name": "",
      "last_name": ""
    },
    "modified_by": {
      "id": 1,
      "username": "admin",
      "first_name": "",
      "last_name": ""
    },
    "user_capabilities": {
      "delete": true,
      "start": true
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
    }
  },
  "created": "2025-11-25T08:23:19.689476Z",
  "modified": "2025-11-25T08:23:19.714576Z",
  "name": "New Project",
  "description": "",
  "local_path": "_11__new_project",
  "scm_type": "git",
  "scm_url": "https://github.com/ansible/ansible-tower-samples",
  "scm_branch": "",
  "scm_refspec": "",
  "scm_clean": false,
  "scm_track_submodules": false,
  "scm_delete_on_update": false,
  "credential": null,
  "timeout": 0,
  "scm_revision": "",
  "unified_job_template": 11,
  "launch_type": "manual",
  "status": "pending",
  "execution_environment": null,
  "failed": false,
  "started": null,
  "finished": null,
  "canceled_on": null,
  "elapsed": 0.0,
  "job_args": "",
  "job_cwd": "",
  "job_env": {},
  "job_explanation": "",
  "execution_node": "",
  "result_traceback": "",
  "event_processing_finished": false,
  "launched_by": {
    "id": 1,
    "name": "admin",
    "type": "user",
    "url": "/api/controller/v2/users/1/"
  },
  "work_unit_id": null,
  "project": 11,
  "job_type": "check",
  "job_tags": "update_git,install_roles,install_collections"
}
```


Проверка информации о статусе обновления:


```
{
  "count": 2,
  "next": null,
  "previous": null,
  "results": [
    {
      "id": 16,
      "type": "project_update",
      "url": "/api/controller/v2/project_updates/16/",
      "related": {
        "created_by": "/api/controller/v2/users/1/",
        "unified_job_template": "/api/controller/v2/projects/11/",
        "stdout": "/api/controller/v2/project_updates/16/stdout/",
        "execution_environment": "/api/controller/v2/execution_environments/3/",
        "project": "/api/controller/v2/projects/11/",
        "cancel": "/api/controller/v2/project_updates/16/cancel/",
        "scm_inventory_updates": "/api/controller/v2/project_updates/16/scm_inventory_
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
↪updates/",
        "notifications": "/api/controller/v2/project_updates/16/notifications/",
        "events": "/api/controller/v2/project_updates/16/events/"
      },
      "summary_fields": {
        "organization": {
          "id": 1,
          "name": "Default",
          "description": "The default organization for Astra Automation"
        },
        "execution_environment": {
          "id": 3,
          "name": "Control Plane Execution Environment",
          "description": "",
          "image": "hub.astra-automation.ru/aa-1.2/aa-control-ee:latest"
        },
        "project": {
          "id": 11,
          "name": "New Project",
          "description": "",
          "status": "successful",
          "scm_type": "git",
          "allow_override": false
        },
        "unified_job_template": {
          "id": 11,
          "name": "New Project",
          "description": "",
          "unified_job_type": "project_update"
        },
        "instance_group": {
          "id": 1,
          "name": "controlplane",
          "is_container_group": false
        },
        "created_by": {
          "id": 1,
          "username": "admin",
          "first_name": "",
          "last_name": ""
        },
        "user_capabilities": {
          "delete": true,
          "start": true
        }
      },
      "created": "2025-11-25T08:21:52.080566Z",
      "modified": "2025-11-25T08:21:52.414131Z",
      "name": "New Project",
      "description": "",
      "unified_job_template": 11,
      "launch_type": "manual",
      "status": "successful",
      "execution_environment": 3,
      "failed": false,
      "started": "2025-11-25T08:21:52.555787Z",
      "finished": "2025-11-25T08:21:59.748433Z",
      "canceled_on": null,
      "elapsed": 7.193,
      "job_explanation": "",
      "execution_node": "10.205.212.90",
      "launched_by": {
        "id": 1,
        "name": "admin",
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
        "type": "user",
        "url": "/api/controller/v2/users/1/"
      },
      "work_unit_id": "1020521290P82YebHM",
      "local_path": "_11__new_project",
      "scm_type": "git",
      "scm_url": "https://github.com/ansible/ansible-tower-samples",
      "scm_branch": "",
      "scm_refspec": "",
      "scm_clean": false,
      "scm_track_submodules": false,
      "scm_delete_on_update": false,
      "credential": null,
      "timeout": 0,
      "scm_revision": "347e44fea036c94d5f60e544de006453ee5c71ad",
      "project": 11,
      "job_type": "check",
      "job_tags": "update_git,install_roles,install_collections"
    },
    {
      "id": 17,
      "type": "project_update",
      "url": "/api/controller/v2/project_updates/17/",
      "related": {
        "created_by": "/api/controller/v2/users/1/",
        "unified_job_template": "/api/controller/v2/projects/11/",
        "stdout": "/api/controller/v2/project_updates/17/stdout/",
        "execution_environment": "/api/controller/v2/execution_environments/3/",
        "project": "/api/controller/v2/projects/11/",
        "cancel": "/api/controller/v2/project_updates/17/cancel/",
        "scm_inventory_updates": "/api/controller/v2/project_updates/17/scm_inventory_
↪updates/",
        "notifications": "/api/controller/v2/project_updates/17/notifications/",
        "events": "/api/controller/v2/project_updates/17/events/"
      },
      "summary_fields": {
        "organization": {
          "id": 1,
          "name": "Default",
          "description": "The default organization for Astra Automation"
        },
        "execution_environment": {
          "id": 3,
          "name": "Control Plane Execution Environment",
          "description": "",
          "image": "hub.astra-automation.ru/aa-1.2/aa-control-ee:latest"
        },
        "project": {
          "id": 11,
          "name": "New Project",
          "description": "",
          "status": "successful",
          "scm_type": "git",
          "allow_override": false
        },
        "unified_job_template": {
          "id": 11,
          "name": "New Project",
          "description": "",
          "unified_job_type": "project_update"
        },
        "instance_group": {
          "id": 1,
          "name": "controlplane",
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
          "is_container_group": false
        },
        "created_by": {
          "id": 1,
          "username": "admin",
          "first_name": "",
          "last_name": ""
        },
        "user_capabilities": {
          "delete": true,
          "start": true
        }
      },
      "created": "2025-11-25T08:23:19.689476Z",
      "modified": "2025-11-25T08:23:20.090734Z",
      "name": "New Project",
      "description": "",
      "unified_job_template": 11,
      "launch_type": "manual",
      "status": "successful",
      "execution_environment": 3,
      "failed": false,
      "started": "2025-11-25T08:23:20.198178Z",
      "finished": "2025-11-25T08:23:26.286009Z",
      "canceled_on": null,
      "elapsed": 6.088,
      "job_explanation": "",
      "execution_node": "10.205.212.90",
      "launched_by": {
        "id": 1,
        "name": "admin",
        "type": "user",
        "url": "/api/controller/v2/users/1/"
      },
      "work_unit_id": "1020521290u1HqqqwP",
      "local_path": "_11__new_project",
      "scm_type": "git",
      "scm_url": "https://github.com/ansible/ansible-tower-samples",
      "scm_branch": "",
      "scm_refspec": "",
      "scm_clean": false,
      "scm_track_submodules": false,
      "scm_delete_on_update": false,
      "credential": null,
      "timeout": 0,
      "scm_revision": "347e44fea036c94d5f60e544de006453ee5c71ad",
      "project": 11,
      "job_type": "check",
      "job_tags": "update_git,install_roles,install_collections"
    }
  ]
}
```


Изменение проекта:


```
{
  "id": 11,
  "type": "project",
  "url": "/api/controller/v2/projects/11/",
  "related": {
    "named_url": "/api/controller/v2/projects/Updated Project++Default/",
    "created_by": "/api/controller/v2/users/1/",
    "modified_by": "/api/controller/v2/users/1/",
    "current_job": "/api/controller/v2/project_updates/18/",
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
    "last_job": "/api/controller/v2/project_updates/17/",
    "teams": "/api/controller/v2/projects/11/teams/",
    "playbooks": "/api/controller/v2/projects/11/playbooks/",
    "inventory_files": "/api/controller/v2/projects/11/inventories/",
    "update": "/api/controller/v2/projects/11/update/",
    "project_updates": "/api/controller/v2/projects/11/project_updates/",
    "scm_inventory_sources": "/api/controller/v2/projects/11/scm_inventory_sources/",
    "schedules": "/api/controller/v2/projects/11/schedules/",
    "activity_stream": "/api/controller/v2/projects/11/activity_stream/",
    "notification_templates_started": "/api/controller/v2/projects/11/notification_
↪templates_started/",
    "notification_templates_success": "/api/controller/v2/projects/11/notification_
↪templates_success/",
    "notification_templates_error": "/api/controller/v2/projects/11/notification_
↪templates_error/",
    "access_list": "/api/controller/v2/projects/11/access_list/",
    "object_roles": "/api/controller/v2/projects/11/object_roles/",
    "copy": "/api/controller/v2/projects/11/copy/",
    "organization": "/api/controller/v2/organizations/1/",
    "current_update": "/api/controller/v2/project_updates/18/",
    "last_update": "/api/controller/v2/project_updates/17/"
  },
  "summary_fields": {
    "organization": {
      "id": 1,
      "name": "Default",
      "description": "The default organization for Astra Automation"
    },
    "last_job": {
      "id": 17,
      "name": "New Project",
      "description": "",
      "finished": "2025-11-25T08:23:26.286009Z",
      "status": "successful",
      "failed": false
    },
    "last_update": {
      "id": 17,
      "name": "New Project",
      "description": "",
      "status": "successful",
      "failed": false
    },
    "current_update": {
      "id": 18,
      "name": "Updated Project",
      "description": "",
      "status": "pending",
      "failed": false
    },
    "current_job": {
      "id": 18,
      "name": "Updated Project",
      "description": "",
      "status": "pending",
      "failed": false
    },
    "created_by": {
      "id": 1,
      "username": "admin",
      "first_name": "",
      "last_name": ""
    },
    "modified_by": {
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
      "id": 1,
      "username": "admin",
      "first_name": "",
      "last_name": ""
    },
    "object_roles": {
      "admin_role": {
        "description": "Can manage all aspects of the project",
        "name": "Admin",
        "id": 106
      },
      "use_role": {
        "description": "Can use the project in a job template",
        "name": "Use",
        "id": 107
      },
      "update_role": {
        "description": "May update the project",
        "name": "Update",
        "id": 108
      },
      "read_role": {
        "description": "May view settings for the project",
        "name": "Read",
        "id": 109
      }
    },
    "user_capabilities": {
      "edit": true,
      "delete": true,
      "start": true,
      "schedule": true,
      "copy": true
    },
    "resolved_environment": {
      "id": 2,
      "name": "Default execution environment",
      "description": "",
      "image": "hub.astra-automation.ru/aa-1.2/aa-full-ee:latest"
    }
  },
  "created": "2025-11-25T08:21:51.600418Z",
  "modified": "2025-11-25T08:23:58.333025Z",
  "name": "Updated Project",
  "description": "",
  "local_path": "_11__new_project",
  "scm_type": "git",
  "scm_url": "https://github.com/ansible/ansible-examples",
  "scm_branch": "",
  "scm_refspec": "",
  "scm_clean": false,
  "scm_track_submodules": false,
  "scm_delete_on_update": false,
  "credential": null,
  "timeout": 0,
  "scm_revision": "347e44fea036c94d5f60e544de006453ee5c71ad",
  "last_job_run": "2025-11-25T08:23:26.286009Z",
  "last_job_failed": false,
  "next_job_run": null,
  "status": "pending",
  "organization": 1,
  "scm_update_on_launch": false,
  "scm_update_cache_timeout": 0,
  "allow_override": false,
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
  "custom_virtualenv": null,
  "default_environment": null,
  "signature_validation_credential": null,
  "last_update_failed": false,
  "last_updated": "2025-11-25T08:23:26.286009Z"
}
```


Получение деталей проекта (после обновления):


```
{
  "id": 11,
  "type": "project",
  "url": "/api/controller/v2/projects/11/",
  "related": {
    "named_url": "/api/controller/v2/projects/Updated Project++Default/",
    "created_by": "/api/controller/v2/users/1/",
    "modified_by": "/api/controller/v2/users/1/",
    "last_job": "/api/controller/v2/project_updates/18/",
    "teams": "/api/controller/v2/projects/11/teams/",
    "playbooks": "/api/controller/v2/projects/11/playbooks/",
    "inventory_files": "/api/controller/v2/projects/11/inventories/",
    "update": "/api/controller/v2/projects/11/update/",
    "project_updates": "/api/controller/v2/projects/11/project_updates/",
    "scm_inventory_sources": "/api/controller/v2/projects/11/scm_inventory_sources/",
    "schedules": "/api/controller/v2/projects/11/schedules/",
    "activity_stream": "/api/controller/v2/projects/11/activity_stream/",
    "notification_templates_started": "/api/controller/v2/projects/11/notification_
↪templates_started/",
    "notification_templates_success": "/api/controller/v2/projects/11/notification_
↪templates_success/",
    "notification_templates_error": "/api/controller/v2/projects/11/notification_
↪templates_error/",
    "access_list": "/api/controller/v2/projects/11/access_list/",
    "object_roles": "/api/controller/v2/projects/11/object_roles/",
    "copy": "/api/controller/v2/projects/11/copy/",
    "organization": "/api/controller/v2/organizations/1/",
    "last_update": "/api/controller/v2/project_updates/18/"
  },
  "summary_fields": {
    "organization": {
      "id": 1,
      "name": "Default",
      "description": "The default organization for Astra Automation"
    },
    "last_job": {
      "id": 18,
      "name": "Updated Project",
      "description": "",
      "finished": "2025-11-25T08:24:07.074135Z",
      "status": "successful",
      "failed": false
    },
    "last_update": {
      "id": 18,
      "name": "Updated Project",
      "description": "",
      "status": "successful",
      "failed": false
    },
    "created_by": {
      "id": 1,
      "username": "admin",
      "first_name": "",
      "last_name": ""
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
    },
    "modified_by": {
      "id": 1,
      "username": "admin",
      "first_name": "",
      "last_name": ""
    },
    "object_roles": {
      "admin_role": {
        "description": "Can manage all aspects of the project",
        "name": "Admin",
        "id": 106
      },
      "use_role": {
        "description": "Can use the project in a job template",
        "name": "Use",
        "id": 107
      },
      "update_role": {
        "description": "May update the project",
        "name": "Update",
        "id": 108
      },
      "read_role": {
        "description": "May view settings for the project",
        "name": "Read",
        "id": 109
      }
    },
    "user_capabilities": {
      "edit": true,
      "delete": true,
      "start": true,
      "schedule": true,
      "copy": true
    },
    "resolved_environment": {
      "id": 2,
      "name": "Default execution environment",
      "description": "",
      "image": "hub.astra-automation.ru/aa-1.2/aa-full-ee:latest"
    }
  },
  "created": "2025-11-25T08:21:51.600418Z",
  "modified": "2025-11-25T08:23:58.333025Z",
  "name": "Updated Project",
  "description": "",
  "local_path": "_11__new_project",
  "scm_type": "git",
  "scm_url": "https://github.com/ansible/ansible-examples",
  "scm_branch": "",
  "scm_refspec": "",
  "scm_clean": false,
  "scm_track_submodules": false,
  "scm_delete_on_update": false,
  "credential": null,
  "timeout": 0,
  "scm_revision": "b50586543c6c4be907fdc88f9f78a2b35d2a895f",
  "last_job_run": "2025-11-25T08:24:07.074135Z",
  "last_job_failed": false,
  "next_job_run": null,
  "status": "successful",
  "organization": 1,
  "scm_update_on_launch": false,
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
  "scm_update_cache_timeout": 0,
  "allow_override": false,
  "custom_virtualenv": null,
  "default_environment": null,
  "signature_validation_credential": null,
  "last_update_failed": false,
  "last_updated": "2025-11-25T08:24:07.074135Z"
}
```


Удаление проекта:

```
Status code: 204
```

Получение списка проектов (после удаления):


```
{
  "count": 2,
  "next": null,
  "previous": null,
  "results": [
    {
      "id": 5,
      "type": "project",
      "url": "/api/controller/v2/projects/5/",
      "related": {
        "created_by": "/api/controller/v2/users/1/",
        "modified_by": "/api/controller/v2/users/1/",
        "teams": "/api/controller/v2/projects/5/teams/",
        "playbooks": "/api/controller/v2/projects/5/playbooks/",
        "inventory_files": "/api/controller/v2/projects/5/inventories/",
        "update": "/api/controller/v2/projects/5/update/",
        "project_updates": "/api/controller/v2/projects/5/project_updates/",
        "scm_inventory_sources": "/api/controller/v2/projects/5/scm_inventory_sources/",
        "schedules": "/api/controller/v2/projects/5/schedules/",
        "activity_stream": "/api/controller/v2/projects/5/activity_stream/",
        "notification_templates_started": "/api/controller/v2/projects/5/notification_
↪templates_started/",
        "notification_templates_success": "/api/controller/v2/projects/5/notification_
↪templates_success/",
        "notification_templates_error": "/api/controller/v2/projects/5/notification_
↪templates_error/",
        "access_list": "/api/controller/v2/projects/5/access_list/",
        "object_roles": "/api/controller/v2/projects/5/object_roles/",
        "copy": "/api/controller/v2/projects/5/copy/",
        "organization": "/api/controller/v2/organizations/1/"
      },
      "summary_fields": {
        "organization": {
          "id": 1,
          "name": "Default",
          "description": "The default organization for Astra Automation"
        },
        "created_by": {
          "id": 1,
          "username": "admin",
          "first_name": "",
          "last_name": ""
        },
        "modified_by": {
          "id": 1,
          "username": "admin",
          "first_name": "",
          "last_name": ""
        },
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
      "object_roles": {
        "admin_role": {
          "description": "Can manage all aspects of the project",
          "name": "Admin",
          "id": 22
        },
        "use_role": {
          "description": "Can use the project in a job template",
          "name": "Use",
          "id": 23
        },
        "update_role": {
          "description": "May update the project",
          "name": "Update",
          "id": 24
        },
        "read_role": {
          "description": "May view settings for the project",
          "name": "Read",
          "id": 25
        }
      },
      "user_capabilities": {
        "edit": true,
        "delete": true,
        "start": true,
        "schedule": true,
        "copy": true
      }
    },
    "created": "2025-11-12T10:21:56.220632Z",
    "modified": "2025-11-12T10:21:56.220653Z",
    "name": "Demo Project",
    "description": "",
    "local_path": "_5__demo_project",
    "scm_type": "git",
    "scm_url": "https://github.com/ansible/ansible-tower-samples",
    "scm_branch": "",
    "scm_refspec": "",
    "scm_clean": false,
    "scm_track_submodules": false,
    "scm_delete_on_update": false,
    "credential": null,
    "timeout": 0,
    "scm_revision": "347e44fea036c94d5f60e544de006453ee5c71ad",
    "last_job_run": null,
    "last_job_failed": false,
    "next_job_run": null,
    "status": "successful",
    "organization": 1,
    "scm_update_on_launch": false,
    "scm_update_cache_timeout": 0,
    "allow_override": false,
    "custom_virtualenv": null,
    "default_environment": null,
    "signature_validation_credential": null,
    "last_update_failed": false,
    "last_updated": null
  },
  {
    "id": 7,
    "type": "project",
    "url": "/api/controller/v2/projects/7/",
    "related": {
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
        "created_by": "/api/controller/v2/users/1/",
        "modified_by": "/api/controller/v2/users/1/",
        "credential": "/api/controller/v2/credentials/4/",
        "last_job": "/api/controller/v2/project_updates/6/",
        "teams": "/api/controller/v2/projects/7/teams/",
        "playbooks": "/api/controller/v2/projects/7/playbooks/",
        "inventory_files": "/api/controller/v2/projects/7/inventories/",
        "update": "/api/controller/v2/projects/7/update/",
        "project_updates": "/api/controller/v2/projects/7/project_updates/",
        "scm_inventory_sources": "/api/controller/v2/projects/7/scm_inventory_sources/",
        "schedules": "/api/controller/v2/projects/7/schedules/",
        "activity_stream": "/api/controller/v2/projects/7/activity_stream/",
        "notification_templates_started": "/api/controller/v2/projects/7/notification_
↪templates_started/",
        "notification_templates_success": "/api/controller/v2/projects/7/notification_
↪templates_success/",
        "notification_templates_error": "/api/controller/v2/projects/7/notification_
↪templates_error/",
        "access_list": "/api/controller/v2/projects/7/access_list/",
        "object_roles": "/api/controller/v2/projects/7/object_roles/",
        "copy": "/api/controller/v2/projects/7/copy/",
        "organization": "/api/controller/v2/organizations/1/",
        "default_environment": "/api/controller/v2/execution_environments/2/",
        "last_update": "/api/controller/v2/project_updates/6/"
      },
      "summary_fields": {
        "organization": {
          "id": 1,
          "name": "Default",
          "description": "The default organization for Astra Automation"
        },
        "default_environment": {
          "id": 2,
          "name": "Default execution environment",
          "description": "",
          "image": "hub.astra-automation.ru/aa-1.2/aa-full-ee:latest"
        },
        "credential": {
          "id": 4,
          "name": "SCM access",
          "description": "",
          "kind": "scm",
          "cloud": false,
          "kubernetes": false,
          "credential_type_id": 8
        },
        "last_job": {
          "id": 6,
          "name": "NGINX",
          "description": "",
          "finished": "2025-11-17T06:03:37.360186Z",
          "status": "successful",
          "failed": false
        },
        "last_update": {
          "id": 6,
          "name": "NGINX",
          "description": "",
          "status": "successful",
          "failed": false
        },
        "created_by": {
          "id": 1,
          "username": "admin",
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
        "first_name": "",
        "last_name": ""
      },
      "modified_by": {
        "id": 1,
        "username": "admin",
        "first_name": "",
        "last_name": ""
      },
      "object_roles": {
        "admin_role": {
          "description": "Can manage all aspects of the project",
          "name": "Admin",
          "id": 57
        },
        "use_role": {
          "description": "Can use the project in a job template",
          "name": "Use",
          "id": 58
        },
        "update_role": {
          "description": "May update the project",
          "name": "Update",
          "id": 59
        },
        "read_role": {
          "description": "May view settings for the project",
          "name": "Read",
          "id": 60
        }
      },
      "user_capabilities": {
        "edit": true,
        "delete": true,
        "start": true,
        "schedule": true,
        "copy": true
      }
    },
    "created": "2025-11-17T06:03:28.735832Z",
    "modified": "2025-11-17T06:03:28.735856Z",
    "name": "NGINX",
    "description": "",
    "local_path": "_7__nginx",
    "scm_type": "git",
    "scm_url": "https://gitflic.ru/project/dgaripov/ac-project.git",
    "scm_branch": "",
    "scm_refspec": "",
    "scm_clean": false,
    "scm_track_submodules": false,
    "scm_delete_on_update": false,
    "credential": 4,
    "timeout": 0,
    "scm_revision": "88c68e7f0a4c21e1d559d697ae85c9e7a6b31d3a",
    "last_job_run": "2025-11-17T06:03:37.360186Z",
    "last_job_failed": false,
    "next_job_run": null,
    "status": "successful",
    "organization": 1,
    "scm_update_on_launch": false,
    "scm_update_cache_timeout": 0,
    "allow_override": false,
    "custom_virtualenv": null,
    "default_environment": 2,
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
      "signature_validation_credential": null,
      "last_update_failed": false,
      "last_updated": "2025-11-17T06:03:37.360186Z"
    }
  ]
}
```


Получение конфигурации Automation Controller:


```
{
  "time_zone": "UTC",
  "license_info": {
    "instance_count": 10,
    "license_date": "1767225599",
    "license_type": "enterprise",
    "sku": "24929",
    "usage": "",
    "pool_id": "8a8883dc94ae933601952859afc905c2",
    "satellite": false,
    "valid_key": true,
    "product_name": "Astra Automation",
    "support_level": "",
    "account_number": null,
    "subscription_id": null,
    "subscription_name": "Astra Automation",
    "deleted_instances": 0,
    "reactivated_instances": 0,
    "current_instances": 2,
    "automated_instances": 2,
    "automated_since": 1763030365,
    "free_instances": 8,
    "time_remaining": 3166479,
    "trial": false,
    "grace_period_remaining": 5758479,
    "compliant": true,
    "date_warning": false,
    "date_expired": false
  },
  "version": "2.1.0+51f9160",
  "eula": "\u041b...",
  "analytics_status": "off",
  "analytics_collectors": {
    "config": {
      "name": "config",
      "version": "1.6",
      "description": "General platform configuration."
    },
    "counts": {
      "name": "counts",
      "version": "1.2",
      "description": "Counts of objects such as organizations, inventories, and projects"
    },
    "cred_type_counts": {
      "name": "cred_type_counts",
      "version": "1.0",
      "description": "Counts of credentials by credential type"
    },
    "events_table": {
      "name": "events_table",
      "version": "1.5",
      "description": "Automation task records"
    },
    "host_metric_summary_monthly_table": {
      "name": "host_metric_summary_monthly_table",
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
      "version": "1.0",
      "description": "HostMetricSummaryMonthly export, full sync"
    },
    "host_metric_table": {
      "name": "host_metric_table",
      "version": "1.0",
      "description": "Host Metric data, incremental/full sync"
    },
    "instance_info": {
      "name": "instance_info",
      "version": "1.3",
      "description": "Cluster topology and capacity"
    },
    "inventory_counts": {
      "name": "inventory_counts",
      "version": "1.2",
      "description": "Inventories, their inventory sources, and host counts"
    },
    "org_counts": {
      "name": "org_counts",
      "version": "1.0",
      "description": "Counts of users and teams by organization"
    },
    "projects_by_scm_type": {
      "name": "projects_by_scm_type",
      "version": "1.0",
      "description": "Counts of projects by source control type"
    },
    "query_info": {
      "name": "query_info",
      "version": "1.0",
      "description": "Metadata about the analytics collected"
    },
    "switched_off_instance_info": {
      "name": "switched_off_instance_info",
      "version": "1.0",
      "description": "Switched off instances"
    },
    "unified_job_template_table": {
      "name": "unified_job_template_table",
      "version": "1.1",
      "description": "Data on job templates"
    },
    "unified_jobs_table": {
      "name": "unified_jobs_table",
      "version": "1.4",
      "description": "Data on jobs run"
    },
    "workflow_job_node_table": {
      "name": "workflow_job_node_table",
      "version": "1.0",
      "description": "Data on workflow runs"
    },
    "workflow_job_template_node_table": {
      "name": "workflow_job_template_node_table",
      "version": "1.0",
      "description": "Data on workflows"
    }
  },
  "become_methods": [
    ["sudo", "Sudo"],
    ["su", "Su"],
    ["pbrun", "Pbrun"],
    ["pfexec", "Pfexec"],
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
    ["dzdo", "DZDO"],
    ["pmrun", "Pmrun"],
    ["runas", "Runas"],
    ["enable", "Enable"],
    ["doas", "Doas"],
    ["ksu", "Ksu"],
    ["machinectl", "Machinectl"],
    ["sesu", "Sesu"]
  ],
  "project_base_dir": "/var/lib/awx/projects",
  "project_local_paths": [],
  "custom_virtualenvs": []
}
```


9.2.4 Спецификация API

ГЛАВА 10

Управление контентом

Внутренний реестр Astra Automation является средством распространения образов среды исполнения и коллекций Ansible. Он содержит контент, выбранный владельцем платформы:

- разработанный собственными силами;
- загруженный из Automation Hub[99], в том числе сертифицированные и проверенные коллекции Ansible и образы среды исполнения;
- загруженный из Ansible Galaxy[100] и других сторонних источников.

Примечание

В основе Private Automation Hub лежит Pulp Project[101] – программное обеспечение с открытым исходным кодом, предназначенное для получения, выгрузки и распространения пакетов ПО в собственной или облачной инфраструктуре.

10.1 Особенности архитектуры

Важно

Эта часть документации находится в стадии разработки.

10.1.1 Хранилище

По умолчанию Private Automation Hub использует для хранения содержимого локальную файловую систему. При развертывании на нескольких узлах следует использовать для хранения содержимого NFS или облачное хранилище S3.

При использовании NFS нужно будет смонтировать том вручную на каждом узле Private Automation Hub.

Чтобы использовать хранилище S3, необходимо выполнить следующие действия:

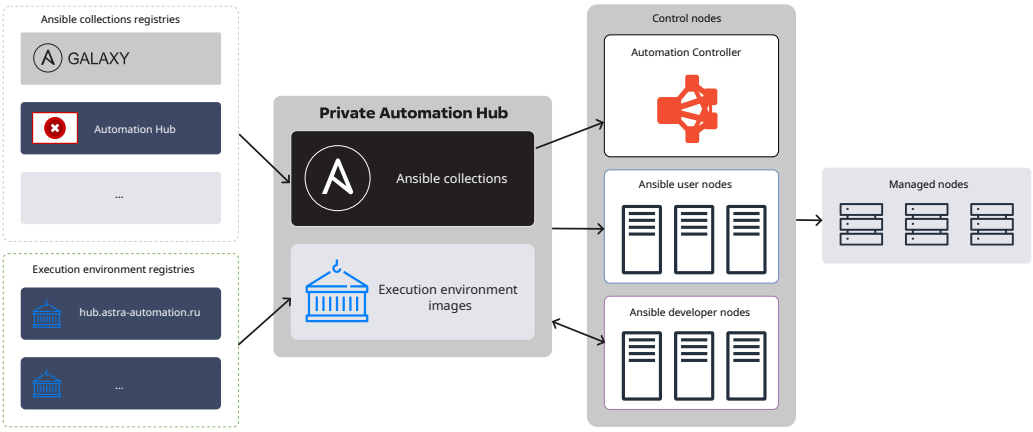
[99] https://hub.astra-automation.ru/
[100] https://galaxy.ansible.com/
[101] https://pulpproject.org/

1. Задать настройки доступа к хранилищу в *описании инвентаря установщика платформы*;
2. Запустить установщик платформы.

При изменении типа хранилища обновление ссылок в базе данных Private Automation Hub не производится. Это значит, что при условии сохранения путей к файлам смена локального хранилища на NFS или в обратную сторону не требует дополнительных действий. В то же время переход на использование хранилища S3 или в обратную сторону требует удаления и повторной загрузки всего содержимого.

10.2 Структура управления

Private Automation Hub используется для распространения коллекций Ansible и образов среды исполнения. Они необходимы для выполнения сценариев автоматизации с помощью различных средств Ansible, например, *Automation Controller* или *Ansible Navigator*.



Коллекции могут быть загружены из внешних источников (remotes) и с рабочих станций разработчиков. Образы среды исполнения могут быть загружены из внешних реестров (image registries) и с рабочих станций разработчиков.

10.2.1 Внешние источники коллекций

Внешний источник коллекций – это сторонний сервер, реализующий определенный API для работы с коллекциями, например:

- Automation Hub[102];
- Ansible Galaxy[103];
- другие Private Automation Hub.

10.2.2 Репозитории

Репозиторий ограничивает доступность хранящихся в нем версий коллекций для пользователей. В зависимости от настроек репозитория, размещенные в нем версии коллекций могут быть доступны следующим категориям пользователей:

- только администраторы;
- только авторизованные пользователи;
- все пользователи.

Репозиторий может быть защищенным. *Защищенные репозитории* нельзя удалить, однако, можно изменить некоторые из их настроек.

Список защищенных репозиториев (в алфавитном порядке):

[102] https://hub.astra-automation.ru/
[103] https://galaxy.ansible.com/

- `aa-certified` – *сертифицированные* коллекции, разработанные ПАО Группа Астра и ее партнерами;
- `community` – коллекции, разработанные сообществом пользователей Ansible;
- `published` – коллекции, размещенные владельцем реестра для определенных целей;
- `rejected` – коллекции, не прошедшие успешно процедуру проверки (validation);
- `staging` – коллекции, ожидающие согласования в процессе их проверки (validation);
- `validated` – *проверенные* коллекции.

Остальные репозитории создаются пользователями Private Automation Hub и могут быть удалены или изменены по их усмотрению.

Репозиторий может быть настроен на получение коллекций из внешнего источника. В этом случае для него становится доступна *синхронизация*.

Синхронизация

Синхронизация – это процесс, в результате которого содержимое репозитория Private Automation Hub приводится в соответствие с содержимым связанного внешнего источника коллекций.

Синхронизация может выполняться в режиме **Зеркало** (Mirror). В этом случае Private Automation Hub выполняет с версиями коллекций следующие действия:

- удаляет версии, отсутствующие в источнике;
- загружает из внешнего источника версии, которых нет в репозитории;
- отмечает как устаревшие версии, отмеченные как устаревшие в источнике;
- отмечает как актуальные версии, отмеченные как актуальные в источнике.

Приватность

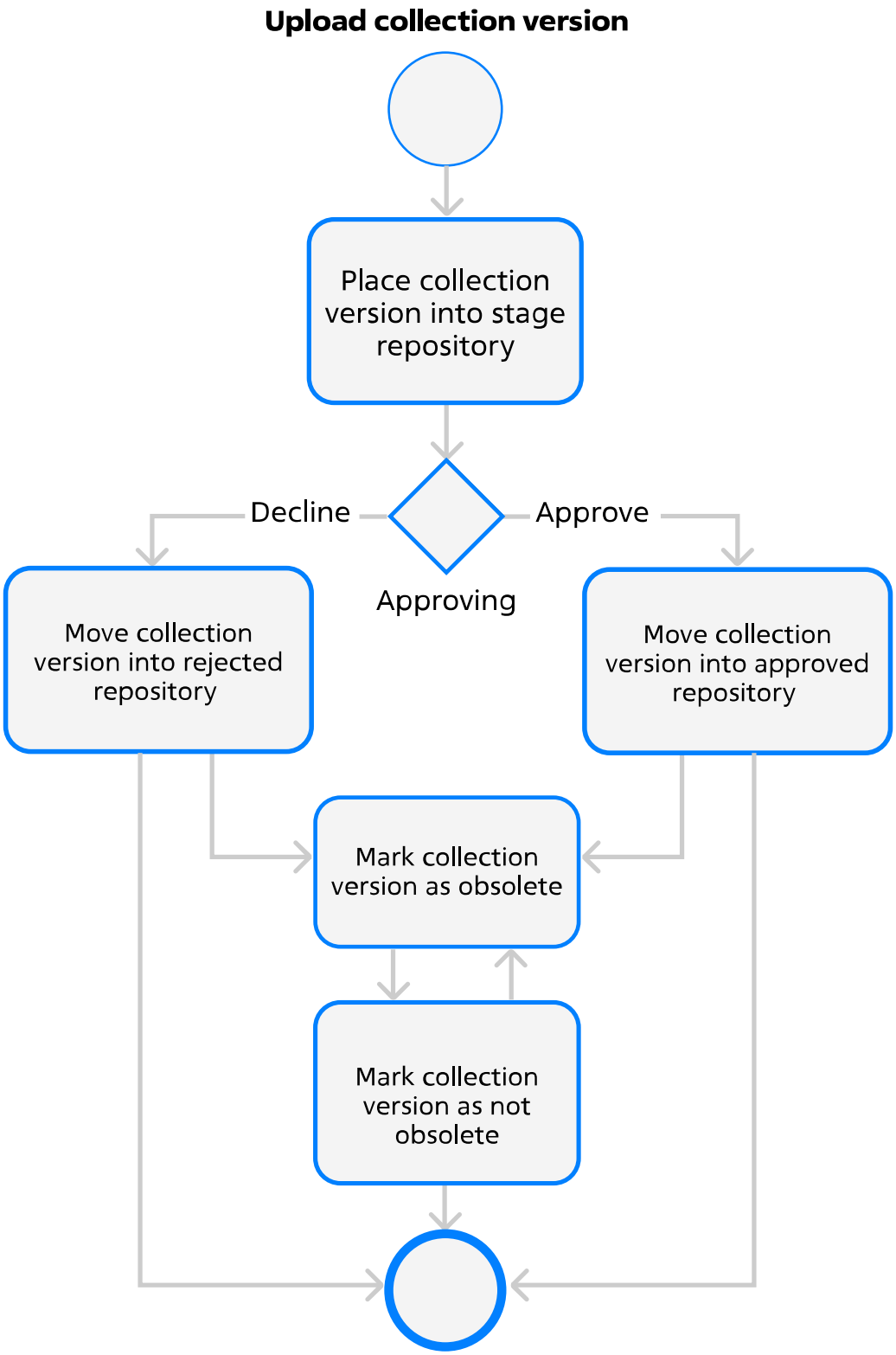
По умолчанию содержимое репозиториев доступно для использования всем пользователям Private Automation Hub. Если отметить репозиторий как приватный, доступ к хранящимся в нем коллекциям будет только у пользователей, явно указанных в настройках репозитория.

Жизненный цикл коллекции

Private Automation Hub позволяет гибко настроить жизненный цикл коллекций и их версий:

- версии коллекций могут быть доступны пользователям сразу или после *согласования*;
- версия коллекции может быть отмечена как устаревшая.

Жизненный цикл коллекции с учетом этапа согласования показан на схеме:



1. Версия коллекции загружается вручную разработчиком или автоматически самим Private Automation Hub при синхронизации с внешним источником коллекций. В обоих случаях версия коллекции помещается в репозиторий с меткой `pipeline: staging`.
2. Версия коллекции проходит согласование.

   Если версия коллекция согласована, она перемещается в соответствующий репозиторий с меткой `pipeline: approved`.

   Если версия коллекции отклонена, она перемещается в защищенный репозиторий `rejected`.

Этап согласования пропускается, если у соответствующего репозитория нет метки `pipeline: approved`.

3. Независимо от того, в каком репозитории находится версия коллекции, она может быть отмечена как устаревшая. С устаревшей коллекции может быть снята отметка об устаревании.
4. Любая версия любой коллекции может быть удалена из Private Automation Hub.

Конвейер задач

В настройках репозитория может быть задан *конвейер задач* – порядок публикации версии коллекции для пользователей:

- **Согласовано** (Approved) – версия коллекции попадает в репозиторий только после согласования.
- **Промежуточный вариант** (Staging) – версия коллекции из этого репозитория недоступна пользователям и ожидает согласования.
- **Ничего** (None) – версия коллекции доступна пользователям без согласования.

Метки

Метки управляют доступностью коллекций для пользователей.

- `pipeline: approved` – версии коллекций доступны пользователям.

  Эта метка назначается автоматически при выборе конвейера задач **Согласовано** (Approved). Напрямую загружать версии коллекций в репозитории с этой меткой нельзя – сначала их нужно разместить в промежуточном репозитории и согласовать.

- `pipeline: rejected` – версии коллекций недоступны пользователям, так как были отклонены при согласовании.

  Эта метка назначена защищенному репозиторию `rejected` и не может быть присвоена другим репозиториям.

- `pipeline: staging` – версии коллекций недоступны пользователям и ожидают согласования.

  Эта метка присваивается автоматически при выборе конвейера задач **Промежуточный вариант** (Staging). Из репозитория с этой меткой согласованные версии коллекций попадают в репозиторий с меткой `pipeline: approved`. Отклоненные версии попадают в защищенный репозиторий с меткой `pipeline: rejected`.

- `hide_from_search` – версии коллекции не выводятся в результатах поиска.

  Эта метка может быть присвоена при выборе конвейера задач **Согласовано** (Approved) или **Ничего** (None). При выборе конвейера задач **Промежуточный вариант** (Staging) эта метка присваивается автоматически.

10.2.3 Пространства имен

Для удобства управления коллекциями и предотвращения коллизий названий в Private Automation Hub используются *пространства имен*.

Например, коллекции `community.postgresql` и `astra.postgresql` относятся к пространствам имен `community` и `astra` соответственно.

Требования к названию пространств имен:

- длина не менее двух знаков;
- может содержать буквы латинского алфавита, цифры и символ _;
- не может начинаться с символа _.

10.2.4 Образы среды исполнения

Образы среды исполнения могут быть загружены в Private Automation Hub из других реестров образов или напрямую с рабочих станций разработчиков контента Ansible.

Коллекции

Private Automation Hub предъявляет к коллекциям следующие требования:

- Название:
  - длина не менее двух знаков;
  - может содержать буквы латинского алфавита, цифры и символ _;
  - не может начинаться с символа _.
- Коллекция должна иметь определенную структуру файлов и каталогов (см. *Коллекции Ansible*).
- Семантическое версионирование[104].

К коду коллекций, хранящихся в репозиториях Git, предъявляются следующие требования:

- Номера версий должны указываться в тегах[105].
- Тег **должен** содержать номер версии, например, `1.0.1`.
- Тег **может** содержать префикс v, например, `v1.0.1`.

Разграничение доступа

В соответствии с *ролевой моделью доступа* привилегии пользователям Private Automation Hub предоставляют путем назначения им напрямую или через команды пользователей определенных ролей на следующие типы ресурсов:

- **Репозитории** (Repositories).
- **Внешние репозитории** (Remotes).
- **Среды исполнения** (Execution Environments).
- **Пространства имен** (Namespaces).
- **Система** (System) – роли общего уровня.

Графический интерфейс позволяет назначать пользователю роли по отношению к отдельному ресурсу следующими способами:

- В окне конкретного ресурса. Инструкции по предоставлению доступа см. в описании конкретного ресурса. Например, чтобы назначить пользователю доступ к пространству имен, выполните действия, представленные в *описании пространства имен*.
- В окне *Пользователи (Users)*.

Создание ключа подписи контента

Ключи подписи обеспечивают проверку подлинности и целостности контента, который загружается в Private Automation Hub и используется для выполнения автоматизации. Приватный ключ применяется для подписания коллекций, а соответствующий публичный ключ – для проверки подписи системой Private Automation Hub и другими узлами.

Предупреждение

Приватный ключ должен храниться только на сервере, выполняющем подпись. Его компрометация позволит злоумышленникам выпускать доверенные (подписанные) пакеты. Никогда не передавайте приватный ключ третьим лицам.

[104] https://semver.org/
[105] https://git-scm.com/docs/git-tag

Для создания ключа выполните следующие действия:

1. На любой машине создайте пару ключей GPG с помощью команды:

```
gpg --full-generate-key
```

   В результате выполнения команды запускается мастер создания ключа, который задает несколько интерактивных вопросов.

   При ответе на вопросы придерживайтесь следующих рекомендаций:

   - тип ключа – RSA and RSA;
   - длина – 4096 бит;
   - срок действия – ограниченный, например, 2 года;
   - используйте пароль для дополнительной защиты.

2. После генерации ключа убедитесь, что он появился в локальном хранилище с помощью команды:

```
gpg --list-keys
```

3. Экспортируйте публичный и приватный ключи:

```
gpg --armor --export <your_email> > public_key.asc \
gpg --armor --export-secret-keys <your_email> > private_key.asc
```

4. Скопируйте файл `public_key.asc` на установочный узел.

5. В секцию `[all:vars]` файла `inventory` добавьте следующие параметры:

```
automationhub_create_default_collection_signing_service='true'
automationhub_auto_sign_collections='true'
automationhub_require_content_approval='true'
automationhub_collection_signing_service_key='/home/<user>/public_key.asc'
automationhub_collection_signing_service_script='/home/<user>/collection_signing.sh'
```

6. Создайте файл `/home/<user>/collection_signing.sh` со следующим содержимым:

```
#!/usr/bin/env bash

FILE_PATH=$1
SIGNATURE_PATH="$1.asc"

ADMIN_ID="$PULP_SIGNING_KEY_FINGERPRINT"
PASSWORD="<your_gpg_password>"

GPG_OUTPUT=$(gpg --quiet --batch --pinentry-mode loopback --yes --passphrase \
   "$PASSWORD" --homedir ~/.gnupg/ --detach-sign --default-key "$ADMIN_ID" \
   --armor --output "$SIGNATURE_PATH" "$FILE_PATH" 2>&1)
STATUS=$?

if [ $STATUS -eq 0 ]; then
   echo "{\"file\": \"$FILE_PATH\", \"signature\": \"$SIGNATURE_PATH\"}"
else
   echo "GPG signing failed: $GPG_OUTPUT" >&2
   exit $STATUS
fi
```

7. Сделайте файл исполняемым:

```
chmod +x /home/<user>/collection_signing.sh
```

8. Скопируйте файл `private_key.asc` на узел Private Automation Hub и выполните следующую команду, чтобы импортировать его в хранилище ключей GPG, используемое Private Automation Hub:

```
sudo -iu pulp gpg --import /tmp/private_key.asc
```

9. На установочном узле платформы в каталоге `/opt/rbta/aa/astra-automation-setup/` выполните команду:

```
sudo ./aa-setup
```

10.3 Применение Private Automation Hub

Применение Private Automation Hub включает следующие типовые операции:

- *Настройка Private Automation Hub на получение содержимого из сторонних реестров*, в том числе Automation Hub[106].
- *Настройка проектов Ansible на работу с Private Automation Hub* независимо от способа управления задачами автоматизации.
- *Публикация образов служебных контейнеров*.
- *Публикация коллекций Ansible с рабочих станций разработчиков*.

10.3.1 Синхронизация коллекций

Синхронизация Private Automation Hub с другими реестрами коллекций позволяет решить следующие задачи:

- сокращение времени загрузки коллекций в Automation Controller и на рабочие места пользователей Ansible;
- поддержание актуальности версий коллекций в Private Automation Hub;
- тонкая настройка доступности версий коллекций.

В результате синхронизации в Private Automation Hub будут доступны коллекции, распространяемые через сторонние реестры. Например, вы можете настроить синхронизацию Private Automation Hub с реестром Automation Hub[107], чтобы всегда иметь под рукой необходимые версии коллекций, созданных ПАО Группа Астра и ее партнерами.

Для настройки синхронизации коллекций с внешним реестром необходимо выполнить описанные далее действия.

Примечание

В изолированных сетях без доступа к интернету такой способ синхронизации невозможен. Для этого случая предусмотрена *загрузка коллекций из среды исполнения*.

Получение полномочий

Некоторые реестры коллекций, например, Ansible Galaxy[108], не требуют полномочий для доступа к содержимому. Другие предоставляют доступ только авторизованным пользователям. В этом случае для синхронизации коллекций необходимо получить соответствующие полномочия.

- Авторизация в Automation Hub осуществляется по токену. Для его создания воспользуйтесь *инструкцией*.
- Прочие реестры коллекций могут использовать для авторизации имя пользователя и пароль, токен, ключи SSH и тому подобное. Для получения необходимых полномочий обратитесь к владельцу соответствующего реестра.

[106] https://hub.astra-automation.ru/
[107] https://hub.astra-automation.ru/
[108] https://galaxy.ansible.com/

Private Automation Hub может использоваться как источник коллекций. Для авторизации в нем можно использовать следующие способы:

- токен;
- имя пользователя и пароль.

Настройка синхронизации

Чтобы Private Automation Hub мог получать коллекции из стороннего реестра коллекций, выполните следующие действия:

1. Согласно *инструкции* добавьте внешний репозиторий со следующими свойствами:
   - **Название** (Name): произвольное название внешнего репозитория, например, `Automation Hub`.
   - **URL**: URL репозитория во внешнем реестре, например:

     ```
     https://hub.example.com/api/galaxy/content/<repository>/
     ```

     Важно

     URL репозитория должен завершаться символом `/`.

     Здесь <repository> – название репозитория во внешнем реестре.

     По умолчанию коллекции загружаются из репозитория `published`. В этом случае допускается указать сокращенный URL:

     ```
     https://hub.example.com/api/galaxy/
     ```

   - **Требования YAML** (YAML requirements) – если вы хотите синхронизировать все доступные коллекции, оставьте это поле пустым. Чтобы синхронизировать только определенные коллекции, укажите их названия, например:

     ```
     ---
     collections:
       - name: astra.ald_pro
       - name: astra.ceph
       - name: astra.nginx
     ```

   Также укажите полномочия, используемые для авторизации во внешнем реестре. При настройке синхронизации Private Automation Hub с реестром Automation Hub заполните поле **Токен**, а поля **Имя пользователя** и **Пароль** оставьте пустыми.
2. Согласно *инструкции* создайте репозиторий, в котором будут размещаться коллекции, полученные из стороннего реестра. В поле **Внешний репозиторий** (Remote) выберите внешний репозиторий, созданный на предыдущем шаге.

Синхронизация

Следуя *инструкции* запустите синхронизацию репозитория с внешним источником. Дождитесь перехода задания синхронизации в статус **Успешно** (Success).

Совет

Для отслеживания процесса синхронизации в реальном времени выполните следующие действия:

1. На панели навигации выберите пункт **Управление задачами** (Task management).
2. Нажмите на ссылку с названием задачи синхронизации.

Сообщения о ходе синхронизации выводятся на панели **Сообщения о ходе исполнения** (Progress messages).

Публикация

Чтобы загруженные коллекции были доступны пользователям, разместите их в репозитории со следующими свойствами:

- Присвоена *метка* `pipeline: approved`.

  Если версии коллекций загружены в репозиторий с меткой `pipeline: staging`, проведите согласование.

  Если версии коллекций загружены в репозиторий без меток, измените его конвейер задач на **Согласовано** (Approved). Метка `pipeline: approved` будет присвоена автоматически.

- Включена настройка **Создать дистрибутив** (Create distribution).

Совет

В свойствах защищенных репозиториев `aa-certified`, `validated`, `community` и `published` эта настройка включена по умолчанию.

Проверка результатов синхронизации

Для проверки успешности синхронизации выполните следующие действия в Private Automation Hub и внешнем реестре:

1. На панели навигации выберите пункт *Коллекции* ▸ *Пространства имен* (*Collections* ▸ *Namespaces*).
2. Нажмите на ссылку с названием синхронизированного пространства имен.
3. Убедитесь, что выбранное пространство имен в Private Automation Hub и внешнем реестре содержит один и тот же набор коллекций.

Рекомендуемые настройки синхронизации с Automation Hub

В Private Automation Hub по умолчанию доступны репозитории `validated` и `aa-certified`. Чтобы настроить их на получение *проверенных* и *сертифицированных* коллекций из Automation Hub, выполните следующие действия:

1. Создайте токен для доступа к Automation Hub согласно *инструкции*.
2. В свойствах внешнего репозитория `aa-certified` укажите значение токена для доступа к Automation Hub.
3. Добавьте внешний репозиторий согласно *инструкции*:
   - **Название**: `aa-validated`.
   - **URL**:

     ```
     https://hub.astra-automation.ru/api/galaxy/content/validated/
     ```

   - **Токен**: укажите значение токена для доступа к Automation Hub.
4. Измените настройки репозитория `validated` согласно *инструкции*. В поле **Внешний репозиторий** (Remote) выберите внешний репозиторий `aa-validated`.

Совет

Изменять настройки репозитория `aa-certified` не нужно – он по умолчанию настроен на получение содержимого из внешнего репозитория `aa-certified`.

5. Запустите синхронизацию обоих репозиториев Private Automation Hub с репозиториями Automation Hub согласно *инструкции*.

Загрузка из среды исполнения

Добавлено в версии 1.2-upd1.

Образ `aa-full-ee` содержит полный перечень коллекций из пространства `astra` и других наиболее часто используемых коллекций. Этот набор обычно достаточен для начала использования Astra Automation. Чтобы загрузить коллекции из этого набора в приватный реестр, образ содержит утилиту `export_collections.py`.

Примечание

Этот способ идеально подходит для работы в изолированных сетях, не имеющих доступа к интернету, поскольку образ `aa-full-ee` включен в *Astra Automation offline bundle*.

Чтобы проверить список коллекций, доступных в образе `aa-full-ee`, выполните следующую команду:

```
podman run -u 0 hub.astra-automation.ru/aa-1.2/aa-full-ee:latest python3 export_
↪collections.py list
```

Загрузка коллекций выглядит следующим образом:

1. Убедитесь, что у вас есть доступ к приватному реестру с привилегиями, необходимыми для загрузки коллекций.
2. Убедитесь, что у вас есть токен доступа к API реестра. Если нет, то создайте его как описано в *настройке доступа к Automation Hub*.
3. Убедитесь, что в приватном реестре созданы пространства имен для загружаемых коллекций.
4. Загрузите требуемые коллекции из тех, которые доступны в образе, с помощью следующей команды:

   ```
   podman run \
       -u 0 \
       -e ANSIBLE_GALAXY_SERVER_GALAXY_URL=https://<IP_or_FQDN> \
       -e ANSIBLE_GALAXY_SERVER_VALIDATED_TOKEN=<API_token> \
       aa-full-ee:latest \
       python3 export_collections.py publish <collection_scope>
   ```

   В данной команде выполните следующие подстановки:

   - <IP_or_FQDN> – IP-адрес или доменное имя реестра коллекций;
   - <API_token> – токен доступа к API реестра коллекций;
   - <collection_scope> – определение загружаемых коллекций одним из способов:
     - `--collection <namespace>.<collection>` – указание пространства имен (<namespace>) и названия коллекции (<collection>) для загрузки одной коллекции, например `astra.ald_pro`;
     - `--namespace <namespace>` – указание пространства имен для загрузки всех коллекций, входящих в него, например `astra`;
     - `--all` – требование загрузить все доступные коллекции.

Подробнее об утилите `export_collections.py` см. в *справочнике*.

10.3.2 Настройка проекта на использование Private Automation Hub

Управлять задачами автоматизации можно различными способами, например, с применением *Automation Controller* и *Ansible Navigator*. В обоих случаях для получения коллекций из внешних источников используется утилита `ansible-galaxy`. С настройками по умолчанию она может загружать коллекции только из реестра Ansible Galaxy[109].

Для загрузки коллекций из Private Automation Hub необходимо выполнить ряд дополнительных настроек.

Совет

Использование образов среды исполнения, содержащих все необходимые коллекции, является более эффективным способом управления зависимостями Ansible, чем их загрузка из реестров коллекций. Для создания собственного образа среды исполнения, содержащего коллекции из Private Automation Hub, следуйте *инструкции*.

Предварительные настройки

Предварительно в Private Automation Hub необходимо *создать токен доступа* для проектов.

Важно

Период действия токена по умолчанию – 24 часа. По истечении указанного времени его нужно будет создать заново.

При создании нового токена предыдущий перестает действовать немедленно.

Указание реквизитов

При управлении задачами автоматизации с помощью *Ansible Navigator* укажите в файле `ansible.cfg` реквизиты доступа к Private Automation Hub:

1. В секцию `[galaxy]` добавьте параметр `server_list`. В его значении одной строкой через запятую укажите список условных названий репозиториев, из которых будет загружаться содержимое, например:

```
[galaxy]
server_list = my_private_hub, galaxy
```

Важно

Особенности заполнения параметра `server_list`:

- Допускается использование пробела после запятой.
- Указанный список заменяет собой список реестров по умолчанию. Это значит, что для использования реестра Ansible Galaxy[110] нужно будет выполнить следующие дополнительные действия:
  1. Указать в значении параметра `server_list` условное название реестра Ansible Galaxy.
  2. Создать дополнительную секцию с описанием настроек подключения к Ansible Galaxy.

2. По шаблону создайте и заполните секции для всех упомянутых серверов:

[109] https://galaxy.ansible.com/
[110] https://galaxy.ansible.com/

```
[galaxy_server.<name>]
url = <url>

# При авторизации с использованием токена:
token = <token>

# При авторизации по названию учетной записи пользователя и паролю:
username = <username>
password = <password>
```

Здесь:

- <name> – условное название репозитория, указанное в `server_list`.
- `url` – URL репозитория.

  Важно

  URL репозитория должен завершаться символом `/`.

  По умолчанию коллекции загружаются из репозитория `published`. В этом случае допускается указать сокращенный URL:

  ```
  https://hub.example.com/api/galaxy/
  ```

  Чтобы получить доступ к репозиторию, отличному от `published`, приведите URL к следующему виду:

  ```
  https://hub.example.com/api/galaxy/content/<repository>/
  ```

  Здесь <repository> – название репозитория в Private Automation Hub.
- <token> – значение токена, используемого для доступа к реестру.
- <username> – название учетной записи пользователя реестра.
- <password> – пароль пользователя реестра.

Совет

Для доступа к реестру Ansible Galaxy[111] указание учетных данных не требуется.

Создание полномочия

При управлении задачами автоматизации с помощью Automation Controller достаточно создать полномочие типа *API-токен Ansible Galaxy/Automation Hub* и связать его с организацией.

Хотя вы все еще можете добавить в проект файл `ansible.cfg`, созданный согласно приведенной выше инструкции, использование полномочия имеет ряд преимуществ:

- не требуется добавление в файл `ansible.cfg` учетных данных в открытом виде или настройка Automation Controller на работу с *внешней системой управления секретами*;
- при устаревании или отзыве токена не нужно вносить изменения в файлы проекта;
- действие полномочия распространяется на всех пользователей организации.

Для получения и установки зависимостей из Private Automation Hub выполните следующие действия:

1. В корневом каталоге проекта создайте файл зависимостей `requirements.yml`, например:

[111] https://galaxy.ansible.com/

```
---
collections:
  - name: astra.rupost
    version: 0.14.4
  - name: astra.nginx
    version: 1.8.2
  - name: astra.postgresql
    version: 3.0.0
```

2. Согласно *инструкции* создайте полномочие типа *API-токен Ansible Galaxy/Automation Hub*. Поля формы в секции **Сведения о типе** (Type Details) заполните следующим образом:

   - **URL сервера Galaxy** (Galaxy Server URL) – URL репозитория в Private Automation Hub.

     Важно

     Указанный URL должен завершаться символом /.

     По умолчанию коллекции загружаются из репозитория published. В этом случае допускается указать сокращенный URL:

     ```
     https://hub.example.com/api/galaxy/
     ```

     Чтобы получить доступ к репозиторию, отличному от published, приведите URL к следующему виду:

     ```
     https://hub.example.com/api/galaxy/content/<repository>/
     ```

     Здесь <repository> – название репозитория в Private Automation Hub.

   - **Токен API** (API Token): содержимое токена.

3. Если для защиты подключения к Private Automation Hub используется самоподписанный сертификат, в настройках *включите игнорирование результатов проверки сертификатов SSL* при работе с реестрами коллекций.

   Если эта настройка выключена, проверка сертификатов при загрузке и установке зависимостей проекта будет завершаться ошибкой:

```
[WARNING]: Skipping Galaxy server https://hub.example.com/api/galaxy/content/aa-
↪validated/.
Got an unexpected error when getting available versions of
collection astra.chrony: Unknown error when attempting to call Galaxy at
'https://hub.example.com/api/galaxy/content/aa-validated/api': <urlopen error
[SSL: CERTIFICATE_VERIFY_FAILED] certificate verify failed: self signed
certificate (_ssl.c:1129)>. <urlopen error [SSL: CERTIFICATE_VERIFY_FAILED]
certificate verify failed: self signed certificate (_ssl.c:1129)>
ERROR! Failed to resolve the requested dependencies map. Could not satisfy the␣
↪following requirements:
* astra.chrony:* (direct request)
```

4. Согласно *инструкции* измените свойства организации, добавив в список **Учетные данные Galaxy** (Galaxy Credentials) созданное полномочие.

5. Синхронизируйте проект с источником согласно *инструкции*. Automation Controller использует связанное с организацией полномочие для загрузки необходимых коллекций из Private Automation Hub.

Пример

Пусть проект использует коллекции из следующих реестров:

- Ansible Galaxy[112];
- Private Automation Hub, доступный по адресу `https://private-hub.example.com`.

  При этом необходимо настроить доступ к следующим репозиториям:

  - `published` – репозиторий по умолчанию;
  - `testing` – коллекции, не прошедшие полное тестирование;
  - `automation-hub` – коллекции, полученные из Automation Hub при *синхронизации*.

Чтобы настроить проект на работу со всеми указанными реестрами и репозиториями, содержимое `ansible.cfg` необходимо привести к следующему виду:

```
[galaxy]
server_list = galaxy, hub_published, hub_testing, automation_hub

[galaxy_server.galaxy]
url = https://galaxy.ansible.com/

[galaxy_server.hub_published]
url = https://private-hub.example.com/api/galaxy/
token = b23...41e

[galaxy_server.hub_testing]
url = https://private-hub.example.com/api/galaxy/content/testing/
token = b23...41e

[galaxy_server.automation_hub]
url = https://private-hub.example.com/api/galaxy/content/automation-hub/
token = b23...41e
```

При использовании Automation Controller достаточно описать зависимости в файле `requirements.yml` и создать три полномочия типа *API-токен Ansible Galaxy/Automation Hub*, по одному на каждый репозиторий.

10.3.3 Загрузка образов среды исполнения

Загрузка образов среды исполнения в Private Automation Hub позволяет решить следующие задачи:

- сокращение времени загрузки образов среды исполнения на узлы Automation Controller и рабочие места пользователей Ansible;
- предоставление пользователям образов среды исполнения, разработанных собственными силами.

В результате загрузки образов в Private Automation Hub они будут доступны пользователям точно так же, как если бы они были загружены в один из публичных реестров, например, Docker Hub[113] или Quay.io[114].

Загрузка образов может быть выполнена из внешних реестров образов или напрямую с рабочих станций разработчиков контента Ansible.

Загрузка из внешнего реестра

Для загрузки образов из внешнего реестра выполните следующие действия:

1. Согласно *инструкции* создайте запись о внешнем реестре (remote registry).

[112] https://galaxy.ansible.com/
[113] https://hub.docker.com
[114] https://quay.io/

2. Согласно *инструкции* создайте запись о среде исполнения, содержащую следующие сведения:
   - внешний реестр (выберите реестр, созданный на предыдущем шаге);
   - название пространства имен контейнеров;
   - название образа;
   - список меток, по которым выполняется отбор версий образа.

     В названиях меток допускается использование подстановочного символа *.
3. Согласно *инструкции* запустите синхронизацию среды исполнения с внешним реестром.

Пример

Чтобы загрузить из реестра Automation Hub[115] все образы `aa-full-ee` ветки `0.1.*`, кроме `0.1.0`, выполните следующие действия:

1. Создайте запись о внешнем реестре со следующим URI:

   ```
   https://hub.astra-automation.ru/
   ```

2. Создайте запись о среде исполнения со следующими свойствами:
   - **Внешний реестр** (Registry): выберите реестр, созданный на предыдущем шаге.
   - **Название вышестоящего объекта** (Upstream name): `aa-1.2/aa-full-ee`.
   - **Текущие включенные метки** (Currently included tags): `0.1.*`.
   - **Текущие исключенные метки** (Currently excluded tags): `0.1.0`.

Загрузка с рабочей станции

Для загрузки образа среды исполнения с рабочей станции выполните на ней следующие действия:

1. Предоставьте Podman учетные данные для доступа к Private Automation Hub:

   ```
   podman login -u=<username> -p=<password> <hub>
   ```

   Здесь:
   - <username> – название учетной записи пользователя Private Automation Hub.

     Указанный пользователь должен обладать привилегиями на загрузку образов среды исполнения.
   - <password> – пароль.
   - <hub> – доменное имя Private Automation Hub.

   В случае успешной авторизации в терминал выводится следующая строка:

   ```
   Login Succeeded!
   ```

   Подробности о команде `login` см. в документации Podman[116].
2. Присвойте образу название и метку, с которыми он должен быть опубликован в Private Automation Hub:

   ```
   podman tag <image_local>:<label_local> <hub>/<image_hub>:<label_hub>
   ```

   Здесь:
   - <image_local> – название образа, собранного на рабочей станции;

[115] https://hub.astra-automation.ru/
[116] https://docs.podman.io/en/latest/markdown/podman-login.1.html

- <label_local> – метка образа, собранного на рабочей станции;
- <hub> – FQDN Private Automation Hub;
- <image_hub> – название, под которым образ должен быть опубликован в Private Automation Hub;
- <label_hub> – метка, с которой образ должен быть опубликован в Private Automation Hub.

3. Загрузите образ в Private Automation Hub:

```
podman push <hub>/<image_hub>:<tag_hub>
```

Здесь:

- <hub> – FQDN Private Automation Hub.
- <image_hub> – название, под которым образ должен быть опубликован в Private Automation Hub.
- <tag_hub> – метка, с которой образ должен быть опубликован в Private Automation Hub.

Подробности о команде push см. в документации Podman[117].

Пример

Private Automation Hub доступен по FQDN `hub.example.com`. На рабочей станции разработчика собран образ среды исполнения с названием `ee-automation-image:latest`. В Private Automation Hub этот образ нужно опубликовать с названием и меткой `custom-ee-image:1.0.0`.

1. Авторизуйтесь в Private Automation Hub:

```
podman login -u=<user> -p=<password> hub.example.com
```

2. Присвойте образу название и метку, с которыми он должен быть опубликован в Private Automation Hub:

```
podman tag ee-automation-image:latest hub.example.com/custom-ee-image:1.0.0
```

3. Опубликуйте образ в Private Automation Hub:

```
podman push hub.example.com/custom-ee-image:1.0.0
```

4. Убедитесь, что образ доступен в Private Automation Hub:
    1. После авторизации выберите на панели навигации *Среды исполнения* ▸ *Среды исполнения* (*Execution Environments* ▸ *Execution Environments*).
    2. Убедитесь, что в таблице сред исполнения появилась запись `custom-ee-image`.
    3. Нажмите на ссылку `custom-ee-image`.
    4. Выберите вкладку **Образы** (Images) и убедитесь в наличии образа с меткой `1.0.0`.

10.3.4 Размещение коллекций

Размещение коллекций в Private Automation Hub состоит из следующих этапов:

1. Загрузка.

    На этом этапе версия коллекции с рабочей станции разработчика загружается в хранилище Private Automation Hub и попадает в один из репозиториев. Загрузку можно выполнить следующими способами:

    - *утилита командной строки ansible-galaxy*;

[117] https://docs.podman.io/en/latest/markdown/podman-push.1.html

- *утилита export_collections.py*;
- *веб-интерфейс*.

2. Согласование.

   Этот этап обязателен, если версия коллекции загружена в репозиторий с конвейером задач **Промежуточный вариант** (Staging).

3. Публикация.

   Если версия коллекции прошла согласование, она размещается в репозитории с конвейером задач **Согласовано** (Approved) и становится доступной пользователям.

   Версии коллекции, загруженные в репозитории с конвейером **Ничего** (None), доступны пользователям сразу.

Подробности о жизненном цикле коллекций см. в *соответствующей секции*.

Настройка Private Automation Hub

Примечание

Для выполнения этого этапа необходимы привилегии администратора Private Automation Hub.

Для каждой команды разработчиков коллекций Ansible выполните следующие действия:

1. Создайте пространство имен. Для этого следуйте *инструкции*.
2. Создайте приватный репозиторий. Для этого следуйте *инструкции*.

Перед началом работы

Авторизуйтесь в Private Automation Hub с учетной записью пользователя-разработчика коллекций Ansible и выполните следующие действия:

1. Создайте токен для доступа к Private Automation Hub.

   Для этого следуйте *инструкции*.

2. Используйте утилиту `ansible-galaxy` для создания архива с файлами коллекции. Для этого в корневом каталоге проекта выполните команду:

   ```
   ansible-galaxy collection build
   ```

   При успешной сборке `ansible-galaxy` создает в корневом каталоге проекта архив с расширением `.tar.gz`.

Загрузка версии коллекции с помощью `ansible-galaxy`

Чтобы загрузить версию коллекции в Private Automation Hub с помощью `ansible-galaxy`, выполните команду:

```
ansible-galaxy collection publish \
   --server <PAH_URI>/api/galaxy/<repository> \
   --token <token> \
   /path/to/collection.tar.gz
```

Здесь:

- <PAH_URI> – URI Private Automation Hub.
- <repository> – название репозитория.

  Если название репозитория не указано, коллекция загружается в промежуточный репозиторий по умолчанию.

Важно

Нельзя загружать версии коллекций напрямую в репозитории с конвейером задач **Согласовано** (Approved).

- <token> – токен для доступа к Private Automation Hub.
- `/path/to/collection.tar.gz` – путь к архиву с версией коллекции.

Загрузку версий коллекций можно упростить, если указать настройки репозиториев в конфигурационном файле `ansible.cfg`:

1. Создайте секцию `[galaxy]` и добавьте в нее переменную `server_list`, в значении которой укажите названия репозиториев, например:

```
[galaxy]
server_list = private_hub_publish, private_hub_testing, dev_ops_hub
```

   Здесь `private_hub_publish`, `private_hub_testing` и `dev_ops_hub` – названия репозиториев.

2. Создайте секции, содержащие настройки для каждого репозитория, например:

```
[galaxy_server.private_hub_publish]
url = https://private-hub.example.com/api/galaxy/publish/
token = <token_private_hub>

[galaxy_server.private_hub_testing]
url = https://private-hub.example.com/api/galaxy/testing/
token = <token_private_hub>

[galaxy_server.dev_ops_hub]
url = https://dev-ops-hub.example.com/api/galaxy/
token = <token_dev_ops_hub>
```

   Здесь в переменных `url` указаны ссылки на репозитории, при этом:

   - Для Private Automation Hub `https://private-hub.example.com/` указаны настройки для репозиториев `publish` и `testing`. Это позволяет использовать `ansible-galaxy` для прямой загрузки версий коллекций в указанные репозитории. Доступ к обоим репозиториям осуществляется с использованием одного и того же токена <token_private_hub>.
   - Для Private Automation Hub `https://dev-ops-hub.example.com/` ссылка не содержит названия репозитория. Все загружаемые версии коллекций будут автоматически попадать в промежуточный репозиторий по умолчанию. Доступ к репозиторию осуществляется с использованием токена <token_dev_ops_hub>.

3. Для загрузки версии коллекции выполните команду:

```
ansible-galaxy collection publish --server <repository_name> /path/to/collection.
↪tar.gz
```

   Здесь <repository_name> – название репозитория, указанное в `ansible.cfg`.

Совет

Если в `ansible.cfg` заданы настройки только для одного репозитория, указывать его в аргументах `ansible-galaxy` не обязательно:

```
ansible-galaxy collection publish /path/to/collection.tar.gz
```

Загрузка версии коллекции через веб-интерфейс

Чтобы загрузить в Private Automation Hub версию коллекции через веб-интерфейс, воспользуйтесь соответствующей *инструкцией*.

Публикация

Если версия коллекции загружена в репозиторий с конвейером задач **Промежуточный вариант** (Staging), выполните согласование согласно *инструкции*.

Если версия коллекции загружена в репозиторий с конвейером задач **Ничего** (None), дополнительных действий не требуется.

10.4 Графический интерфейс

Раздел **Контент автоматизации** (Automation Content) обеспечивает централизованное управление всеми артефактами автоматизации в Private Automation Hub:

- организация и публикация контента в пространствах имен;
- хранение и распространение коллекций;
- хранение и распространение образов сред исполнения и сред принятия решений;
- настройка ключей подписей для проверки целостности;
- подключение к внешним реестрам и удаленным источникам;
- утверждение и согласование коллекций перед публикацией;
- управление токенами API для интеграции с внешними системами.

Ниже описаны основные разделы интерфейса **Контент автоматизации** (Automation Content).

10.4.1 Пространства имен

Раздел **Пространства имен** (Namespace) содержит вкладки на которых расположены таблицы с информацией о пространствах имен коллекций Ansible.

В разделе можно выполнять следующие действия:

- создавать пространства имен для организации коллекций;
- управлять доступом для команд и пользователей;
- добавлять описание, логотип и ссылки на документацию;
- публиковать и обновлять коллекции в рамках пространства.

После создания пространства имен пользователи, обладающие правами, могут публиковать и обновлять коллекции в его рамках. Это упрощает управление контентом, обеспечивает структурированность хранилища и разделение ответственности между командами.

10.4.2 Коллекции

Раздел **Коллекции** (Collections) используется для управления *коллекциями Ansible*.

В разделе можно выполнять следующие действия:

- просматривать список доступных коллекций;
- публиковать новые и обновлять существующие версии коллекций;
- утверждать или отклонять коллекции перед распространением;
- управлять зависимостями и метаданными.

После утверждения коллекции становятся доступными для скачивания и применения в автоматизации вашей организации. Раздел поддерживает как работу с коллекциями разработчиков, так и с внешними сертифицированными источниками.

10.4.3 Среды исполнения

Раздел **Среды исполнения** (Execution Environments) содержит список образов сред исполнения и сред принятия решений (далее – образов).

В разделе можно выполнять следующие действия:

- создавать и импортировать образы;
- просматривать и редактировать сведения образов;
- удалять загруженные образы;
- настраивать доступ пользователей и команд к образам.

10.4.4 Ключи подписи

Раздел **Ключи подписи** (Signature Keys) используется для просмотра криптографических ключей, обеспечивающих проверку целостности и подлинности коллекций и контейнерных образов.

В разделе можно выполнять следующие действия:

- просматривать список зарегистрированных ключей подписи;
- просматривать сведения о ключе;
- скачивать публичные ключи для проверки подлинности контента вне платформы.

Примечание

Создавать новые ключи или удалять существующие через графический интерфейс нельзя. Добавление или отзыв ключей выполняется при установке или обновлении платформы, либо с использованием командной строки и утилит GPG.

10.4.5 Репозитории

Раздел **Репозитории** (Repositories) используется для управления репозиториями Private Automation Hub.

В разделе можно выполнять следующие действия:

- просматривать сведения о существующих репозиториях;
- создавать новые репозитории;
- изменять настройки существующих репозиториев;
- управлять доступом к репозиториям;
- получать настройки для использования репозитория утилитами Ansible;
- синхронизировать содержимое репозиториев Private Automation Hub с внешними репозиториями;
- управлять версиями репозиториев;
- удалять репозитории.

10.4.6 Внешние реестры

Раздел **Внешние реестры** (Remote Registries) используется для интеграции с внешними реестрами контейнеров, например, Docker Hub.

В разделе можно выполнять следующие действия:

- добавлять записи о внешних реестрах контейнеров;
- синхронизировать списки образов с внешними реестрами контейнеров;
- удалять записи о внешних реестрах контейнеров.

10.4.7 Управление задачами

Раздел **Управление задачами** (Task Management) позволяет отслеживать системные задачи, выполняемые внутри Automation Hub.

В разделе можно выполнять следующие действия:

- просматривать сведения о выполняющихся и выполненных задачах;
- останавливать выполняющиеся задачи.

Раздел особенно полезен администраторам для контроля активности и устранения ошибок.

10.4.8 Согласования коллекций

Раздел **Согласования коллекций** (Collection Approvals) используется для проверки и утверждения коллекций перед публикацией.

В разделе можно выполнять следующие действия:

- просматривать коллекции, ожидающие утверждения;
- согласовывать или отклонять их по результатам проверки.

Это гарантирует, что только проверенный и безопасный контент попадет в рабочие репозитории организации.

10.4.9 Внешние репозитории

Раздел **Внешние репозитории** (Remotes) используется для управления внешними источниками в Private Automation Hub, например: Ansible Galaxy, Automation Hub, собственные корпоративные репозитории.

В разделе можно выполнять следующие действия:

- просматривать подробные сведения о внешнем репозитории;
- добавлять новые внешние источники;
- редактировать основные сведения о внешних репозиториях;
- предоставлять доступ к записям о внешних репозиториях для команд и отдельных пользователей.

10.4.10 Подключение к Private Automation Hub

Раздел **Подключение к Private Automation Hub** (API Token) используется для создания токена, позволяющего авторизоваться в Private Automation Hub.

Пространства имен

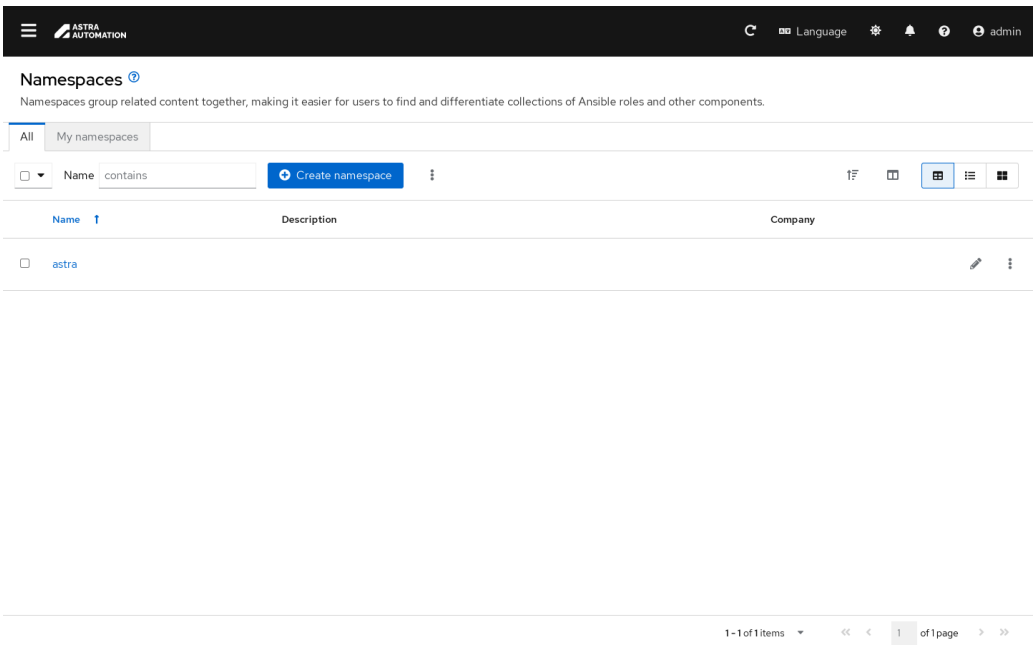
Окно **Пространства имен** (Namespaces) предназначено для централизованного управления *пространствами имен коллекций Ansible* с помощью следующих операций:

- организация структуры контента по проектам и командам;
- назначение владельцев и политики доступа к коллекциям;
- контроль публикаций и распространение контента внутри организации.

Для перехода к окну **Пространства имен** (Namespaces) выберите на панели навигации *Контент автоматизации* ▸ *Пространства имен* (*Automation Content* ▸ *Namespaces*).

Описание окна

Внешний вид окна **Пространства имен** (Namespaces) представлен на схеме:



Окно **Пространства имен** (Namespaces) состоит из двух вкладок:

- **Все** (All) – все пространства имен, существующие в Private Automation Hub.
- **Мои пространства имен** (My namespaces) – пространства имен, доступ к которым предоставлен активному пользователю непосредственно или через команды, в которых он состоит.

В обеих вкладках выводится таблица, содержащая следующие столбцы:

- флаги для выбора нескольких записей;
- **Название** (Name) – ссылка на окно просмотра сведений о пространстве имен;
- **Описание** (Description);
- **Компания** (Company) – компания, связанная с пространством имен;
- кнопки вызова часто выполняемых действий:
  - переход в окно редактирования свойств пространства имен;
  - переход к журналу импортов коллекций в пространство имен;
  - удаление пространства имен.

Просмотр информации о пространстве имен

Для просмотра подробной информации о пространстве имен нажмите на его название.

Окно просмотра сведений о пространстве имен состоит из следующих вкладок:

- **Подробности** (Details) – основная информация о пространстве имен.
- **Коллекции** (Collections) – таблица коллекций, имеющихся в пространстве имен.

> **Примечание**
>
> В списке отображаются только коллекции, прошедшие согласование.

- **Настройка CLI** (CLI configuration) – URL со ссылкой на Private Automation Hub. Эта ссылка может быть использована для настройки `ansible-galaxy` и Automation Controller на использование Private Automation Hub в качестве источника коллекций.
- **Командный доступ** (Team Access) – таблица команд и назначенных им *ролей*;

- **Доступ пользователей** (User Access) – таблица пользователей и назначенных им *ролей*.

Создание пространства имен

Для создания пространства имен выполните следующие действия:

1. В окне **Пространства имен** (Namespaces) нажмите кнопку *Создать пространство имен* (Create namespace).
2. Заполните форму **Создать пространство имен** (Create namespace):
   - **Название** (Name) – название пространства имен.

     Требования к названию пространств имен:

     - длина не менее двух знаков;
     - может содержать буквы латинского алфавита, цифры и символ _;
     - не может начинаться с символа _.

     Важно

     Будьте внимательны при выборе названия пространства имен – его нельзя будет изменить.

   - **Описание** (Description) – описание пространства имен.
   - **Компания** (Company) – компания, связанная с пространством имен.
   - **URL логотипа** (Logo URL) – ссылка на логотип пространства имен. Логотип по указанной ссылке будет загружен в Private Automation Hub при сохранении изменений.
   - **Ресурсы** (Resources) – дополнительные сведения о пространстве имен в формате Markdown.
   - **Полезные ссылки** (Useful links) – список ссылок, которые могут быть полезны при поиске информации о пространстве имен и хранящихся в нем коллекциях, например, сайт компании-разработчика.
3. Нажмите кнопку *Создать пространство имен* (Create namespace).
4. *Загрузите версии коллекций*.
5. *Настройте параметры CLI*.
6. *Настройте параметры доступа*.

Изменение свойств пространства имен

Для изменения основных сведений о пространстве имен выполните следующие действия:

1. В окне **Пространства имен** (Namespaces) нажмите на название пространства имен, сведения которого необходимо изменить.
2. В окне с информацией о пространстве имен нажмите кнопку *Редактировать пространство имен* (Edit namespace).
3. Во вкладке **Редактировать <название_пространства_имен>** (Edit <namespace_name>) отредактируйте необходимые сведения.
4. Нажмите кнопку *Сохранить пространство имен* (Save namespace).

Загрузка коллекций

Для загрузки версии коллекции в пространство имен выполните следующие действия:

1. В окне **Пространства имен** (Namespaces) нажмите на название пространства имен, в которое необходимо загрузить коллекцию.
2. Перейдите во вкладку и нажмите кнопку *Загрузить коллекцию* (Upload collection).
3. В открывшемся окне заполните форму:
   - **Файл коллекций** (Collection file) – файл в формате `.tar.gz`, сформированный с помощью команды `ansible-galaxy build`.
   - Переключатель доступных для выбора репозиториев:
     - **Тестовые репозитории** (Staging Repos) – для выбора будут доступны только промежуточные репозитории. Загруженная версия коллекции будет ожидать согласования.
     - **Хранилища без конвейера** (Repositories without pipeline) – для выбора будут доступны все репозитории, кроме имеющих метки `pipeline: approved` и `pipeline: rejected`.

   Важно

   Необходимы привилегии, позволяющие загружать версии коллекций в указанный репозиторий.
4. Нажмите кнопку *Загрузить коллекцию* (Upload collection).

Настройка CLI

Для настройки утилит командной строки на работу с Private Automation Hub выполните следующие действия:

1. В окне **Пространства имен** (Namespaces) нажмите на название пространства имен, которое необходимо настроить.
2. Выберите вкладку **Настройка CLI** (CLI configuration).
3. Скопируйте ссылку, отображаемую во вкладке.
4. Укажите параметры подключения к Private Automation Hub в конфигурационном файле `ansible.cfg` согласно *инструкции*.

При использовании Automation Controller *создайте полномочие* типа «API-токен Ansible Galaxy/Automation Hub».

Настройка доступа

Во вкладках **Доступ пользователей** (User Access) и **Командный доступ** (Team Access) выводятся списки пользователей или команд и назначенных им *ролей*. Порядок настройки одинаков:

1. В окне **Пространства имен** (Namespaces) нажмите на название пространства имен, доступ к которому необходимо настроить.
2. Выберите вкладку **Командный доступ** (Team Access) или **Доступ пользователей** (User Access).
3. Нажмите кнопку *Добавить роль* (Add roles).
4. В открывшемся окне выберите нужного пользователя или команду и нажмите кнопку *Далее* (Next).

5. Выберите роли, предоставляемые пользователю или команде:

Роль	Предоставляемые привилегии
`galaxy.collection_namespace_owner`	• Изменение свойств пространства имен • Загрузка коллекций
`galaxy.collection_publisher`	• Создание пространств имен • Изменение свойств пространства имен • Загрузка коллекций

6. Нажмите кнопку *Далее* (Next).
7. Во вкладке **Обзор** (Review) проверьте корректность настроек и нажмите кнопку *Завершить* (Finish).

Удаление пространства имен

Предупреждение

- При удалении пространства имен также удаляются все содержащиеся в нем коллекции.
- Это действие нельзя отменить.

Чтобы удалить пространство имен, выполните следующие действия:

1. В таблице установите флаги в строках с пространствами имен, которые необходимо удалить.
2. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите пункт *Удалить пространства имен* (Delete namespaces).
3. Подтвердите удаление.

Коллекции

Окно **Коллекции** (Collections) служит центральной точкой для работы с *коллекциями Ansible*, доступными в Private Automation Hub, с помощью следующих операций:

- просмотр опубликованных коллекций и их версий;
- изучение доступных ролей, модулей и расширений.

Для перехода к окну **Коллекции** (Collections) выберите на панели навигации *Контент автоматизации* ▸ *Коллекции* (*Automation Content* ▸ *Collections*).

Описание окна

Внешний вид окна **Коллекции** (Collections) представлен на схеме:

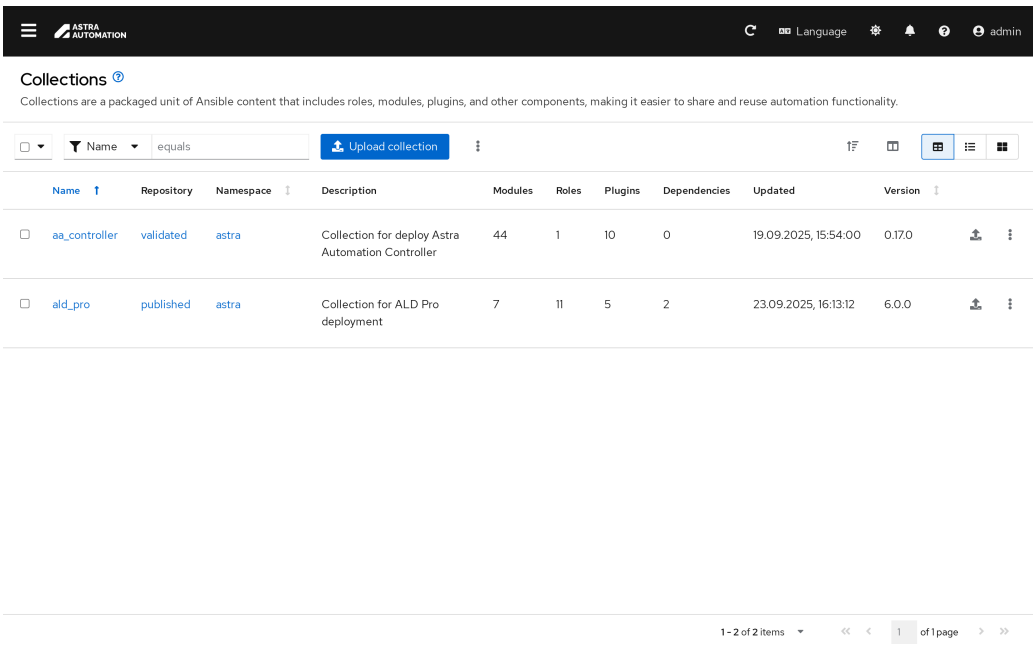


Таблица коллекций состоит из столбцов согласно настройке с помощью кнопки *Настроить столбцы* (Manage columns). Полный список столбцов:

- флаги для выбора нескольких записей;
- **Название** (Name) – ссылка на окно просмотра сведений о коллекции;
- **Предоставлено** (Provided by) – название организации или учетной записи пользователя, опубликовавших коллекцию;
- **Репозиторий** (Repository) – ссылка на окно просмотра сведений о репозитории, в котором опубликована коллекция;
- **Пространство имен** (Namespace) – ссылка на окно просмотра сведений о пространстве имен, в которое загружена коллекция;
- **Описание** (Description) – описание коллекции;
- **Модули** (Modules) – количество модулей в коллекции;
- **Роли** (Roles) – количество ролей в коллекции;
- **Расширения** (Plugins) – количество расширений в коллекции;
- **Зависимости** (Dependencies) – количество зависимостей коллекции;
- **Обновлено** (Updated) – дата и время последнего обновления коллекции;
- **Версия** (Version) – версия коллекции;
- **Подписанное состояние** (Signed state) – статус подписи коллекции, показывающий, была ли коллекция подписана для подтверждения ее подлинности и целостности;
- кнопки вызова часто выполняемых действий:
  - загрузка новой версии коллекции;
  - отмена коллекции как устаревшей;
  - копирование версии в репозитории;
  - удаление коллекции из репозитория;
  - удаление коллекции из системы.

Просмотр информации о коллекции

Для просмотра подробной информации о коллекции нажмите на ее название.

Окно просмотра сведений о коллекции состоит из выпадающего списка, позволяющего выбрать версию, и следующих вкладок:

- **Подробности** (Details) – подробные сведения о коллекции;
- **Установить** (Install) – инструкция по установке коллекции с использованием утилиты `ansible-galaxy`;
- **Документация** (Documentation) – документация коллекции, входящих в нее ролей и модулей;
- **Содержимое** (Contents) – краткая информация о модулях и ролях, предоставляемых коллекцией;
- **Журнал импорта** (Import log) – журнал импорта коллекции в Private Automation Hub;
- **Зависимости** (Dependencies) – список зависимостей коллекции;
- **Распределения** (Distributions) – список репозиториев, в которых размещена коллекция, а также краткая инструкция по настройке `ansible-galaxy` для их использования.

Загрузка коллекции

Для загрузки в Private Automation Hub версии коллекции выполните следующие действия:

1. В окне **Коллекции** (Collections) нажмите кнопку *Загрузить коллекцию* (Upload collection).
2. В открывшемся окне заполните форму:
   - **Файл коллекций** (Collection file) – файл в формате `.tar.gz`, сформированный с помощью команды `ansible-galaxy build`.
   - Переключатель доступных для выбора репозиториев:
     - **Тестовые репозитории** (Staging Repos) – для выбора будут доступны только промежуточные репозитории. Загруженная версия коллекции будет ожидать согласования.
     - **Хранилища без конвейера** (Repositories without pipeline) – для выбора будут доступны все репозитории, кроме имеющих метки `pipeline: approved` и `pipeline: rejected`.

     Важно

     Необходимы привилегии, позволяющие загружать версии коллекций в указанный репозиторий.
3. Нажмите кнопку *Загрузить коллекцию* (Upload collection).

Удаление коллекции или ее версии

Для удаления версии коллекции выполните следующие действия:

1. В окне **Коллекции** (Collections) нажмите на название коллекции, версию которой необходимо удалить.
2. В окне просмотра сведений о коллекции выберите в выпадающем списке версию, которую хотите удалить.
3. Нажмите кнопку ⋮ и в открывшемся меню выберите соответствующий пункт:
   - **Удалить всю коллекцию из системы** (Delete entire collection from system) – удаление всех версий коллекции из Private Automation Hub.

- **Удалить всю коллекцию из репозитория** (Delete entire collection from repository) – удаление записи о коллекции из репозитория.
- **Удалить версию из системы** (Delete version from system) – удаление из Private Automation Hub только выбранной версии коллекции.
- **Удалить версию из репозитория** (Delete version from repository) – удаление из репозитория записи только о выбранной версии коллекции.

4. Подтвердите удаление и нажмите кнопку *Удалить коллекцию* (Delete collections).

Отмена коллекции как устаревшей

Чтобы отменить коллекцию как устаревшую, выполните следующие действия:

1. В окне **Коллекции** (Collections) нажмите на название коллекции, которую необходимо отменить.
2. Нажмите кнопку ⋮ и в открывшемся меню выберите пункт **Отменить коллекцию как устаревшие** (Deprecate collection).

> **Предупреждение**
>
> После отмены коллекции она пропадет из таблицы, поэтому рекомендуется запомнить репозиторий в котором опубликована коллекция. Это понадобится для отмены устаревания коллекции.
>
> Процесс отмены устаревания коллекции описан *ниже*.

3. Подтвердите устаревание коллекции и нажмите кнопку *Отменить коллекцию как устаревшие* (Deprecate collection).

Отмена устаревания коллекции

Чтобы отменить устаревание коллекции, выполните следующие действия:

1. Перейдите в окно **Репозитории** (Repositories).
2. Нажмите на название репозитория в котором опубликована коллекция, устаревание которой необходимо отменить.
3. Перейдите во вкладку **Версии** (Versions).
4. Нажмите на номер последней версии.
5. Перейдите во вкладку **Коллекции** (Collections).
6. Нажмите на название коллекции, устаревание которой необходимо отменить.
7. Нажмите кнопку ⋮ и в открывшемся меню выберите пункт **Отменить устаревание коллекции** (Undeprecate collection).
8. Подтвердите отмену устаревания коллекции и нажмите кнопку *Отменить устаревание коллекций* (Undeprecate collection).

Копирование в другие репозитории

Чтобы скопировать версию коллекции в другие репозитории, выполните следующие действия:

1. В списке коллекций нажмите на название копируемой коллекции.
2. В окне просмотра сведений о коллекции выберите в выпадающем списке необходимую версию.
3. Нажмите кнопку ⋮ и в открывшемся меню выберите пункт **Скопировать версию в репозитории** (Copy version to repositories).

4. В открывшемся окне включите флаги в строках с необходимыми репозиториями и нажмите кнопку *Выбрать* (Select).

Загрузка новой версии

Чтобы загрузить в Private Automation Hub новую версию коллекции, выполните следующие действия:

1. В списке коллекций нажмите на название коллекции, версию которой необходимо загрузить.
2. Нажмите кнопку ⋮ и в открывшемся меню выберите пункт **Загрузить новую версию** (Upload new version).
3. В открывшемся окне заполните форму:
   - **Файл коллекций** (Collection file) – файл в формате `.tar.gz`, сформированный с помощью команды `ansible-galaxy build`.
   - Переключатель доступных для выбора репозиториев:
     - **Тестовые репозитории** (Staging Repos) – для выбора будут доступны только промежуточные репозитории. Загруженная версия коллекции будет ожидать согласования.
     - **Хранилища без конвейера** (Repositories without pipeline) – для выбора будут доступны все репозитории, кроме имеющих метки `pipeline: approved` и `pipeline: rejected`.

     Важно

     Необходимы привилегии, позволяющие загружать версии коллекций в указанный репозиторий.
4. Нажмите кнопку *Загрузить коллекцию* (Upload collection).

Среды исполнения

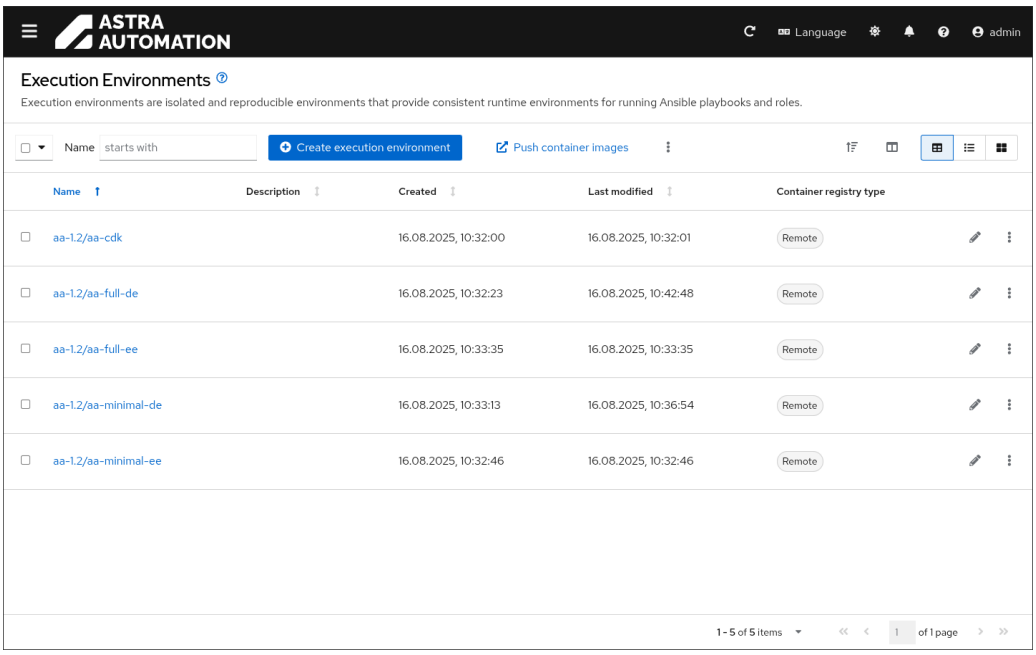
Окно **Среды исполнения** (Execution Environments) предоставляет интерфейс для управления образами сред исполнения и сред принятия решений, применяемых платформой автоматизации, с помощью следующих операций:

- просмотр доступных образов и их версий;
- добавление пользовательских образов;
- контроль соответствия образов корпоративной политике (состав пакетов, зависимости, подпись).

Для перехода к окну **Execution Environments** (Среды исполнения) выберите на панели навигации *Контент автоматизации* ▸ *Среды исполнения* (*Automation Content* ▸ *Execution Environments*).

Таблица сред исполнения

Внешний вид окна **Среды исполнения** (Execution Environments) представлен на схеме:



На панели инструментов размещаются поле поиска, кнопка *Создать среду исполнения* (Create execution environment), ссылка на руководство по загрузке образов в Private Automation Hub и кнопка ⋮ для вызова действий над выделенными записями.

Таблица сред исполнения состоит из следующих столбцов:

- **Название** (Name) – название образа в Private Automation Hub.

  При нажатии на ссылку происходит переход в окно просмотра подробной информации об образе.

- **Описание** (Description) – краткое описание образа.
- **Дата создания** (Created) – дата и время создания образа.
- **Последнее изменение** (Last modified) – дата и время последнего изменения образа.
- **Тип реестра контейнеров** (Container registry type) – тип источника образов:
  - **Внешний репозиторий** (Remote) – образы загружаются в Private Automation Hub из внешнего репозитория;
  - **Локальный** (Local) – образы загружаются в Private Automation Hub напрямую, например, с рабочих мест разработчиков образов.
- Кнопка перехода в окно изменения свойств образа.
- Кнопка ⋮ – вызов меню дополнительных действий.

Просмотр

Для получения подробных сведений об образе нажмите на ссылку с его названием. Окно просмотра сведений об образе состоит из следующих вкладок:

- **Подробности** (Details) – руководство по загрузке образа из Private Automation Hub и инструкция по его использованию (при наличии).
- **Действия** (Activity) – история изменения образа. Здесь выводятся краткие сведения о добавленных и удаленных тегах, загрузке и так далее.
- **Образы** (Images) – таблица версий образа, состоит из следующих столбцов:
  - **Метка** (Tag) – метка, указывающая версию образа, например, `0.6.4` или `latest`.
  - **Опубликовано** (Published) – дата и время публикации образа в реестре Private Automation Hub.
  - **Слои** (Layers) – количество слоев в образе.

  - **Размер** (Size) – размер образа.
  - **Краткая свода** (Digest) – тип и значение хеша образа.
  - Ссылка на загрузку образа из Private Automation Hub.
  - Кнопка ⋮ – вызов меню дополнительных действий.
- **Командный доступ** (Team Access) – таблица команд и назначенных им ролей на доступ к образам.
- **Доступ пользователей** (User Access) – таблица пользователей и назначенных им ролей на доступ к образам.

Создание среды исполнения

Для создания записи о среде исполнения выполните следующие действия:

1. В окне **Среды исполнения** (Execution Environments) нажмите кнопку *Создать среду исполнения* (Create execution environment).
2. Заполните форму **Создать среду исполнения** (Create execution environment):
   - **Название** (Name) – название образа, например, `custom-ee`.

     Важно

     Будьте внимательны при выборе названия – его нельзя будет изменить. Название образа используется при формировании ссылок на загрузку из Private Automation Hub.

     Требования к названию:
     - Допускается использование строчных букв латинского алфавита, цифр, дефиса, подчеркивания, точки и косой черты.
     - Первый и последний символы – не дефис.
   - **Название вышестоящего объекта** (Upstream name) – название раздела во внешнем реестре образов, например, `aa-1.2/aa-full-ee`.
   - **Реестр** (Registry) – внешний реестр образов.

     Значение в этом поле выбирается из справочника *внешних реестров*.
   - **Добавить тег(и) для включения** (Add tag(s) to include) – список тегов, образы с которыми необходимо загрузить из внешнего реестра, например, `0.6.4`, `latest`. Возможна загрузка версий образов с использованием шаблона, например, `0.4.*`.
   - **Добавить тег(и) для исключения** (Add tag(s) to exclude) – список тегов, образы с которыми не следует загружать из внешнего реестра. Возможно исключение версий с использованием шаблона, например, `dev-*`.
3. Нажмите кнопку *Создать среду исполнения* (Create execution environment).
4. Если источником образов служит внешний реестр, *запустите синхронизацию образов*.

Изменение основной информации об образе

Для изменения основной информации об образе выполните следующие действия:

1. В окне **Среды исполнения** (Execution Environments) нажмите на ссылку с названием необходимого образа.
2. Нажмите кнопку *Редактировать среду исполнения* (Edit execution environment).
3. Заполните форму **Редактировать <название_образа>** (Edit <execution_environment>).
4. Нажмите кнопку *Сохранить среду исполнения* (Save execution environment).

5. *Запустите синхронизацию образов с внешним реестром*.

Описание среды исполнения

Чтобы загрузить в Private Automation Hub руководство по использованию образа среды исполнения или среды принятия решений, выполните следующие действия:

1. В таблице образов нажмите на ссылку с названием необходимой записи.
2. Во вкладке **Подробности** (Details) нажмите кнопку *Добавить* (Add).
3. Добавьте в поле **README** инструкцию по использованию образа среды исполнения в формате Markdown.
4. Нажмите кнопку *Сохранить* (Save).

Синхронизация с внешним реестром

Чтобы запустить синхронизацию образов с внешним реестром, выполните следующие действия:

1. В таблице образов нажмите кнопку ⋮ в строке с необходимой записью.
2. В открывшемся меню выберите **Синхронизировать среду исполнения** (Sync execution environment).

Для получения информации о ходе синхронизации перейдите в раздел *Управление задачами* (Task Management).

Удаление образов

Примечание

После удаления образов *измените список тегов* в основных данных о среде исполнения. В противном случае удаленные образы будут загружены заново при первой же синхронизации.

Для удаления образов выполните следующие действия:

1. В окне **Среды исполнения** (Execution Environments) включите флаги напротив удаляемых записей.
2. Нажмите кнопку ⋮ и в открывшемся меню выберите **Удалить среду исполнения** (Delete execution environment).
3. Подтвердите удаление.

Настройка доступа

Доступ к образу может быть предоставлен отдельным пользователям и группам. В обоих случаях назначаются роли из представленного списка:

- `galaxy.execution_environment_collaborator`;
- `galaxy.execution_environment_namespace_owner`;
- `galaxy.execution_evnironment_publisher`.

Описание привилегий, предоставляемых каждой ролью, см. в *справочнике*.

Настройка доступа команд

Чтобы добавить роли для команд, выполните следующие действия:

1. В окне **Среды исполнения** (Execution Environments) нажмите на ссылку с названием необходимой записи.

2. Выберите вкладку **Командный доступ** (Team Access).
3. Нажмите кнопку *Добавить роль* (Add roles).
4. На этапе **Выбрать команды** (Select team(s)) включите флаги напротив названий необходимых команд и нажмите кнопку *Далее* (Next).
5. На этапе **Выбрать роли для применения** (Select roles to apply) включите флаги напротив названий необходимых ролей и нажмите кнопку *Далее* (Next).
6. На этапе **Обзор** (Review) убедитесь в корректности настроек и нажмите кнопку *Завершить* (Finish).

Чтобы отозвать роли у команд, выполните следующие действия:

1. В окне **Среды исполнения** (Execution Environments) нажмите на ссылку с названием необходимой записи.
2. Выберите вкладку **Командный доступ** (Team Access).
3. Включите флаги напротив записей, которые хотите удалить.
4. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите **Удалить роли** (Remove roles).
5. Подтвердите удаление.

Настройка доступа пользователей

Чтобы добавить роли для отдельных пользователей, выполните следующие действия:

1. В окне **Среды исполнения** (Execution Environments) нажмите на ссылку с названием необходимой записи.
2. Выберите вкладку **Доступ пользователей** (User Access).
3. Нажмите кнопку *Добавить роль* (Add roles).
4. На этапе **Выбрать пользователей** (Select users(s)) включите флаги напротив необходимых учетных записей и нажмите кнопку *Далее* (Next).
5. На этапе **Выбрать роли для применения** (Select roles to apply) включите флаги напротив названий необходимых ролей и нажмите кнопку *Далее* (Next).
6. На этапе **Обзор** (Review) убедитесь в корректности настроек и нажмите кнопку *Завершить* (Finish).

Чтобы отозвать роли у отдельных пользователей, выполните следующие действия:

1. В окне **Среды исполнения** (Execution Environments) нажмите на ссылку с названием необходимой записи.
2. Выберите вкладку **Доступ пользователей** (User Access).
3. Включите флаги напротив записей, которые хотите удалить.
4. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите **Удалить роли** (Remove roles).
5. Подтвердите удаление.

Удаление отдельных версий образа

Для удаления версии образа выполните следующие действия:

1. В окне **Среды исполнения** (Execution Environments) нажмите на ссылку с названием необходимого образа.
2. Выберите вкладку **Образы** (Images).
3. В строке с удаляемой записью нажмите кнопку ⋮ и в открывшемся меню выберите **Удалить** (Delete).
4. Подтвердите удаление.

5. Чтобы версия образа не загружалась при синхронизации, *настройте список тегов для исключения*.

Удаление всех версий образа

Предупреждение

При удалении образа удаляются все его версии, загруженные в Private Automation Hub.

Для удаления образов выполните следующие действия:

1. В окне **Среды исполнения** (Execution Environments) включите флаги напротив удаляемых записей.
2. Нажмите кнопку ⋮ и в открывшемся меню выберите **Удалить среды исполнения** (Delete execution environments).
3. Подтвердите удаление.

Ключи подписи

Окно **Ключи подписи** (Signature Keys) позволяет управлять инфраструктурой цифровой подписи контента с помощью следующих операций:

- просмотр и контроль ключей, применяемых для проверки доверенности коллекций и репозиториев;
- обеспечение политики безопасности и целостности доставляемого контента в организации.

Для перехода к окну **Ключи подписи** (Signature Keys) выберите на панели навигации *Контент автоматизации* ▸ *Ключи подписи* (*Automation Content* ▸ *Signature Keys*).

Описание окна

Внешний вид окна **Ключи подписи** (Signature Keys) представлен на схеме:

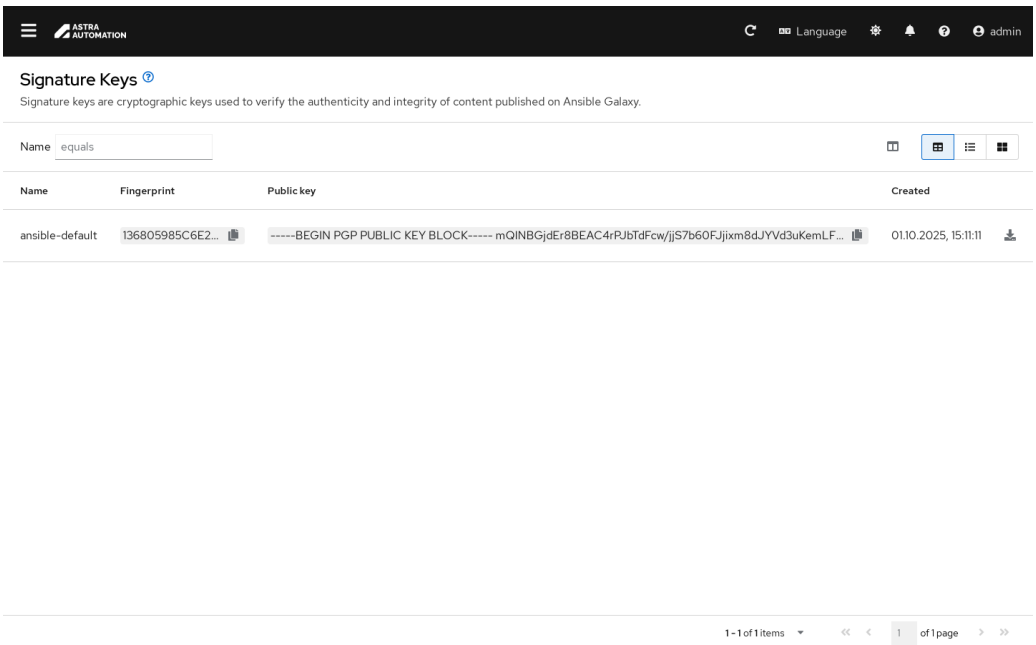


Таблица ключей подписи состоит из следующих столбцов:

- **Название** (Name) – уникальное название ключа, позволяющее быстро идентифицировать конкретный публичный ключ в списке;

- **Подтверждение подлинности** (Fingerprint) – уникальная последовательность символов, представляющая криптографический отпечаток ключа;
- **Открытый ключ** (Public key) – открытый ключ в формате GPG, который используется для проверки цифровой подписи;
- **Дата создания** (Created) – дата и время создания ключа;
- кнопка *Загрузить ключ* (Download key) предназначенная для скачивания на компьютер файла с содержимым публичного ключа в формате GPG.

Создание ключа

UI платформы не поддерживает создание ключей для подписи коллекций, поэтому ключи необходимо сгенерировать вручную с помощью утилиты GPG, как описано в *инструкции по сознанию ключа подписи*.

Репозитории

Окно **Репозитории** (Repositories) позволяет выполнять следующие действия:

- *просматривать сведения о существующих репозиториях*;
- *создавать новые репозитории*;
- *изменять настройки существующих репозиториев*;
- *управлять доступом к репозиториям*;
- *получать настройки для использования репозитория утилитами Ansible*;
- *синхронизировать содержимое репозиториев Private Automation Hub с внешними репозиториями*;
- *управлять версиями репозиториев*;
- *удалять репозитории*.

Для перехода в окно **Репозитории** (Repositories) выберите на панели навигации *Контент автоматизации* ▸ *Репозитории* (*Automation Content* ▸ *Repositories*).

Таблица репозиториев

Внешний вид окна **Репозитории** (Repositories) представлен на схеме:

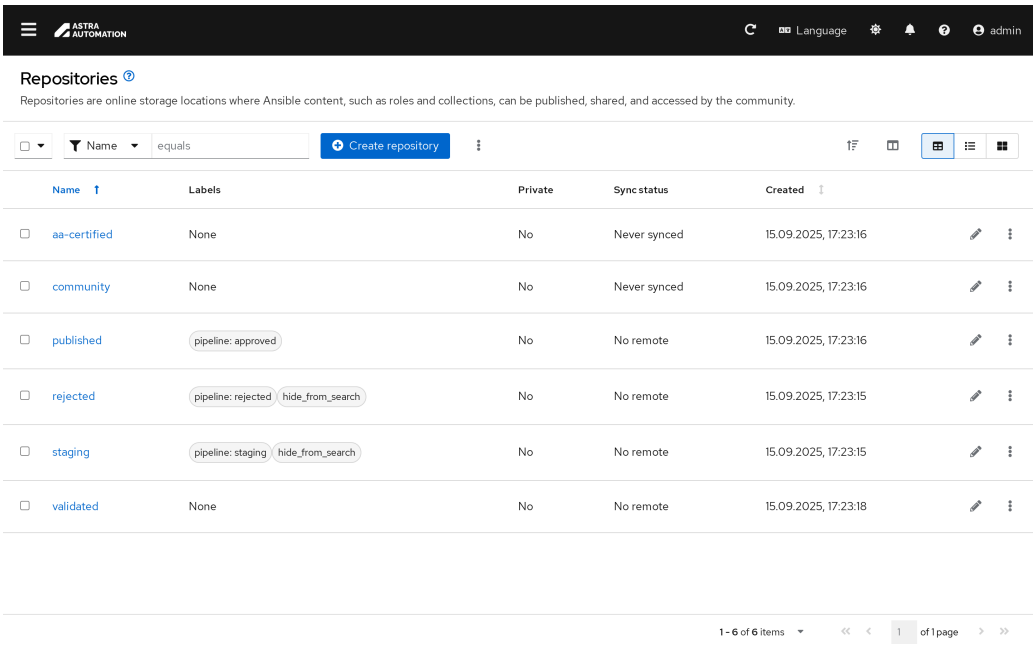


Таблица репозиториев состоит из следующих столбцов:

- **Название** (Name) – название репозитория. При нажатии на него происходит переход в окно просмотра подробной информации о репозитории.
- **Метки** (Labels) – метки, которыми отмечен репозиторий.
- **Частный** (Private) – если репозиторий приватный, доступ к нему имеют только аутентифицированные пользователи.
- **Состояние синхронизации** (Sync status) – количество времени с момента последней синхронизации репозитория с внешним репозиторием.
- **Дата создания** (Created) – дата и время создания репозитория.
- Кнопки для вызова часто выполняемых действий:
  - редактирование репозитория;
  - синхронизация репозитория;
  - копирование настроек CLI;
  - удаление репозитория.

Просмотр

Для получения подробных сведений о репозитории нажмите на его название в таблице репозиториев. Окно просмотра сведений о репозитории состоит из следующих вкладок:

- **Подробности** (Details) – общие сведения о репозитории: название, описание, количество версий, сохраненное в Private Automation Hub, URL репозитория и так далее.
- **Версии коллекций** (Collection versions) – список коллекций с указанием версий, загруженных в Private Automation Hub.
- **Версии** (versions) – версии репозитория, хранящегося в Private Automation Hub.

  Таблица версий состоит из следующих колонок:
  - **Номер версии** (Version number) – номер версии репозитория;
  - **Дата создания** (Created date) – дата и время создания версии;
  - кнопка, позволяющая вернуть состояние репозитория к выбранной версии.
- **Распределения** (Distributions) – список точек распространения содержимого репозитория, а также краткая инструкция по настройке `ansible-galaxy` для их использования.
- **Командный доступ** (Team Access) – таблица команд и назначенных им *ролей*.
- **Доступ пользователей** (User Access) – таблица пользователей и назначенных им *ролей*.

Создание репозитория

Для создания репозитория выполните следующие действия:

1. В окне **Репозитории** (Repositories) нажмите кнопку *Создать репозиторий* (Create repository).
2. Заполните форму **Создать репозиторий** (Create repository):
   - **Название** (Name) – название репозитория в Private Automation Hub.

     Важно

     Будьте внимательны при выборе названия – его нельзя будет изменить.

   - **Описание** (Description) – краткое описание репозитория, например, для каких целей он создается.

- **Сохраняемое количество версий** (Retained number of versions) – количество версий репозитория, которое будет храниться в Private Automation Hub. При создании новой версии самая старая будет удаляться автоматически.
- **Создать дистрибутив** (Create a distribution) – если этот флаг включен, пользователи Private Automation Hub смогут синхронизировать, загружать и искать содержимое, хранящееся в репозитории. В противном случае содержимое репозитория будет недоступно для пользователей.

> **Примечание**
>
> В свойствах защищенных репозиториев `aa-certified`, `validated`, `community` и `published` эта настройка по умолчанию **включена**, и выключить ее нельзя.
>
> В свойствах защищенных репозиториев `rejected` и `staging` эта настройка по умолчанию **выключена**, и включить ее нельзя.

- **Конвейер задач** (Pipeline) – порядок попадания содержимого в репозиторий:
  - **Пусто** (None) – версия коллекции становится доступной пользователям сразу после загрузки в репозиторий.
  - **Согласовано** (Approved) – версия коллекции попадает в репозиторий после согласования. Загрузка версий коллекций напрямую в этот репозиторий невозможна.
  - **Промежуточный** (Staging) – хранящиеся в репозитории версии коллекций недоступны пользователям и ожидают согласования. Загружать версии коллекций в этот репозиторий могут любые пользователи, имеющие привилегии на загрузку содержимого в пространства имен.
- **Метки** (Labels) – метки репозитория.
- **Скрыть из поиска** (Hide from search) – если этот флаг включен, репозиторию присваивается метка `hide_from_search`. Эта метка предотвращает отображение коллекций из этого репозитория в результатах поиска.
- **Сделать приватным** (Make private) – если этот флаг включен, содержимое репозитория смогут использовать только его пользователи-владельцы.
- **Внешний** (Remote) – внешний репозиторий, из которого при синхронизации загружается содержимое.

3. Нажмите кнопку *Создать репозиторий* (Create repository).

Изменение репозитория

Для изменения свойств существующего репозитория выполните следующие действия:

1. В окне **Репозитории** (Repositories) нажмите на название репозитория, свойства которого необходимо изменить.
2. В окне свойств выбранного репозитория нажмите кнопку *Редактировать репозиторий* (Edit repository).
3. Измените свойства репозитория.

> **Важно**
>
> Название существующего репозитория нельзя изменить. У защищенных репозиториев также недоступны для изменения некоторые другие свойства.

- **Описание** (Description) – краткое описание репозитория, например, для каких целей он создается.

- **Сохраняемое количество версий** (Retained number of versions) – количество версий репозитория, которое будет храниться в Private Automation Hub. При создании новой версии самая старая будет удаляться автоматически.
- **Создать дистрибутив** (Create a distribution) – если этот флаг включен, пользователи Private Automation Hub смогут синхронизировать, загружать и искать содержимое, хранящееся в репозитории. В противном случае содержимое репозитория будет недоступно для пользователей.

  Примечание

  В свойствах защищенных репозиториев `aa-certified`, `validated`, `community` и `published` эта настройка по умолчанию **включена**, и выключить ее нельзя.

  В свойствах защищенных репозиториев `rejected` и `staging` эта настройка по умолчанию **выключена**, и включить ее нельзя.

- **Конвейер задач** (Pipeline) – порядок попадания содержимого в репозиторий:
  - **Пусто** (None) – версия коллекции становится доступной пользователям сразу после загрузки в репозиторий.
  - **Согласовано** (Approved) – версия коллекции попадает в репозиторий после согласования. Загрузка версий коллекций напрямую в этот репозиторий невозможна.
  - **Промежуточный** (Staging) – хранящиеся в репозитории версии коллекций недоступны пользователям и ожидают согласования. Загружать версии коллекций в этот репозиторий могут любые пользователи, имеющие привилегии на загрузку содержимого в пространства имен.
- **Метки** (Labels) – метки репозитория.
- **Скрыть из поиска** (Hide from search) – если этот флаг включен, репозиторию присваивается метка `hide_from_search`. Эта метка предотвращает отображение коллекций из этого репозитория в результатах поиска.
- **Сделать приватным** (Make private) – если этот флаг включен, содержимое репозитория смогут использовать только его пользователи-владельцы.
- **Внешний** (Remote) – внешний репозиторий, из которого при синхронизации загружается содержимое.

4. Нажмите кнопку *Сохранить репозиторий* (Save repository).

Настройка владения

Владельцами репозитория могут быть пользователи и команды. Порядок настройки одинаков.

1. В окне **Репозитории** (Repositories) нажмите на название репозитория, доступ к которому необходимо настроить.
2. Выберите вкладку **Командный доступ** (Team Access) или **Доступ пользователей** (User Access).
3. Нажмите кнопку *Добавить роль* (Add roles).
4. В открывшемся окне выберите нужного пользователя или команду и нажмите кнопку *Далее* (Next).
5. Во вкладке **Выбрать роли для применения** (Select roles to apply) включите флаг роли `galaxy.ansible_repository_owner` и нажмите кнопку **Далее** (Next).
6. Во вкладке **Обзор** (Review) проверьте корректность настроек и нажмите кнопку *Завершить* (Finish).

Настройка CLI на работу с репозиторием

Чтобы настроить инструменты командной строки Ansible на работу с репозиторием, выполните следующие действия:

1. В окне **Репозитории** (Repositories) нажмите на название репозитория, который необходимо настроить.
2. Нажмите кнопку ⋮ и в открывшемся меню выберите пункт *Скопировать конфигурацию CLI* (Copy CLI configuration). Необходимые настройки будут автоматически скопированы в буфер обмена.
3. Добавьте строки из буфера обмена в конфигурационный файл `ansible.cfg`.

Синхронизация с внешним репозиторием

Чтобы синхронизировать содержимое репозитория Private Automation Hub с внешним репозиторием, выполните следующие действия:

1. В окне **Репозитории** (Repositories) нажмите на название репозитория, содержимое которого необходимо синхронизировать.
2. Нажмите кнопку ⋮ и в открывшемся меню выберите пункт *Синхронизировать репозиторий* (Sync repository).
3. В открывшемся окне укажите настройки синхронизации:
   - **Зеркало** (Mirror) – если этот переключатель включен, Private Automation Hub выполняет с версиями коллекций следующие действия:
     1. Удаляет версии, отсутствующие в источнике.
     2. Загружает из внешнего источника версии, которых нет в репозитории.
     3. Отмечает как устаревшие версии, отмеченные как устаревшие в источнике.
     4. Отмечает как актуальные версии, отмеченные как актуальные в источнике.
   - **Оптимизировать** (Optimize) – если этот переключатель включен, загрузка содержимого из внешнего репозитория будет выполняться только в том случае, если внешний репозиторий подтвердит наличие изменений.
4. Нажмите кнопку *Синхронизация* (Sync).

Управление версиями репозитория

Если в настройках репозитория количество сохраненных версий больше единицы, при необходимости можно вернуть репозиторий в одно из прежних состояний:

1. В окне **Репозитории** (Repositories) нажмите на название репозитория, который необходимо вернуть в одно из прежних состояний.
2. Выберите вкладку **Версии** (Version).
3. В строке версии, к которой необходимо вернуть репозиторий, нажмите кнопку ⋮ и в открывшемся меню выберите **Вернуть к этой версии** (Revert to this version).

Удаление репозитория

Предупреждение

При удалении репозитория также удаляются все связанные с ним ресурсы.

Для удаления репозиториев выполните следующие действия:

1. В таблице установите флаги в строках с репозиториями, которые необходимо удалить.

2. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите пункт *Удалить репозитории* (Delete repositories).
3. Подтвердите удаление.

Внешние реестры

Окно **Внешние реестры** (Remote Registries) позволяет выполнять следующие действия:

- *добавлять записи о внешних реестрах контейнеров*;
- *синхронизировать списки образов с внешними реестрами контейнеров*;
- *удалять записи о внешних реестрах контейнеров*.

Для перехода в окно **Внешние реестры** (Remote Registries) выберите на панели навигации *Контент автоматизации* ▸ *Внешние реестры* (*Automation Content* ▸ *Remote Registries*).

Таблица внешних реестров

Внешний вид окна **Внешние реестры** (Remote Registries) представлен на схеме:

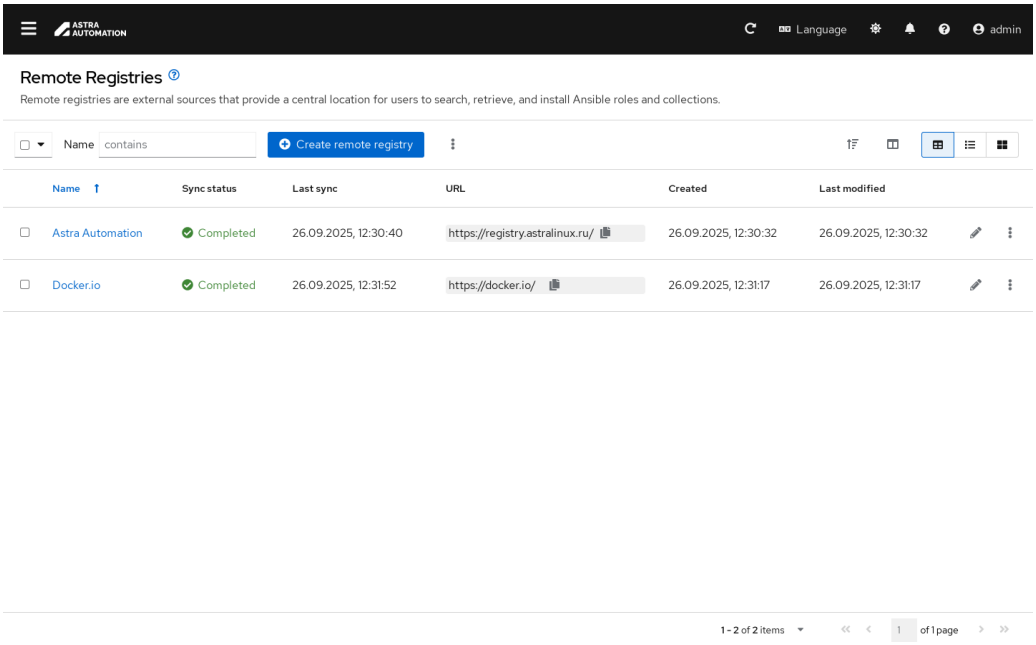


Таблица внешних реестров состоит из столбцов согласно настройке с помощью кнопки *Настроить столбцы* (Manage columns). Полный список столбцов:

- флаги для выбора нескольких записей;
- **Название** (Name) – название внешнего реестра;
- **Состояние синхронизации** (Sync status) – краткие сведения об успешности последней синхронизации Private Automation Hub с внешним реестром;
- **Последняя синхронизация** (Last sync) – дата и время последней синхронизации;
- **URL** – ссылка на внешний реестр;
- **Дата создания** (Created) – дата и время создания записи о внешнем реестре;
- **Последнее изменение** (Last modified) – дата и время последнего изменения записи о внешнем реестре;
- кнопки вызова часто выполняемых действий:
  - редактирование свойств внешнего реестра;
  - синхронизация Private Automation Hub с внешним реестром;
  - индексирование среды исполнения;

– удаление внешнего реестра.

Добавление внешнего реестра

Для добавления записи о внешнем реестре выполните следующие действия:

1. В окне **Внешние реестры** (Remote Registries) нажмите кнопку *Создать внешний реестр* (Create remote registry).
2. В открывшемся окне **Создать внешний реестр** (Create remote registry) заполните форму:
   - **Название** (Name) – произвольное название внешнего реестра. Должно быть уникальным на уровне Private Automation Hub.

     Важно

     Будьте внимательны при выборе названия внешнего реестра – его нельзя будет изменить.

   - **URL** – ссылка на внешний реестр. Должна начинаться с указания используемого протокола – `http://` или `https://`, например:

     ```
     https://hub.astra-automation.ru/
     ```

   - **Название учетной записи** (Username) и **Пароль** (Password) – учетные данные пользователя, которые следует использовать для авторизации при доступе к внешнему реестру.

   При нажатии на ссылку *Показать дополнительные параметры* (Show advanced options) в форме появляются дополнительные поля.

   - **URL прокси-сервера** (Proxy URL) – IP-адрес или FQDN используемого прокси-сервера.
   - **Название учетной записи в прокси-сервере** (Proxy username) и **Пароль прокси-сервера** (Proxy password) – учетные данные пользователя, которые следует использовать для авторизации на прокси-сервере.
   - **Проверка TLS** (TLS validation) – если флаг включен, выполняется одноранговая проверка сертификата TLS, используемого для защиты подключения к внешнему реестру.
   - **Ключ клиента** (Client key) – файл с приватным ключом клиента.
   - **Сертификат клиента** (Client certificate) – файл сертификата клиента.
   - **Сертификат CA** (CA certificate) – файл сертификата CA.
   - **Одновременность загрузки** (Download concurrency) – максимальное количество параллельных загрузок, которые Private Automation Hub может запустить при работе с внешним реестром.
   - **Лимит ставки** (Rate Limit) – ограничение на количество запросов в секунду при работе Private Automation Hub с внешним реестром.
3. Нажмите кнопку *Создать внешний реестр* (Create remote registry).

Синхронизация с внешним реестром образов

Чтобы синхронизировать хранящиеся в Private Automation Hub образы среды исполнения, связанные с *внешним реестром образов*, выполните следующие действия:

1. В таблице записей о внешних реестрах нажмите кнопку *Синхронизировать внешний реестр* (Sync remote registry).
2. Дождитесь перехода записи о внешнем реестре в статус **Завершено** (Completed).

Удаление внешнего реестра

Примечание

При удалении записи о внешнем реестре загруженные из него образы среды исполнения остаются в Private Automation Hub. Инструкции по удалению образов приведены *здесь*.

Для удаления записей о внешнем реестре выполните следующие действия:

1. В таблице установите флаги в строках с внешними реестрами, которые необходимо удалить.
2. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите пункт *Удалить внешние реестры* (Delete remote registries).
3. Подтвердите удаление.

Управление задачами

Окно **Управление задачами** (Task management) предоставляет средства мониторинга внутренних фоновых процессов Private Automation Hub:

- *отслеживание состояния и истории системных операций* (синхронизации, публикации, индексации);
- *остановка активных задач* при возникновении ошибок или необходимости отмены операции;
- анализ результатов выполнения задач для диагностики и устранения ошибок.

Для перехода в окно **Управление задачами** (Task management) выберите на панели навигации *Контент автоматизации* ▸ *Управление задачами* (*Automation Content* ▸ *Task Management*).

Таблица записей о задачах

Внешний вид окна **Управление задачами** (Task management) представлен на схеме:

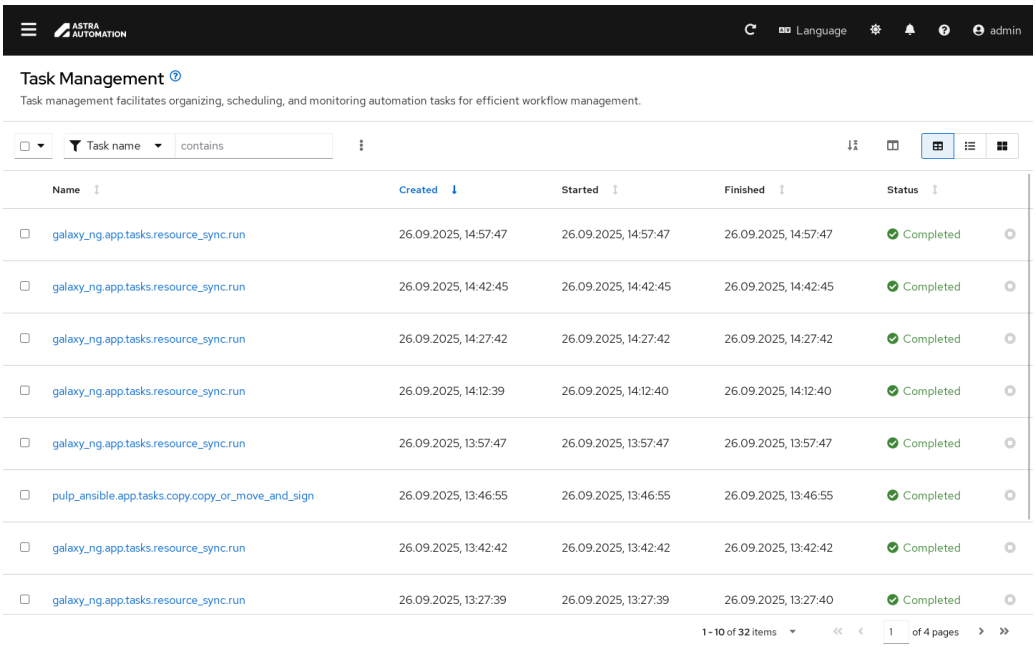


Таблица записей о задачах состоит из следующих колонок:

- Флаги для выбора нескольких записей.
- **Название** (Name) – ссылка на окно просмотра подробных сведений о задаче.

- **Дата создания** (Created) – дата и время создания задачи.
- **Начато** (Started) – дата и время запуска задачи.
- **Закончено** (Finished) – дата и время завершения выполнения задачи.
- **Статус** (Status) – текущий статус задачи.
- Кнопка *Остановить задачу* (Stop task).

  Можно отменить только работающие или ожидающие выполнения задачи.

Просмотр

Для получения подробных сведений о задаче нажмите на ее название в таблице. Окно просмотра сведений о задаче состоит из следующих панелей:

- **Подробности задачи** (Task detail) – название, статус, а также дата и время создания, запуска и завершения задачи.
- **Группы задач** (Task groups) – группы, к которым относится задача, ссылка на родительскую задачу и дочерние задачи.
- **Зарезервировать ресурсы** (Reserve resources) – типы связанных ресурсов, используемое расширение и название ресурса.
- **Сообщения о ходе исполнения** (Progress messages) – сведения о ходе выполнения задачи.

Остановка

Чтобы остановить выполнение задачи, выполните следующие действия:

1. В таблице задач нажмите кнопку *Остановить задачу* (Stop task) в строке с задачей, которую необходимо остановить.
2. Подтвердите остановку задачи.

Согласования коллекций

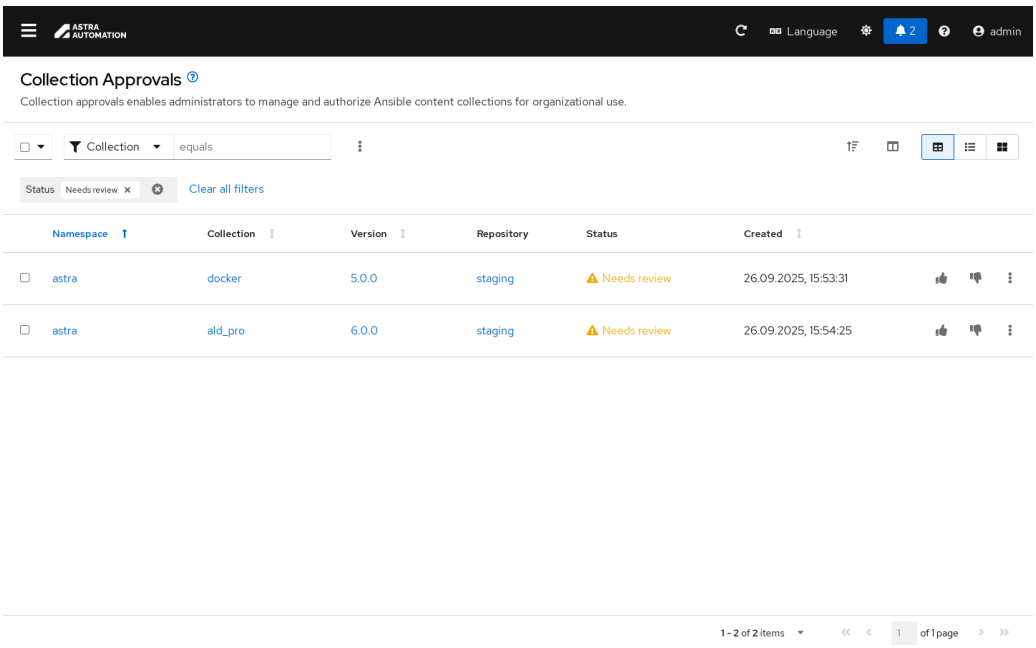
Окно **Согласование коллекций** (Collection Approvals) используется для согласования публикации коллекции в корпоративном каталоге контента (Private Automation Hub) с помощью следующих операций:

- отслеживание новых версий коллекций, ожидающих утверждения;
- проведение проверок и валидации содержимого коллекций перед публикацией;
- утверждение или отклонение выпуска, обеспечивая контроль качества корпоративного контента.

Для перехода в окно **Согласование коллекций** (Collection Approvals) выберите на панели навигации *Контент автоматизации* ▸ *Согласования коллекций* (*Automation Content* ▸ *Collection Approvals*).

Описание окна

Внешний вид окна **Согласования коллекций** (Collection Approvals) представлен на схеме:



Окно **Согласования коллекций** (Collection Approvals) состоит из панели фильтрации и таблицы версий коллекций, ожидающих согласования.

Таблица версий коллекций состоит из следующих столбцов:

- Флаги для выбора нескольких записей.
- **Пространство имен** (Namespace) – название *пространства имен*, к которому относится коллекция.
- **Коллекция** (Collection) – название коллекции.
- **Версия** (Version) – ссылка с номером версии коллекции. В этой же колонке находится ссылка на загрузку архива с версией коллекции.
- **Репозиторий** (Repository) – название *репозитория*, в котором сейчас находится версия коллекции.
- **Статус** (Status) – текущий статус версии коллекции.
- **Дата создания** (Date created) – дата и время создания репозитория.
- Кнопки вызова часто выполняемых действий:
  - согласование версии коллекции;
  - отклонение версии коллекции;
  - перехода в журнал импорта.

Просмотр

Перед согласованием коллекции рекомендуется просмотреть сведения о ней. Эту задачу можно выполнить несколькими способами.

Загрузка кода коллекции на локальный компьютер

Чтобы изучить исходный код коллекции, загрузите его на локальный компьютер.

1. Нажмите на ссылку в колонке **Версия** (Version).
2. Перейдите во вкладку **Установить** (Install).
3. Нажмите на кнопку *Загрузить архив tarball* (Download tarball), чтобы сохранить архив на локальный компьютер.
4. Создайте каталог для распаковки архива, например:

```
mkdir astra-nginx-1.8.0/
```

5. Распакуйте содержимое архива в созданный каталог:

```
tar xf astra-nginx-1.8.0.tar.gz -C ./astra-nginx-1.8.0/
```

Просмотр основных сведений о версии коллекции

Чтобы просмотреть общую информацию о версии коллекции до ее согласования, нажмите на ссылку в колонке **Версия** (Version).

Окно просмотра сведений о коллекции состоит из выпадающего списка, позволяющего выбрать версию, и следующих вкладок:

- **Подробности** (Details) – подробные сведения о коллекции.
- **Установить** (Install) – инструкция по установке коллекции с использованием утилиты `ansible-galaxy`.
- **Документация** (Documentation) – документация коллекции, входящих в нее ролей и модулей.
- **Содержимое** (Contents) – краткая информация о модулях и ролях, предоставляемых коллекцией.
- **Журнал импорта** (Import log) – журнал импорта коллекции в Private Automation Hub.
- **Зависимости** (Dependencies) – список зависимостей коллекции.
- **Распределения** (Distributions) – список репозиториев, в которых размещена коллекция, а также краткая инструкция по настройке `ansible-galaxy` для их использования.

Согласование и отклонение версий коллекции

Чтобы согласовать версию коллекции, нажмите кнопку *Согласовать и подписать коллекцию* (Approve and sign collection) в соответствующей строке и подтвердите согласование. Версия коллекции будет перемещена в соответствующий репозиторий. Если в настройках репозитория включено создание дистрибутива, коллекция будет доступна для поиска и загрузки пользователями.

Чтобы отклонить версию коллекции, в соответствующей строке нажмите кнопку *Отклонить коллекцию* (Reject collection) и подтвердите отклонение коллекции.

Важно

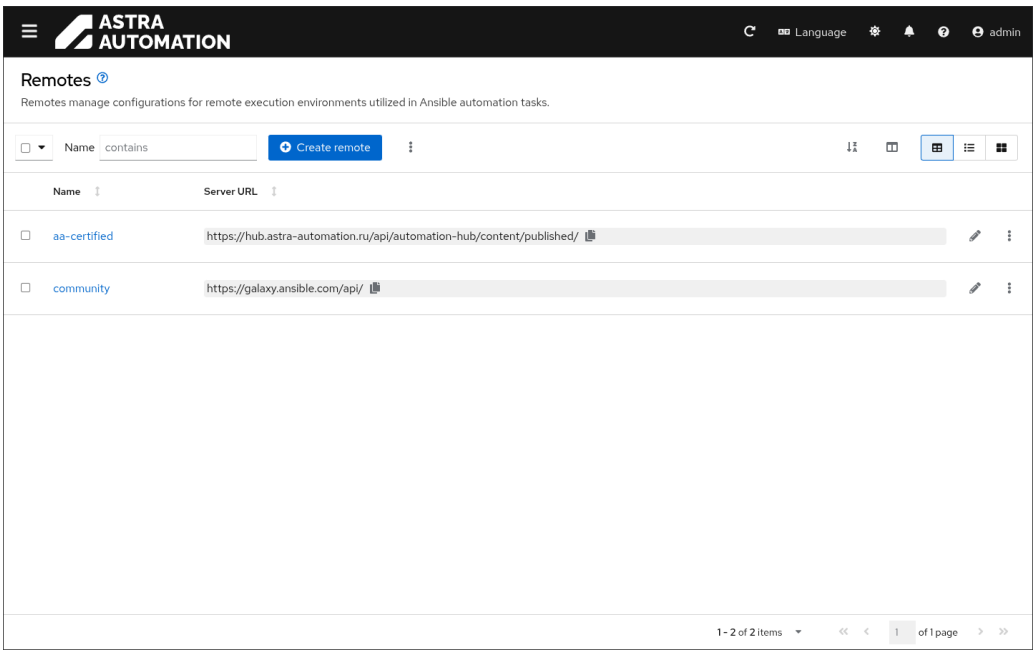
При отклонении версия коллекции не удаляется из Private Automation Hub, а перемещается в защищенный репозиторий `rejected`. Для окончательного удаления версии коллекции *удалите* ее из этого репозитория.

Внешние репозитории

Для перехода к списку записей о внешних репозиториях выберите на панели навигации *Automation Content* ▸ *Внешние репозитории* (*Automation Content* ▸ *Remotes*).

Описание окна

Внешний вид окна **Внешние репозитории** (Remotes) представлен на схеме:



На панели инструментов размещаются поле поиска, кнопка *Создать внешний репозиторий* (Create remote) и кнопка ⋮ для вызова действий над выделенными записями.

Таблица внешних репозиториев состоит из следующих колонок:

- Флаги для выбора нескольких записей.
- **Название** (Name) – ссылка с названием внешнего репозитория в Private Automation Hub.

  При нажатии на ссылку происходит переход в окно просмотра подробной информации о внешнем репозитории.
- **Server URL** – ссылка на внешний репозиторий.
- **Дата создания** (Created) – дата и время создания записи.
- Кнопка для перехода в окно изменения данных о внешнем репозитории.
- Кнопка ⋮ для вызова меню дополнительных действий над отдельной записью.

Просмотр

Для получения подробных сведений о внешнем репозитории нажмите на ссылку с его названием. Окно просмотра сведений о внешнем репозитории состоит из следующих вкладок:

- **Подробности** (Details) – общие сведения о репозитории: название, URL, настройки защиты подключения, список загружаемых коллекций и так далее.
- **Командный доступ** (Team Access) – таблица команд и назначенных им ролей на доступ к записи о внешнем репозитории.
- **Доступ пользователей** (User Access) – таблица пользователей и назначенных им ролей на доступ к записи о внешнем репозитории.

Добавление внешнего репозитория

Для добавления записи о внешнем репозитории выполните следующие действия:

1. В окне **Внешние репозитории** (Remotes) нажмите кнопку *Создать внешний репозиторий* (Create remote).
2. Заполните форму **Создать внешний репозиторий** (Create remote):
   - **Название** (Name) – название внешнего репозитория в Private Automation Hub.

Важно

Будьте внимательны при выборе названия. После создания записи его нельзя будет изменить.

- **URL** – ссылка на внешний репозиторий.
- **Синхронизировать все зависимости** (Sync all dependencies) – если этот флаг включен, при синхронизации в Private Automation Hub из внешнего репозитория загружаются коллекции и все их зависимости. Если переключатель выключен, загружаются только коллекции, без зависимостей.
- **Название учетной записи** (Username) – название учетной записи, используемой для доступа к внешнему репозиторию.
- **Пароль** (Password) – пароль учетной записи, используемой для доступа к внешнему репозиторию.
- **Токен** (Token) – токен для доступа к внешнему репозиторию.
- **URL единого входа** (SSO URL) – URL страницы единого входа, используемой для доступа к внешнему репозиторию.
- **Файл требований** (Requirements file) – список коллекций, которые Private Automation Hub должен загрузить из внешнего репозитория, например:

```
---
collections:
  - name: astra.ald_pro
    version: ">=5.0.0"
  - name: astra.brest
    version: "4.0.3"
  - name: astra.ceph
  - name: astra.nginx
```

Если поле не заполнено, из внешнего репозитория загружаются все имеющиеся в нем коллекции. В противном случае загружаются **только** указанные коллекции.

При нажатии на ссылку *Показать дополнительные параметры* (Show advanced options) в форме появляются дополнительные поля.

- **URL прокси-сервера** (Proxy URL) – IP-адрес или FQDN используемого прокси-сервера.
- **Название учетной записи в прокси-сервере** (Proxy username) и **Пароль прокси-сервера** (Proxy password) – учетные данные пользователя, которые следует использовать для авторизации на прокси-сервере.
- **Проверка TLS** (TLS validation) – если флаг включен, выполняется одноранговая проверка TSL-сертификата, используемого для защиты подключения к внешнему репозиторию.
- **Ключ клиента** (Client key) – файл с приватным ключом клиента.
- **Сертификат клиента** (Client certificate) – файл сертификата клиента.
- **Сертификат CA** (CA certificate) – файл сертификата CA.
- **Параллельность загрузки** (Download concurrency) – максимальное количество параллельных загрузок, которые Private Automation Hub может запустить при работе с внешним репозиторием.
- **Ограничение скорости** (Rate limit) – ограничение на количество запросов в секунду при работе Private Automation Hub с внешним репозиторием.

3. Нажмите кнопку *Создать внешний репозиторий* (Create remote).

Редактирование основных сведений о внешнем репозитории

Для изменения основной информации о внешнем репозитории выполните следующие действия:

1. В окне **Внешние репозитории** (Remotes) нажмите на ссылку с названием нужного внешнего репозитория.
2. Нажмите кнопку *Редактировать внешний репозиторий* (Edit remote).
3. Измените сведения о внешнем репозитории.
4. Нажмите кнопку *Save remote*.

Настройка доступа

Доступ к записям о внешних репозиториях можно настроить как для команд, так и для отдельных пользователей.

Настройка доступа команд

Чтобы добавить команды пользователей в список владельцев записи о внешнем репозитории, выполните следующие действия:

1. В окне **Внешние репозитории** (Remotes) нажмите на ссылку с названием нужного внешнего репозитория.
2. Выберите вкладку **Командный доступ** (Team Access).
3. Нажмите кнопку *Добавить роль* (Add roles).
4. На этапе **Выбрать команды** (Select team(s)) включите флаги напротив названий команд, которым хотите предоставить доступ к записи о внешнем репозитории, и нажмите кнопку *Далее* (Next).
5. На этапе **Выбрать роли для применения** (Select roles to apply) включите флаги напротив названий ролей, которые хотите назначить выбранным командам, и нажмите кнопку *Далее* (Next).
6. На этапе **Обзор** (Review) убедитесь в корректности настроек и нажмите кнопку *Завершить* (Finish).

Чтобы отозвать роли у команд, выполните следующие действия:

1. В окне **Внешние репозитории** (Remotes) нажмите на ссылку с названием нужного внешнего репозитория.
2. Выберите вкладку **Командный доступ** (Team Access).
3. Включите флаги напротив записей, которые хотите удалить.
4. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите **Удалить роли** (Remove roles).
5. Подтвердите удаление.

Настройка доступа пользователей

Чтобы добавить отдельных пользователей в список владельцев записи о внешнем репозитории, выполните следующие действия:

1. В окне **Внешние репозитории** (Remotes) нажмите на ссылку с названием нужного внешнего репозитория.
2. Выберите вкладку **Доступ пользователей** (User Access).
3. Нажмите кнопку *Добавить роль* (Add roles).
4. На этапе **Выбрать пользователей** (Select users(s)) включите флаги напротив учетных записей пользователей, которым хотите предоставить доступ к записи о внешнем репозитории, и нажмите кнопку *Далее* (Next).

5. На этапе **Выбрать роли для применения** (Select roles to apply) включите флаги напротив названий ролей, которые хотите назначить выбранным пользователям, и нажмите кнопку *Далее* (Next).
6. На этапе **Обзор** (Review) убедитесь в корректности настроек и нажмите кнопку *Завершить* (Finish).

Чтобы отозвать роли у отдельных пользователей, выполните следующие действия:

1. В окне **Внешние репозитории** (Remotes) нажмите на ссылку с названием нужного внешнего репозитория.
2. Выберите вкладку **Доступ пользователей** (User Access).
3. Включите флаги напротив записей, которые хотите удалить.
4. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите **Удалить роли** (Remove roles).
5. Подтвердите удаление.

Удаление внешнего репозитория

Примечание

При удалении записи о внешнем репозитории загруженные из него коллекции остаются в Private Automation Hub. Для удаления коллекции следуйте *соответствующей инструкции*.

Для удаления записи о внешнем репозитории выполните следующие действия:

1. В окне **Внешние репозитории** (Remotes) включите флаги напротив удаляемых записей.
2. Нажмите кнопку ⋮ и в открывшемся меню выберите **Удалить внешний репозиторий** (Delete remotes).
3. Подтвердите удаление.

Подключение к Private Automation Hub

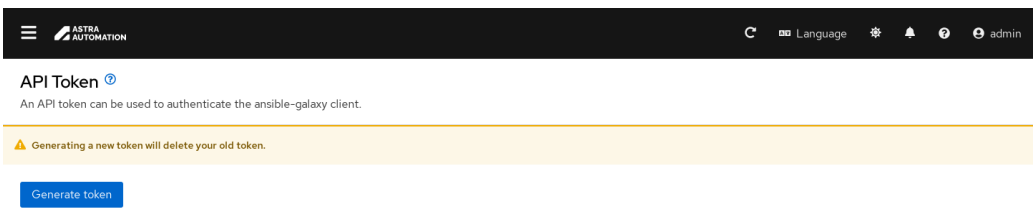
Окно **Подключение к Private Automation Hub** (API Token) позволяет создавать и управлять персональными токенами доступа к Private Automation Hub с помощью следующих операций:

- генерация токена аутентификации для инструментов CLI и автоматизированных сервисов;
- безопасное управление сроками действия и отзывом токенов.

Для перехода в окно **Подключение к Private Automation Hub** (API token) выберите на панели навигации *Контент автоматизации* ▸ *Подключение к Private Automation Hub* (*Automation Content* ▸ *API Token*).

Внешний вид окна

Внешний вид окна **Подключение к Private Automation Hub** (API token) представлен на схеме:



Важно

Период действия токена по умолчанию – 24 часа. По истечении указанного времени его нужно будет создать заново.

При создании нового токена предыдущий перестает действовать немедленно.

Создание токена

Для создания токена выполните следующие действия:

1. Нажмите кнопку *Сгенерировать токен* (Generate token).
2. Скопируйте токен в буфер обмена и сохраните в надежном месте – это единственная возможность его увидеть. В случае утери токена его придется создать заново.
3. Настройте окружение на использование созданного токена согласно *инструкции*.

10.5 API

API для Private Automation Hub представляет собой RESTful API, предоставляющий программный доступ ко всем функциям Private Automation Hub — централизованного репозитория для управления и распространения контента автоматизации в Astra Automation. API позволяет организациям через приложения управлять коллекциями Ansible. Он обеспечивает авторизацию на основе токенов API. Для Private Automation Hub используется API token management, который генерирует постоянные токены без срока действия, обеспечивая безопасный доступ для инструментов командной строки, например `ansible-galaxy`, и внешних систем.

Доступ к API осуществляется через множество точек доступа (endpoints), начиная с базового URI `/api/galaxy/`, и включает полную поддержку методов GET, POST, PUT, PATCH и DELETE для управления ресурсами, такими как collections, namespaces, repositories в зависимости от пользовательских привилегий.

Примечание

Подробное формальное описание HTTP API представлено в *спецификации*.

10.5.1 Спецификация API

ГЛАВА 11

Автоматизация процессов

Основной функцией платформы Astra Automation является исполнение задач автоматизации на основе сценариев автоматизации, содержащихся в заданных проектах.

Подготовка окружения, управление задачами и их исполнение решаются с помощью следующих встроенных механизмов:

- Управление через графический интерфейс пользователя, CLI, API и Ansible.
- Наблюдение за различными аспектами процессов выполнения сценариев автоматизации с использованием средств фильтрации.
- Упрощение запуска процессов с минимальным количеством действий.
- Развитая система безопасности, базирующаяся на ролевой модели и аудите.
- Масштабирование системы управления.
- Обеспечение надежности с помощью кластерных решений и встроенной системы резервного копирования и восстановления.
- Расширение возможностей за счет интеграции с различными репозиториями коллекций Ansible.
- Статическая и динамическая инвентаризация.
- Отправка уведомлений по электронной почте и в различные средства обработки, анализа и распространения сообщений (Grafana, IRC, Mattermost, Slack, Twilio и другие).
- Автоматическое распределение заданий по группам управляемых узлов для снижения пиковой нагрузки.
- И многое другое.

11.1 Общие сведения

Архитектура *Структура управления* Элементы управления и связь между ними.

Графический интерфейс Управление через веб-интерфейс.

Программный интерфейс Формальное описание API в стандарте Open API.

11.1.1 Особенности архитектуры

Архитектура Automation Controller

Общие сведения

Automation Controller — это центральный компонент платформы управления автоматизацией, предоставляющий управляющую плоскость (control plane) через веб-интерфейс, REST API, механизмы управления доступом на основе ролей (RBAC), а также средства интеграции с CI/CD. Он обеспечивает организацию, запуск, масштабирование и делегирование процессов автоматизации на всей инфраструктуре предприятия. Распределённая и модульная архитектура Automation Controller позволяет реализовать автоматизацию различной сложности для большого числа хостов и задач в надёжной и управляемой среде.

Структурные особенности архитектуры

- Кластеризация: Automation Controller может быть развёрнут как в кластере из нескольких узлов, так и на одном сервере для обеспечения отказоустойчивости и масштабируемости.
- Разделение функций: Использование различных типов узлов (control, execution, hop) позволяет оптимизировать производительность для сценариев распределённой автоматизации.
- Централизованное управление: Веб-интерфейс и API предоставляют полный обзор, настройку и контроль всех процессов автоматизации, с детальной видимостью и аудитом.

Основные службы Automation Controller

Каждый узел Automation Controller включает следующий набор служб:

- HTTP служба (*HTTP service*) обеспечивает доступ пользователей и внешних систем к управлению и мониторингу через веб-интерфейс и REST API.
- Служба обратных вызовов (*Callback receiver*) агрегирует статусы и результаты выполнения задач в реальном времени для внутреннего использования и последующего анализа.
- Диспетчер (*Dispatcher*) управляет системой очередей, распределяя задания между исполнительными узлами и контролируя их выполнение.
- Redis выполняет функции высокопроизводительного механизма обмена сообщениями между разными компонентами и оперативного хранения временных данных.
- Syslog NG подготавливает и отправляет журналы событий и операций во внешние центры для мониторинга, аудита и поиска инцидентов.

Примечание

При сбое любой из данных служб вся система автоматически перезапускает все сервисы для восстановления работоспособности; в случае повторных сбоев экземпляр переводится в offline-режим для предотвращения непредсказуемого поведения.

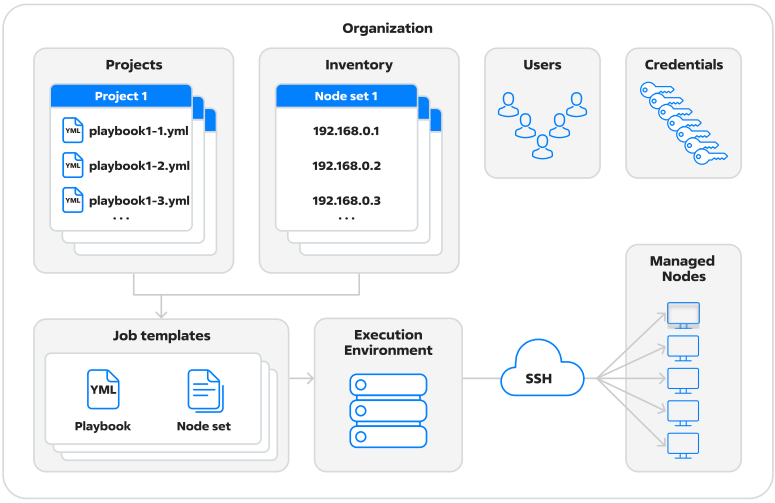
11.1.2 Структура управления

Для управления инфраструктурой контроллер использует следующие компоненты:

- наборы сценариев (playbooks);
- инвентарь (inventory);
- проекты (projects);
- шаблоны (templates);

- среды исполнения (execution environments);
- полномочия (credentials).

Взаимосвязи между компонентами показаны на схеме:



Проект

Проект состоит из одного или нескольких наборов сценариев Ansible. Контроллер позволяет импортировать существующие наборы сценариев из различных источников:

- репозиторий Git;
- репозиторий Subversion;
- архивы форматов `.zip` и `.tar.gz`.

Также существующие наборы сценариев можно вручную разместить в одном из каталогов локальной файловой системы контроллера, после чего указать путь к нужному каталогу в настройках проекта.

Подробности см. в документе *Проекты*.

Наборы сценариев

Для управления узлами контроллер использует *наборы сценариев Ansible*.

Инвентарь

Контроллер предоставляет следующие возможности для управления *инвентарем Ansible*:

- заполнение списка управляемых узлов вручную или путем импорта из стороннего источника;
- использование статического и динамического инвентаря.

Подробности см. в разделе *Инвентарь*.

Шаблон

Шаблоны бывают двух типов:

- шаблоны заданий;
- шаблоны потоков заданий.

Шаблон задания связывает между собой набор управляемых узлов из инвентаря и один из наборов сценариев, доступных в проекте.

Шаблон потока заданий связывает между собой шаблоны заданий, а также предоставляет дополнительные функции, позволяющие настроить выполнение отдельных заданий при наступлении определенных условий.

Подробности о шаблонах см. в разделе *Шаблоны*.

Среда исполнения

Среда исполнения содержит все необходимое для запуска сценариев – фиксированные версии ОС, приложений и библиотек, в том числе Ansible, Python и других. Использование среды исполнения позволяет сделать выполнение сценариев предсказуемым и не зависящим от ОС, в которой развернут контроллер.

Организации, пользователи и команды

В Automation Controller используется управление доступом на основе ролей (*RBAC*), реализованное с помощью организаций, пользователей и команд.

Разграничение доступа

В соответствии с *ролевой моделью доступа* привилегии пользователям Automation Controller предоставляют путем назначения им напрямую или через команды пользователей определенных ролей на следующие типы ресурсов:

- **Полномочия** (Credentials).
- **Среды исполнения** (Execution Environments).
- **Инвентарные списки** (Inventories).
- **Группы исполняющих узлов** (Instance Groups).
- **Шаблоны заданий** (Job Templates).
- **Шаблоны уведомлений** (Notification Template).
- **Проекты** (Project).
- **Шаблоны потоков заданий** (Workflow Job Template).

Графический интерфейс позволяет назначать пользователю роли по отношению к отдельному ресурсу следующими способами:

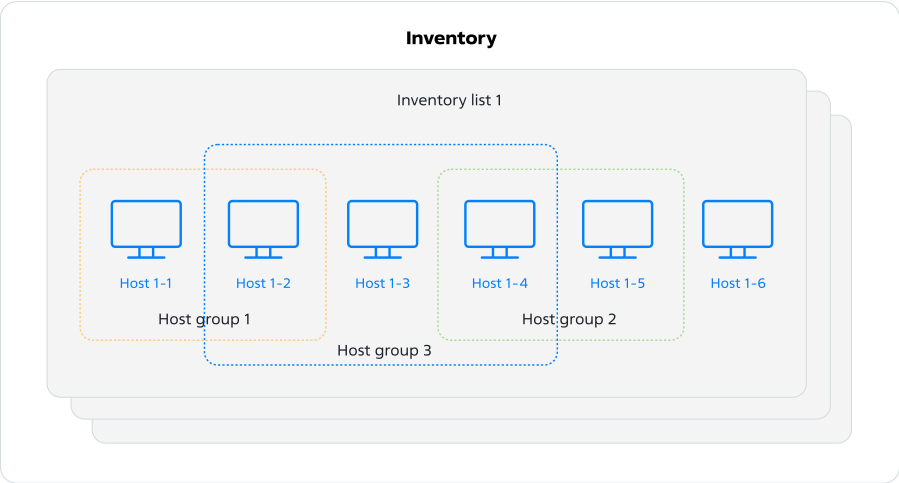
- В окне конкретного ресурса. Инструкции по предоставлению доступа см. в описании конкретного ресурса. Например, чтобы назначить пользователю доступ к проекту, выполните действия, представленные в *описании проекта*.
- В окне *Пользователи (Users)*.

Инвентарь

Инвентарь, связанный с шаблоном заданий, – это набор целевых узлов, на которые направлены задания автоматизации. Привязку этих узлов к шаблону осуществляют через *описание инвентаря (инвентарный список)*.

При создании шаблона задания управляемые узлы из выбранного инвентаря связываются со сценариями проекта.

При описании инвентаря узлы объединяют в группы:



Один и тот же узел может входить в несколько инвентарных списков.

Группы узлов

Группы узлов (inventory groups) используются для логического объединения узлов, входящих в один инвентарь. Один и тот же управляемый узел может входить в несколько групп одновременно.

Группы узлов обладают следующими свойствами:

- Для каждой группы узлов можно задать свои значения переменных Ansible[118].
- При импорте описания инвентаря, содержащего группы узлов, из внешнего источника одноименные группы в инвентаре контроллера создаются автоматически.
- Automation Controller позволяет выполнять отдельные (ad-hoc) команды на узлах выбранной группы без создания шаблонов.

Способы описания инвентаря

В Automation Controller поддерживаются два способа описания инвентаря:

- обычный (standard);
- сборный (constructed).

Способ описания влияет на подключение управляемых узлов к заданиям. Подробности о каждом способе описания инвентаря приведены далее.

Особенности удаления

Удаление описания инвентаря имеет следующие особенности:

- Вместе с обычным инвентарным списком из контроллера удаляются данные обо всех связанных с ним управляемых узлах и группах узлов.
- При удалении обычных и сборных инвентарных списков состав связанных с ними сборных инвентарных списков **не** меняется. Однако, он может измениться, если запустить синхронизацию сборного инвентарного списка.
- Связанные с инвентарным списком шаблоны заданий не удаляются и могут быть связаны с другим инвентарным списком.

Состояния

Описание инвентаря может находиться в одном из состояний:

- **Успех** (Success) – синхронизация описания инвентаря прошла успешно.

[118] https://docs.ansible.com/ansible/latest/playbook_guide/playbooks_variables.html

- **Отключено** (Disabled) – в описание инвентаря не добавлено ни одного источника сведений об управляемых узлах.
- **Ошибка** (Error) – последняя попытка синхронизации описания инвентаря с источником была неудачной.

Статусы управляемых узлов

Контроллер позволяет исключить использование в заданиях управляемых узлов без необходимости изменения существующих или создания новых инвентарных списков, а также внесения изменений в шаблоны заданий. Для этого запись об управляемом узле имеет параметр *статус*, который указывает, может ли управляемый узел быть объектом выполняемых заданий Ansible:

- **Вкл** (On) – может. Этот статус используется по умолчанию при создании и импорте записей об управляемых узлах.
- **Выкл** (Off) – не может. Управляемые узлы с этим статусом исключаются из инвентаря при запуске заданий.

Обычный инвентарь

Обычный инвентарь, также называемый просто инвентарем (inventory), имеет следующие особенности:

- Сведения об управляемых узлах могут быть добавлены вручную или импортированы из внешнего источника, в том числе из файлов инвентаря Ansible, добавленных в проект.
- Поддерживает использование переменных Ansible и групп узлов.

Источники сведений об управляемых узлах

Импорт сведений об управляемых узлах из внешних источников имеет следующие особенности:

- Если в качестве источника выбран отдельный файл, импортируются узлы только из него.
- Если в качестве источника выбран каталог, импортируются узлы из всех размещенных в нем файлов, включая дочерние каталоги.
- Поддерживается фильтрация узлов по названию с помощью регулярных выражений. Этот фильтр применяется в последнюю очередь, после всех остальных фильтров, предоставляемых используемым расширением (plug-in) инвентаря.

  Пусть файл инвентаря содержит записи о тридцати управляемых узлах (диапазон IP-адресов от `10.1.0.1` до `10.1.0.30`):

  ```
  ---
  all:
    hosts:
      # Контроллеры домена
      dc1.example.com:
      dc2.example.com:
      # Сервер печати
      cups.example.com:
      # Клиенты домена
      client01.example.com:
      client02.example.com:
      client03.example.com:
      # ...
      client30.example.com:
  ```

  Для импорта только первых десяти клиентов домена (с `client01.example.com` по `client10.example.com` включительно) следует использовать регулярное выражение следующего вида:

```
client([0][1-9]|10)\.example\.com
```

- В описании импортированных узлов указывается значение **imported**.

  Это значение можно изменить, добавив в файл инвентаря поле `_awx_description` для нужных узлов, например:

```
---
domain-controllers:
  hosts:
    dc1.example.com:
      _awx_description: Primary domain controller
    dc2.example.com:
      _awx_description: First replica of domain controller
```

- Если в файле инвентаря узел принадлежит одной или нескольким группам, в контроллере эти группы будут автоматически созданы, а узел включен в каждую из них.

  Пусть файл инвентаря имеет следующий вид:

```
---
domain-controllers:
  # Контроллеры домена
  hosts:
    dc1.example.com:
    dc2.example.com:
clients:
  # Клиенты домена
  hosts:
    client-1.example.com:
    client-2.example.com:
    client-3.example.com:
    client-4.example.com:
```

  При импорте инвентаря в нем будут автоматически созданы группы `domain-controllers` и `clients`, содержащие контроллеры домена и клиенты домена соответственно.

- Если в источнике группы узлов организованы иерархически, структура групп будет воссоздана при импорте.

Важно

Узлы дочерних групп не связываются с родительской группой автоматически – их нужно добавлять вручную.

Например, пусть есть инвентарь следующего вида:

```
---
all:
  hosts:
    node-[01:10].example.com:
controllers:
  hosts:
    node-01.example.com:
    node-02.example.com:
clients:
  hosts:
    node-[03:10].example.com:
production:
  children:
    controllers:
    clients:
```

При импорте такого инвентаря будут созданы три группы – `production`, `clients`

```
и controllers. Управляемые узлы будут связаны только с группами clients и
controllers.
```

- По умолчанию при повторной синхронизации инвентарного списка с источником импортированные ранее управляемые узлы, группы узлов и переменные не удаляются.
- Поддерживается использование сценариев динамического создания инвентаря.

  Подробности о динамическом создании инвентаря см. в документации Ansible[119].
- Поддерживается синхронизация с источником по расписанию.

Поддержка расширений

Automation Controller поддерживает следующие расширения инвентаря:

Тип источника	Ссылка на руководство
Amazon EC2	amazon.aws.ec2.aws_ec2_inventory[120]
Google Compute Engine	google.cloud.gcp_compute.gcp_compute_inventory[121]
Диспетчер ресурсов Microsoft Azure	azure.azcollection.azure_rm_inventory[122]
VMware vCenter	community.vmware.vmware_vm_inventory[123]
Red Hat Satellite 6	theforeman.foreman.foreman_inventory[124]
OpenStack	openstack.cloud.openstack_inventory[125]
Виртуализация Red Hat	ovirt.ovirt.ovirt_inventory[126]
Платформа автоматизации Red Hat Tower	awx.awx.tower_inventory[127]
Red Hat Insights	redhat.insights.insights_inventory[128]
Terraform State	cloud.terraform.terraform_state[129]

Сборный инвентарь

Сборный инвентарь представляет собой список записей об управляемых узлах, сведения о которых получены путем копирования из обычных инвентарных списков. В этом документе обычные инвентарные списки, из которых копируются данные, называются источниками.

Сборный инвентарь имеет следующие особенности:

- Для формирования сборного инвентаря требуется хотя бы один источник.
- Сборный инвентарь хранит **копии** записей об управляемых узлах и группах управляемых узлов, а не ссылки на них. Это значит, что при обновлении источников сборный инвентарь не обновляется автоматически.
- Существующие группы узлов импортируются из источника независимо от того, будет ли в них хотя бы один узел после выполнения условий отбора узлов.
- Новые группы узлов создаются и заполняются автоматически в соответствии с параметрами, заданными в настройках расширения Ansible для работы с инвентарем.

[119] https://docs.ansible.com/ansible/latest/inventory_guide/intro_dynamic_inventory.html
[120] https://docs.ansible.com/ansible/latest/collections/amazon/aws/aws_ec2_inventory.html
[121] https://docs.ansible.com/ansible/latest/collections/google/cloud/gcp_compute_inventory.html
[122] https://docs.ansible.com/ansible/latest/collections/azure/azcollection/azure_rm_inventory.html
[123] https://docs.ansible.com/ansible/latest/collections/community/vmware/vmware_vm_inventory_inventory.html
[124] https://docs.ansible.com/ansible/latest/collections/theforeman/foreman/foreman_inventory.html
[125] https://docs.ansible.com/ansible/latest/collections/openstack/cloud/openstack_inventory.html
[126] https://docs.ansible.com/ansible/latest/collections/ovirt/ovirt/ovirt_inventory.html
[127] https://docs.ansible.com/ansible/latest/collections/awx/awx/tower_inventory.html
[128] https://github.com/RedHatInsights/insights-host-inventory
[129] https://github.com/ansible-collections/cloud.terraform/blob/main/README.md

- Сведения об управляемых узлах копируются в сборный инвентарь при синхронизации. Для первичного заполнения сборного инвентаря ее необходимо запустить вручную.

Фильтрация узлов

Параметры отбора управляемых узлов задаются в переменных `source_vars` и `limit`.

- `source_vars` – параметры, определяющие название используемого расширения Ansible и правила его использования.

  Примечание

  Для создания сборного инвентаря в Automation Controller используется расширение Ansible ansible.builtin.constructed[130].

- `limit` – правила выборки записей из существующих или временных групп управляемых узлов, сформированных при обработке значения переменной `source_vars`.

В самом простом случае сборный инвентарь формируется путем копирования сведений обо всех управляемых узлах из источников. Значение переменной `source_vars` в этом случае задается следующим образом:

```
---
plugin: ansible.builtin.constructed
```

Здесь `plugin` – переменная, в которой указывается название расширения Ansible, используемого для работы с инвентарем. Это обязательная запись, значение которой должно быть равно `ansible.builtin.constructed` или `constructed`.

Примечание

Рекомендуется использовать полное название расширения – `ansible.builtin.constructed`.

В настройки фильтрации может быть добавлен логический параметр `strict`, значения которого интерпретируются следующим образом:

- `true` – считать любую ошибку обработки сведений об управляемых узлах критической и останавливать создание или обновление сборного инвентарного списка;
- `false` – пропускать управляемые узлы, при обработке сведений о которых возникли ошибки.

Примеры

В рассматриваемых далее примерах в качестве источника используются следующий инвентарь:

```
[controllers]
dc1.example.com
dc2.example.com      timezone=Novosibirsk
dc3.example.com      timezone=Chita
dc4.example.com      timezone=Chita        env=testing

[rubackup]
rb1.example.com
rb2.example.com
rb3.example.com      timezone=Chita
rb4.example.com                            env=testing
```

(продолжается на следующей странице)

[130] https://docs.ansible.com/ansible/latest/collections/ansible/builtin/constructed_inventory.html

(продолжение с предыдущей страницы)

```
[clients]
client1.example.com
client2.example.com
client3.example.com  timezone=Novosibirsk
client4.example.com                        env=testing
client5.example.com  timezone=Novosibirsk  env=testing
client6.example.com  timezone=Chita
client7.example.com  timezone=Chita        env=testing
```

Здесь:

- `controllers` – группа узлов контроллера домена;
- `rubackup` – узлы кластера RuBackup;
- `clients` – клиенты домена;
- `timezone` – часовой пояс, в котором физически находится управляемый узел;
- `env` – окружение, к которому относится управляемый узел.

Пример 1

Пусть необходимо сформировать сборный инвентарь, узлы которого отбираются по следующим признакам:

- часовой пояс не задан или равен `Moscow`;
- окружение тестовое.

В этом случае параметры сборного инвентаря можно задать следующим образом:

- `source_vars`:

```
---
plugin: ansible.builtin.constructed
strict: true
groups:
  timezone_is_moscow: timezone | default("Moscow") == "Moscow"
  env_is_testing: env | default("production") == "testing"
```

- `limit` – `timezone_is_moscow:&env_is_testing`.

Пример 2

Пусть необходимо сформировать сборный инвентарь, узлы которого отбираются по следующим признакам:

- окружение не тестовое;
- не входит в группу `rubackup`.

В этом случае параметры сборного инвентаря можно задать следующим образом:

- `source_vars`:

```
---
plugin: ansible.builtin.constructed
strict: true
groups:
  env_is_not_testing: env | default("production") != "testing"
```

- `limit` – `env_is_not_testing:!rubackup`

Источники кода

Automation Controller позволяет при создании проектов и заполнении инвентаря использовать код из внешних источников. В этом разделе описаны доступные типы источников кода и особенности работы с ними.

При создании проекта может быть использован один из указанных источников:

- ручное управление (локальный каталог);
- репозиторий Git (рекомендуется);
- репозиторий Subversion;
- внешний архив.

Automation Controller хранит копии проектов локально в каталоге, указанном в значении переменной `PROJECTS_ROOT` (по умолчанию `/var/lib/awx/projects/`). Внутри этого каталога контроллер создает подкаталоги для каждого проекта.

При создании или удалении проекта с любым типом источника, кроме «Ручное управление», подкаталог проекта в каталоге проектов соответственно создается или удаляется автоматически.

Ручное управление

Для использования локального каталога в качестве источника кода проекта необходимо соблюдение следующих условий:

- код проекта размещен в одном из подкаталогов каталога проектов;
- владельцами каталога проекта являются пользователь и группа awx;
- на файлы и каталоги проекта установлены следующие режимы доступа:
  - файлы – `0744`;
  - каталоги – `0755`.

Репозиторий Git

Рекомендуемыми источниками проектов для Automation Controller являются репозитории Git. Основной параметр, который при этом должен быть указан – URL репозитория в *VCS*.

URL репозитория в зависимости от используемого протокола подключения может иметь следующий вид:

- HTTPS:

```
https://example.org/project.git
```

- SSH

```
ssh://git@example.org/project.git
```

- GIT

```
git://example.org/project.git
```

Примечание

Поддержка протоколов и особенности их использования определяются возможностями сервиса, в котором размещен репозиторий.

Дополнительные настройки

Для репозитория Git можно указать дополнительные параметры системы управления исходными данными.

Название ветки, название тега или хеш коммита

По умолчанию используется последний коммит из основной ветки репозитория. Эта настройка позволяет использовать другие версии кода:

- Название ветки.

  Используется код из последнего коммита в указанной ветке.

  Например, если в репозитории основная ветка называется `main`, а ветка с экспериментальными возможностями называется `development`, для использования кода из ветки `development` укажите ее название в свойствах источника.

- Название тега.

  Используется код, отмеченный указанным тегом. Например, чтобы использовать коммит, отмеченный тегом `1.7.0`, укажите его название.

- Хеш коммита.

  Используется код из коммита с указанным хешем. Хеш коммита может быть указан полностью или частично (первые несколько символов).

  Для использования коммита с хешем `d7e82ab6ebff598b928edd356803758a3f3435ce` допускается указать значение `d7e82ab6`.

Важно

Некоторые хеши коммитов и ссылки могут быть недоступны без указания спецификации системы управления исходными данными (refspec).

Спецификация (refspec)

Эта настройка устанавливает связь между объектом в удаленном и локальном репозиториях.

Например, чтобы настроить Git на получение содержимого удаленной ветки `release-1.0` в локальную ветку `stable`, используйте следующую спецификацию:

```
refs/heads/release-1.0:refs/heads/origin/stable
```

Важно

Automation Controller использует для работы с удаленными репозиториями Git название `origin`.

Подробности использования спецификаций ссылок см. в документации Git[131].

Полномочия

Эта настройка позволяет указать полномочия типа *Система управления версиями (Source Control)*, которые Automation Controller будет использовать для доступа к репозиторию.

Для создания полномочий через веб-интерфейс воспользуйтесь *инструкцией*.

[131] https://git-scm.com/book/ru/v2/Git-изнутри-Спецификации-ссылок

Настройки управления версиями кода

В настройках Automation Controller можно указать дополнительные параметры, управляющие поведением при работе с репозиторием.

- **Очистить**

  Перед выполнением обновления удаляются все сделанные изменения в локальной копии репозитория.

- **Удалить**

  При обновлении локальная копия репозитория удаляется полностью, после чего загружается заново.

  Примечание

  В зависимости от размера репозитория и скорости соединения процесс загрузки может занять длительное время.

- **Отслеживание подмодулей**

  Если эта настройка включена, отслеживается состояние не только основного репозитория, но и имеющихся подмодулей. Использование этой настройки эквивалентно выполнению команд `git fetch` и `git clone` с параметром `--recurse-submodules`.

  Подробности об использовании подмодулей см. в документации Git[132].

- **Обновить версию при запуске**

  Если эта настройка включена, перед запуском любого задания, использующего код из этого источника, он будет автоматически обновлен.

- **Разрешить переопределение ветки**

  Если эта настройка включена, в шаблоне задания разрешается выбрать ветку или версию кода, отличную от заданной в свойствах проекта.

Репозиторий Subversion

Automation Controller может использовать репозитории Subversion в качестве источников кода проектов. Основной параметр, который при этом должен быть указан – URL репозитория в *VCS*.

URL репозитория в зависимости от используемого протокола подключения может иметь следующий вид:

- HTTP(S):

```
https://example.org/project
```

- SVN

```
svn@example.org/project
```

- SVN+SSH

```
svn+ssh://example.org/project
```

Примечание

Поддержка протоколов и особенности их использования определяются возможностями сервиса, в котором размещен репозиторий.

[132] https://git-scm.com/book/ru/v2/Инструменты-Git-Подмодули

Дополнительные настройки

Для репозитория Subversion можно указать дополнительные параметры:

- номер ревизии;
- полномочия на систему управления исходными данными.

Номер версии

По умолчанию используется самая свежая ревизия кода из основной ветки репозитория. Эта настройка позволяет использовать другие ревизии кода:

- Название ветки.

  Используется последняя ревизия кода в указанной ветке.

- Номер ревизии.

  Используется код из указанной ревизии.

Полномочия

Эта настройка позволяет указать полномочия типа *Система управления версиями (Source Control)*, которые Automation Controller будет использовать для доступа к репозиторию.

Для создания полномочий через веб-интерфейс воспользуйтесь *инструкцией*.

Настройки управления ревизиями кода

В настройках Automation Controller можно указать дополнительные параметры, управляющие поведением при работе с репозиторием.

- **Очистить**

  Перед выполнением обновления удаляются все сделанные изменения в локальной копии репозитория.

- **Удалить**

  При обновлении локальная копия репозитория удаляется полностью, после чего загружается заново.

  Примечание

  В зависимости от размера репозитория и скорости соединения процесс загрузки может занять длительное время.

- **Обновить версию при запуске**

  Если эта настройка включена, перед запуском любого задания, использующего код из этого источника, он будет автоматически обновлен.

- **Разрешить переопределение ветки**

  Если эта настройка включена, в шаблоне задания разрешается выбрать ветку или версию кода, отличную от заданной в свойствах проекта.

Внешний архив

Automation Controller может использовать архивы форматов ZIP и TAR.GZ в качестве источников кода проектов. Основной параметр, который при этом должен быть указан – URL архива, например:

```
https://example.gitflic.ru/user/project/archive/v0.0.1.tar.gz
```

Доступ к архивам возможен по протоколам HTTP и HTTPS.

Примечание

Это решение обычно используют как временное, когда по каким-либо причинам невозможно воспользоваться CMS типа Git. Его практически невозможно использовать для автоматизации процессов уровня предприятия.

Дополнительные настройки

Для внешнего архива доступны следующие дополнительные настройки:

- **Полномочия**

  Эта настройка позволяет указать полномочия типа *Система управления версиями (Source Control)*, которые Automation Controller будет использовать для доступа к архиву в источнике.

  Для создания полномочий через веб-интерфейс воспользуйтесь *инструкцией*.

- **Очистить**

  Перед выполнением обновления удаляются все сделанные изменения в локальной копии репозитория.

- **Удалить**

  При обновлении локальная копия репозитория удаляется полностью, после чего загружается заново.

  Примечание

  В зависимости от размера репозитория и скорости соединения процесс загрузки может занять длительное время.

- **Обновить версию при запуске**

  Если эта настройка включена, перед запуском любого задания, использующего код из этого источника, он будет автоматически обновлен.

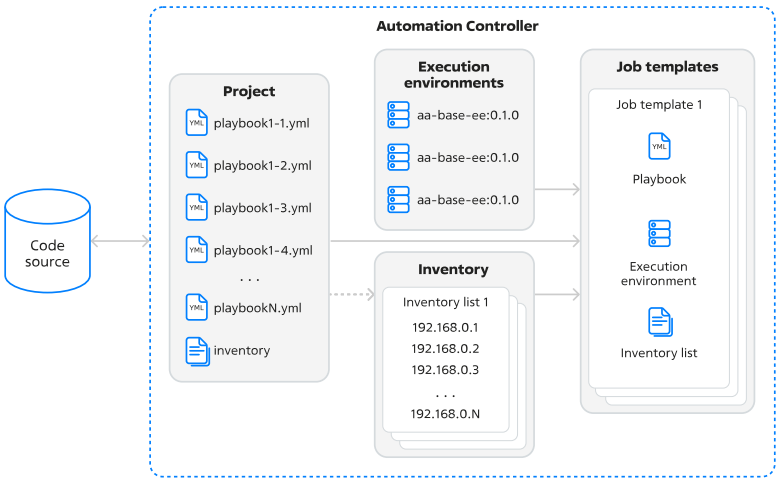
Проекты

В Automation Controller проект – это набор сценариев Ansible. Также проект может содержать описания инвентаря, список зависимостей Ansible и другие файлы.

Наборы сценариев Ansible из проектов используются при создании шаблонов заданий.

Файлы с описаниями инвентаря из проектов можно использовать в качестве источников при заполнении *обычных инвентарных списков*.

Взаимодействие проектов с другими компонентами контроллера показано на схеме:



При создании структуры файлов и каталогов следуйте рекомендациям, приведенным в документе *Типовой проект*.

Проект как источник файлов инвентаря

Automation Controller позволяет использовать проекты для заполнения обычных инвентарных списков.

Поддерживается импорт инвентарных списков из отдельных файлов и каталогов.

Дополнительные параметры проекта

В настройках проекта можно указать дополнительные параметры:

- Среда исполнения.

  Если при создании шаблона не будет выбрана другая среда исполнения, то вместо среды исполнения по умолчанию будет использоваться значение, указанное в настройках проекта.

- Учетные данные для проверки подписи содержимого.

  Указанное полномочие типа «*Открытый ключ GPG*» (GPG Public Key) используется для проверки аутентичности исходного кода проекта.

  Если хотя бы один из подписанных файлов не пройдет проверку, обновление кода проекта будет завершено со статусом **Отказ** (Fail). Также контроллер заблокирует запуск всех заданий на основе шаблонов, использующих код проекта.

- URL системы управления исходными данными.

  Эта настройка доступна для проектов со следующими источниками управления исходными данными:

  - Git
  - Subversion
  - Внешний архив

  Для источников типа «Git» и «Subversion» необходимо указать URL репозитория.

Шаблоны

В этом разделе рассматриваются шаблоны заданий (Job Template) и шаблоны потоков заданий (Workflow Template).

Шаблоны заданий

Шаблон задания (Job Template) связывает между собой *инвентарь* и набор сценариев Ansible из выбранного *проекта*.

Для шаблонов заданий поддерживаются следующие дополнительные параметры:

- тип задания (job type);
- среда исполнения (execution environment);
- полномочия (credentials);
- метки (labels);
- переменные (variables);
- ответвления (forks);
- лимит на количество управляемых узлов (limit);
- степень подробности вывода (verbosity);
- деление на срезы (job slicing);
- таймаут (timeout);
- показ изменений (show changes);
- теги задания (job tags) и пропуск тегов (skip tags);
- повышение привилегий (privilege escalation);
- параллельные задания (concurrent jobs);
- обратные вызовы процесса обеспечения сервиса (provisioning callbacks);
- хранение фактов (enable fact storage);
- запрет перехода на резервную группу узлов управления (prevent instance group fallback);
- настройки webhook (webhook).

Для некоторых параметров доступна опция **Запрос при запуске**. Если она включена, значение для связанного параметра можно задать позже:

- Если задание на основе шаблона запускается вне потока заданий – в момент запуска задания.
- Если шаблон задания используется в шаблоне потока заданий – в момент добавления шаблона задания в шаблон потока заданий.

Тип задания

Поддерживаются задания двух типов:

- Исполнение (Run) – на управляемых узлах будут выполнены необходимые задания автоматизации. Этот тип заданий используется по умолчанию.
- Проверка (Check) – выполняется проверка синтаксиса сценариев, настроек среды исполнения и окружения, в котором будут выполняться задания автоматизации. Изменения в конфигурации управляемых узлов не производятся.

  Использование этого значения эквивалентно запуску команды `ansible-playbook` с параметром `--check`.

Среда исполнения

В настройках шаблона задания можно выбрать *среду исполнения*, используемую для запуска заданий. Эта среда исполнения имеет более высокий приоритет, чем заданная на уровне контроллера, организации и проекта.

Полномочия

Полномочия, используемые для доступа к управляемым узлам.

Поддерживается одновременное использование полномочий разных типов, но не более одной записи каждого типа.

Метки

С помощью меток можно группировать задания, созданные на основе шаблона. Метки также удобно использовать для поиска нужных записей журнала выполнения заданий или их логической группировки.

Переменные

Переменные шаблона задания позволяют задать значения переменных во время выполнения заданий.

Использование этого параметра эквивалентно запуску команды `ansible-playbook` с одним или несколькими аргументами `--extra-vars` (`-e`).

В переменных шаблона задания используется формат `ключ:значение`.

Поддерживаются форматы YAML и JSON.

Подробности о порядке присвоения значений переменным см. в документе *Переменные*.

Ответвления

Количество параллельных процессов, используемых для выполнения сценария.

Если для этого параметра задано значение меньше 1, используется значение по умолчанию, равное 5.

Максимальное значение этого параметра задается в значении *системной настройки* **Максимальное количество ответвлений задания** (Maximum number of forks per job) и по умолчанию равно `200`.

Лимит на управляемые узлы

Фильтр управляемых узлов по их названию.

Если значение не указано, либо равно `all` или `*`, сценарии автоматизации распространяются на все узлы из выбранного инвентарного списка.

Значение по умолчанию: пустая строка (фильтрация узлов не используется).

Подробности о шаблонах названий узлов см. в документации Ansible[133].

Степень подробности

Детализация журнала выполнения задания:

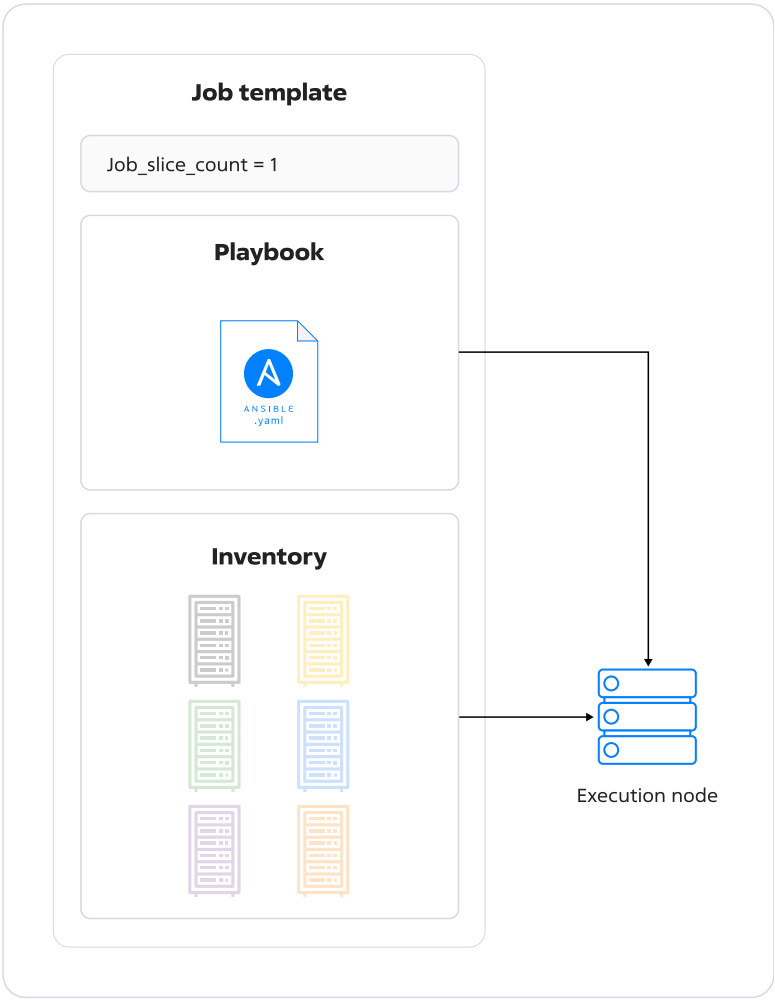
- `0` – нормальный (normal);
- `1` – подробный (verbose);
- `2` – более подробный (more verbose);
- `3` – отладка (debug);
- `4` – отладка подключения (connection debug);
- `5` – отладка WinRM (WinRM debug).

Значение по умолчанию: `0`.

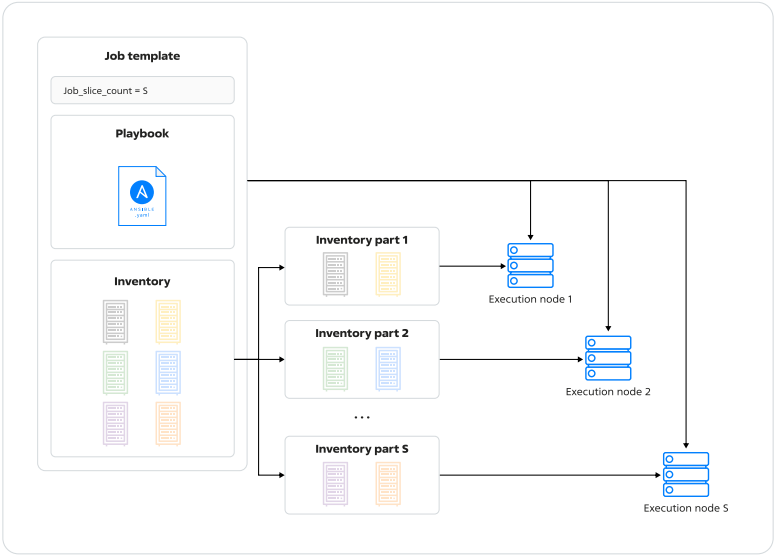
[133] https://docs.ansible.com/ansible/latest/inventory_guide/intro_patterns.html

Деление на срезы

По умолчанию количество срезов равно 1 – все задачи сценариев обычным образом запускаются с одним и тем же инвентарем на одном исполняющем узле.



Если количество срезов больше 1, контроллер делит список узлов исходного инвентаря на указанное количество частей (срезов) и вместо задания запускает поток заданий. Все задания в этом потоке планируются как выполняющиеся параллельно. Каждое задание запускается со своей частью управляемых узлов из исходного инвентаря.



Далее исходное задание называется *разделенным*, а задания потока, сформированного на его основе, – *отдельными* заданиями.

При использовании деления на срезы контроллер реализует следующее поведение:

- Статус выполнения разделенного задания определяется так же как и статус выполнения потока заданий: если выполнение хотя бы одного отдельного задания завершилось ошибкой, неуспешным считается выполнение всего потока.

  Предупреждение

  Не следует использовать деление на срезы для заданий, требующих взаимодействия между управляемыми узлами. Любое отдельное задание может завершиться ошибкой, но Automation Controller не будет пытаться обнаружить или учесть наборы сценариев, выполнение которых завершилось ошибкой при запуске разделенного задания.

- Каждое отдельное задание содержит информацию о шаблоне задания, инвентаре, количестве срезов и порядковом номере среза.
- Выполнение разделенного задания происходит в соответствии с обычным режимом планирования: учитываются настройки ответвлений, постановка в очередь и привязка к предпочтительным узлам.

  Примечание

  Допускается установка крайне высоких значений количества срезов (порядка нескольких тысяч), однако, это может привести к снижению производительности, так как планировщик Automation Controller не предназначен для управления потоками заданий, задания которых одновременно исполняются на тысячах узлов. Для обеспечения высокой производительности рекомендуется использовать количество срезов, не превышающее количество исполняющих узлов.

- В отдельные задания передаются значения указанных пользователем и дополнительных переменных (extra vars), применяются все переменные и *лимиты на управляемые узлы*, указанные в шаблоне задания. Однако, если в отдельное задание переданы лимиты на управляемые узлы, при которых итоговый инвентарный список становится пустым, выполнение задания считается неуспешным.
- Отдельные задания могут выполняться на любом узле, при этом не всегда одновременно (например, из-за высокой нагрузки на систему).
- По умолчанию шаблоны заданий не допускают *параллельный запуск заданий*. Деление на срезы переопределяет это поведение и разрешает одновременное выполнение заданий, даже если эта настройка выключена в свойствах шаблона задания.

Таймаут

Время в секундах на выполнение задания. Если в течение указанного времени задание не выполняется, оно отменяется автоматически.

При значении `0` используется таймаут, заданный в системных настройках контроллера (значение по умолчанию – `0`, время выполнения заданий не ограничивается).

Отрицательное значение разрешает неограниченно долгое выполнение задания.

Значение по умолчанию: `0` (используется значение, заданное в глобальных настройках контроллера).

Показ изменений

Отображение изменений в конфигурации управляемых узлов, сделанных в результате выполнения заданий автоматизации.

Использование этого параметра эквивалентно запуску команды `ansible-playbook` с параметром `--diff`.

Примечание

Для отображения списка изменений необходима поддержка со стороны используемых модулей Ansible.

Теги

Эта настройка позволяет запускать или пропускать задачи, отмеченные определенными тегами.

По умолчанию фильтрация по тегам не используется – выполняются все задачи, описанные в сценарии.

Особенности настройки и применения:

- Названия тегов задаются одной строкой через запятую.
- Если заданы теги задания, то при запуске задания будут выполнены только задачи, отмеченные указанными тегами.

  Использование этого параметра эквивалентно запуску команды `ansible-playbook` с параметром `--tags`.
- Если заданы пропускаемые теги, то при запуске задания задачи с указанными тегами будут пропущены.

  Использование этого параметра эквивалентно запуску команды `ansible-playbook` с параметром `--skip-tags`.

Подробности о тегах см. в документации Ansible[134].

Повышение привилегий

Если эта настройка включена, задачи сценариев выполняются с повышенными привилегиями.

Использование этого параметра эквивалентно запуску команды `ansible-playbook` с параметром `--become`.

Параллельные задания

Если эта настройка включена, на основе шаблона можно запустить одновременное выполнение нескольких заданий с одним и тем же инвентарным списком.

Обратные вызовы процесса обеспечения сервиса

Если эта настройка включена, необходимо указать ключ конфигурации узла. Это URL, по которому узел может связаться с Automation Controller и запросить обновление конфигурации, используя шаблон задания.

Хранилище фактов

Если эта настройка включена, Ansible записывает *факты об узлах* в буфер хранения. Затем собранные факты можно использовать, например, для создания *сборного инвентаря*.

Настройки webhook

Если эта настройка включена, запуск задания на основе шаблона будет происходить при наступлении определенного события в сервисе хранения исходного кода.

При включении этой настройки необходимо задать значения дополнительных параметров:

[134] https://docs.ansible.com/ansible/latest/playbook_guide/playbooks_tags.html

- Сервис – сервис хранения исходного кода.

  Возможные значения:

  - Bitbucket Data Center;
  - GitFlic;
  - GitHub;
  - GitLab.

- Webhook URL – адрес, по которому доступен нужный webhook. Поле заполняется автоматически.
- Ключ webhook – ключ, используемый для подписи данных, отправляемых в Automation Controller. Значение из этого поля следует указать в параметрах webhook используемого сервиса хранения исходного кода.
- Полномочия webhook – полномочия, используемые для передачи данных обратно в webhook.

Шаблоны потоков заданий

Шаблоны потоков заданий (workflow templates) используются для реализации сложной логики управления выполнением заданий. Они поддерживают следующую функциональность:

- Запуск заданий при выполнении определенных условий[135] без необходимости их описания в playbook.
- Обновление кода проектов и инвентарных списков.
- Запуск заданий и потоков заданий только после одобрения пользователей (согласование).
- Параллельный запуск нескольких потоков заданий из одного потока.
- Синхронизация заданий, выполняемых параллельно.
- Запуск потоков заданий по расписанию.
- Рекурсивное использование шаблонов потоков заданий.

  По отношению к родительскому шаблону потока заданий такие шаблоны потоков заданий называются вспомогательными. Артефакты, собранные во время выполнения заданий вспомогательного шаблона, могут быть переданы в следующие узлы.

Шаблон потока заданий представляет собой направленный ациклический граф (*DAG*), начинающийся с корневого узла «НАЧАЛО» (START). Этот узел нельзя изменить или удалить.

Узлами графа могут выступать следующие компоненты и процессы:

- Согласование (approval).

  При переходе к узлу этого типа пользователь с ролью «Одобрить» (Approval) должен подтвердить или отклонить выполнение дальнейших действий.

- Обновление *инвентаря* (inventory update).

  Если инвентарь использует внешние источники, при переходе к узлу этого типа выполняется обновление сведений об управляемых узлах.

- *Шаблон задания* (job template).

  При переходе к узлу этого типа запускается задание на основе указанного шаблона.

- Обновление *проекта* (project update).

  При переходе к узлу этого типа выполняется обновление проекта, если его источником являются репозиторий Git, репозиторий Subversion или внешний архив.

[135] https://docs.ansible.com/ansible/latest/playbook_guide/playbooks_conditionals.html

- Шаблон потока заданий (workflow template).

  В роли узлов этого типа могут выступать уже существующие шаблоны потоков заданий.

- Служебное задание (management job).

  Служебные задания используются для выполнения действий, связанных с внутренним функционированием контроллера. Для узлов этого типа поддерживаются следующие служебные задания:

  - Cleanup Activity Stream – удаление записей в ленте активности;
  - Cleanup Expired OAuth 2 Tokens – очистка токенов OAuth 2 с истекшим сроком действия;
  - Cleanup Expired Sessions – очистка истекших сессий пользователей контроллера;
  - Cleanup Job Details – очистка сведений о выполнении задания.

Ветви графа могут быть одного из трех типов:

- успешное выполнение задания или согласование;
- ошибка при выполнении задания, его отмена или отказ в согласовании;
- безусловный переход, когда результат выполнения предыдущих этапов не имеет значения.

Особенности работы с инвентарем

Инвентарь может быть задан во время создания шаблона потока заданий или во время запуска потока заданий.

Если в настройках шаблона потока заданий включен выбор инвентаря при запуске, инвентарь может быть задан в настройках планировщика или узлов.

Если инвентарь указан в настройка шаблона задания, то значение из настроек шаблона потока заданий игнорируется.

Если в настройках шаблона задания включен выбор инвентаря при запуске, то используется инвентарь, заданный на уровне шаблона потока заданий.

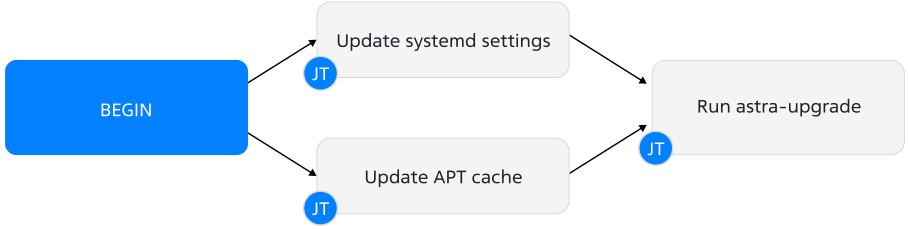
Конвергенция

Если узел имеет несколько входящих ветвей, настройки конвергенции определяют количество выполненных условий, необходимых для запуска задания узла. Поддерживаются следующие значения:

- **Любой** – достаточно успешного выполнения заданий на любом связанном узле;
- **Все** – должны быть успешно выполнены все условия на всех связанных узлах.

Пример

Пусть имеется шаблон потока заданий, показанный на схеме:



Узлы «Update systemd settings» и «Update APT cache» выполняют задания по обновлению настроек systemd и обновлению кеша APT соответственно. При этом указанные задания выполняются параллельно.

Задание «Run astra-upgrade» следует запускать только при успешном выполнении обоих предшествующих заданий. Для этого в настройках конвергенции следует выбрать значение «Все».

Важно

Порядок вычисления значения переменной `set_stats`, хранящей сведения о результатах выполнения задания, не определен. По этой причине для хранения сведений о результатах выполнения заданий рекомендуется использовать уникальные названия ключей.

Дополнительные переменные (extra_vars)

Значения переменных могут быть заданы следующими способами:

- Определение переменных и их значений в свойствах шаблона потока заданий.
- Интерактивный опрос пользователя при запуске потока заданий.
- Дополнительные переменные, переданные в момент запуска потока заданий.

Подробности о порядке присвоения значений переменным см. в документе *Переменные*.

Состояния

Шаблоны потоков заданий могут находиться в одном из следующих состояний:

- Ожидание (Waiting) – для запуска или продолжения выполнения заданий необходимо согласование.
- Выполнение (Running) – выполняются задания.
- Успех (Success) – выполнение задания завершено без ошибок.
- Отмена (Cancel) – выполнение задания отменено.
- Ошибка (Error) – при выполнении заданий произошли ошибки. Для обработки ошибок запущены соответствующие задания.
- Сбой (Failed) – при выполнении заданий произошли ошибки. Продолжение выполнения заданий невозможно.

Опросы

Опросом называется процесс получения значений переменных Ansible путем интерактивного взаимодействия с пользователем. Для этого пользователю, запустившему задание или поток заданий, предлагается ответить на ряд заранее подготовленных вопросов.

Состав вопросов и тип ожидаемых ответов определяется автором опроса.

Опросы обладают следующими свойствами:

- С каждым шаблоном может быть связан только один опрос.
- Каждый вопрос предусматривает получение ответа определенного типа.

  Поддерживаются следующие типы ответов:

  - **Текст** (Text) – строка символов.
  - **Текстовая область** (Textarea) – текст, состоящий из одной или более строк.
  - **Пароль** (Password) – защищенная строка символов.

    Для защиты ввода используется маскировка символов, для защиты значения в таблице базы данных контроллера – защитное преобразование.
  - **Множественный выбор (одиночное выделение)** (Multiple Choice (single select)) – выбор одного значения из предложенного списка.

  - **Множественный выбор (множественное выделение)** (Multiple Choice (multiple select)) – набор записей, выбранных из предложенного списка.
  - **Целое число** (Integer).
  - **Плавающее** (Float) – число с плавающей точкой, например, 3.1415926.
- Для каждого вопроса можно задать значение ответа по умолчанию.

Подробности о порядке присвоения значений переменным см. в документе *Переменные*.

Задания и потоки заданий

В этом документе рассматриваются задания и потоки заданий.

Задания

Задания (jobs) бывают следующих типов:

- запуск playbook (run playbook) – выполнение набора сценариев, указанного в *шаблоне задания*;
- служебное (management jobs) – используется для обслуживания самого контроллера;
- обновление проекта (source control update);
- синхронизация инвентаря (inventory sync);
- команда (command) – выполнение специальных команд (ad-hoc) на выбранных управляемых узлах.

Все задания контроллер ставит в очередь. Задания из очереди выполняются в том порядке, в котором они были в нее добавлены.

Если в шаблоне заданий включено ветвление, для выполнения задания будет использоваться указанное количество параллельных процессов.

Если в шаблоне задания разрешено деление на срезы, связанный с заданием инвентарь делится на части, и задания для управляемых узлов в каждой из них запускаются параллельно.

Запуск заданий может быть привязан к определенной группе узлов контроллера из плоскости исполнения. Для этого в настройках организации, шаблона заданий, шаблона потока заданий или инвентаря необходимо выбрать группу узлов контроллера.

Потоки заданий

Потоки заданий создаются на основе *шаблонов потоков заданий*. При запуске потока заданий обход направленного ациклического графа начинается с точки «НАЧАЛО» (START).

Служебные задания

В контроллере доступны следующие служебные задания:

- Cleanup Activity Stream – очистка истории активности;
- Cleanup Expired OAuth 2 Tokens – очистка токенов OAuth с истекшим сроком действия;
- Cleanup Expired Sessions – очистка данных истекших сессий;
- Cleanup Job Details – очистка истории выполнения заданий.

При выполнении потока заданий некоторые служебные задания вызываются неявно.

Задания синхронизации

Задания синхронизации используются для обновления локальной копии набора сценариев и инвентаря и называются, соответственно, заданиями обновления проекта и заданиями обновления инвентаря.

При выполнении задания обновления проекта контроллер получает версию кода, указанную в настройках *источника*:

- если не указаны ветка, тег, ревизия или хеш коммита – последний коммит в основной ветке проекта;
- если указана ветка – последний коммит в указанной ветке;
- если указан тег – коммит, к которому привязан тег;
- если указана ревизия – указанную ревизию кода.

Если инвентарь использует один из файлов проекта в качестве источника сведений об управляемых узлах, то его автоматическое обновление при обновлении кода проекта не происходит. Для обновления сведений об управляемых узлах необходимо запустить задание обновления инвентаря.

Задания обновления могут находиться в одном из следующих статусов:

- **Назначено** (Pending) – задание создано, но еще не поставлено в очередь и не запущено;
- **Ожидание** (Waiting) – задание поставлено в очередь, но еще не запущено;
- **Выполнение** (Running) – задание выполняется;
- **Успех** (Successful) – задание выполнено без ошибок либо с ошибками, для которых предусмотрен обработчик;
- **Сбой** (Failed) – во время выполнения задания произошли ошибки, обработка которых не предусмотрена.

Управление журналами

Контроллер позволяет просматривать вывод выполнения заданий и специальных команд в реальном времени. Журнал выполнения каждого задания сохраняется в контроллере.

Управление узлами

Группы узлов

Узлы можно объединять в группы узлов (instance groups). Один и тот же узел может входить в несколько групп узлов.

При развертывании контроллера все гибридные и исполняющие узлы автоматически включаются в группу узлов по умолчанию с названием `default`. Название этой группы нельзя изменить.

Выбор группы узлов для запуска заданий

Группы узлов, доступные на уровне контроллера, называются глобальными. По умолчанию задания запускаются на наиболее свободной глобальной группе узлов. Если все глобальные группы узлов заняты, задание ставится в очередь и запускается на узлах первой освободившейся глобальной группы.

Группы узлов могут быть *ассоциированы* с *организациями* (через проекты), *инвентарными списками* и *шаблонами заданий*. В этом случае для запуска заданий используются только ассоциированные группы узлов. Если они будут заняты, контроллер не будет использовать для запуска заданий глобальные группы узлов, а поставит задание в очередь и запустит его на первой освободившейся группе узлов из ассоциированных.

Если группы узлов ассоциированы сразу на нескольких уровнях, выбор групп узлов для запуска заданий выполняется в следующем порядке:

1. шаблон задания;
2. инвентарный список;
3. организация.

Например, пусть в контроллере существуют следующие группы узлов:

- `default`;
- `south`;
- `north`;
- `east`;
- `west`.

Пусть эти группы ассоциированы следующим образом:

1. шаблон задания – `north`;
2. инвентарный список – `east` и `west`;
3. организация – `south`.

В этом случае выбор группы узлов для запуска задания выполняется в следующем порядке:

1. Если свободна хотя бы одна группа, ассоциированная с шаблоном задания, задания запускаются на ее узлах.

   В данном случае проверяется статус группы `north`. Если она занята, выполняется проверка статуса групп узлов, ассоциированных с инвентарным списком.

2. Если свободна хотя бы одна группа, связанная с инвентарным списком, задания запускаются на ее узлах.

   В данном случае проверяется статус групп `east` и `west`. Если обе группы заняты, выполняется проверка статуса групп узлов, связанных с организацией.

3. Если свободна хотя бы одна группа, связанная с организацией, задания запускаются на ее узлах.

   В данном случае проверяется статус группы `south`. Если группа `south` тоже занята, задание ставится в очередь и выполняется на первой освободившейся группе узлов, ассоциированной с шаблоном задания, инвентарным списком или организацией.

Запрет использования резервной группы исполняющих узлов

Запрет перехода на резервную группу узлов изменяет порядок выбора групп узлов для запуска заданий. В зависимости от уровня, на котором он включен, выбор групп узлов выполняется следующим образом:

- Шаблон задания.

  Если шаблон задания ассоциирован с группами узлов, контроллер будет ожидать их освобождения. Если список ассоциированных групп узлов пуст, используются глобальные группы узлов. При этом игнорируются группы, ассоциированные с инвентарным списком и организацией.

- Инвентарный список.

  Если инвентарный список ассоциирован с группами узлов, контроллер будет ожидать их освобождения. Если список ассоциированных групп узлов пуст, используются глобальные группы узлов. При этом игнорируются группы, ассоциированные с организацией.

Переменные

В этом документе описывается порядок разрешения значений переменных Ansible в Automation Controller.

Правило присвоения значения

При присвоении значений переменным используется следующий порядок (значения, указанные на более поздних этапах, заменяют значения, указанные ранее):

1. Значения по умолчанию, заданные в роли (role defaults).
2. Переменные динамического инвентаря (dynamic inventory variables).
3. Переменные инвентаря (inventory variables).
4. Переменные группы узлов (group variables).
5. Переменные управляемого узла (node variables).
6. `group_vars`, связанные с набором сценариев.
7. `host_vars`, связанные с набором сценариев.
8. Факты Ansible (facts).
9. Зарегистрированные переменные (registered variables).
10. Значения, заданные с помощью `set_facts`.
11. Переменные сценария (play variables)
12. Переменные из файлов переменных сценария (play `vars_files`).
13. Переменные роли (role variables) и подключенных (include) файлов.
14. Переменные блока (block variables).
15. Переменные задачи (task variables).
16. Дополнительные переменные из свойств шаблона задания (Job Template extra variables).
17. Переменные, полученные путем интерактивного опроса пользователя при запуске задания на основе шаблона (Job Template Survey). Если опрос пользователя выключен, используются значения по умолчанию.
18. Дополнительные переменные, заданные при запуске задания (Job Launch extra variables).
19. Артефакты задания.
20. Дополнительные переменные из свойств шаблона потока заданий (Workflow Job Template extra variables).
21. Переменные, полученные путем интерактивного опроса пользователя при запуске потока заданий на основе шаблона (Workflow Job Template Survey). Если опрос пользователя выключен, используются значения по умолчанию.
22. Дополнительные переменные, заданные при запуске потока заданий (Workflow Job Launch extra variables).

Пример

Пусть значение переменной `astra_linux_version` задано в нескольких местах:

- инвентарь – `1.7.1`;
- набор сценариев – `1.7.5`;
- шаблон задания – `1.7.3`;
- шаблон потока заданий – `1.7.4`.

Шаблон потока заданий связывает вместе указанные инвентарь, набор сценариев и шаблон задания. Согласно описанному выше порядку, для переменной `astra_linux_version` принимается значение `1.7.4`, заданное последним, на уровне шаблона потока заданий.

Это же значение передается на уровни всех остальных компонентов, связанных с шаблоном потока заданий. Таким образом, при запуске задания и выполнении сценария для переменной `astra_linux_version` также будет использоваться значение `1.7.4`.

Полномочия

Полномочия используются для доступа к внешним ресурсам, управления узлами и проверки целостности содержимого. Полномочия могут содержать следующие данные:

- названия учетных записей;
- пароли;
- токены;
- приватные ключи SSH;
- публичные ключи GPG;
- и другие данные, которые можно использовать для получения доступа к ресурсам и подключения к управляемым узлам.

Конфиденциальные данные, используемые полномочием, называются *секретами*. По умолчанию все секреты хранятся в базе данных Automation Controller. Также Automation Controller может использовать секреты, хранящиеся во *внешних системах управления секретами*.

При создании любых полномочий обязательно должны быть указаны их название и тип. Если ни один из существующих *встроенных типов полномочий* не подходит для выполнения поставленных задач, можно создать *собственный тип полномочий*.

Полномочие может быть связано с организацией, и тогда использовать его могут только пользователи этой организации. Если связи полномочия с организацией нет, то оно доступно для использования всеми пользователями контроллера.

Встроенные типы полномочий

В Automation Controller встроены следующие типы полномочий:

- *ALD Pro*;
- *API-токен Ansible Galaxy/Automation Hub* (Ansible Galaxy/Automation Hub API Token);
- *Astra Automation Platform*;
- *AWS Secrets Manager lookup*;
- *Bitbucket Data Center HTTP Access Token*;
- *Centrify Vault Credential Provider Lookup*;
- *CyberArk Central Credential Provider Lookup*;
- *CyberArk Conjur Secrets Manager Lookup*;
- *GitHub App Installation Access Token*;
- *Google Compute Engine*;
- *HashiCorp Vault Secret Lookup*;
- *HashiCorp Vault Signed SSH*;
- *Insights*;
- *Microsoft Azure Key Vault*;
- *OpenStack*;
- *Red Hat Satellite 6*;
- *Thycotic DevOps Secrets Vault*;
- *Thycotic Secret Server*;

- *Vault*;
- *VMware vCenter*;
- *Web-сервисы Amazon* (Amazon Web Services);
- *Виртуализация Red Hat* (Red Hat Virtualization);
- *Диспетчер ресурсов Microsoft Azure* (Microsoft Azure Resource Manager);
- *Конфигурация terraform backend* (Terraform backend configuration);
- *Личный токен доступа к GitFlic* (GitFlic Personal Access Token);
- *Личный токен доступа к GitHub* (GitHub Personal Access Token);
- *Машина* (Machine);
- *Открытый ключ GPG* (GPG Public Key);
- *Платформа автоматизации Red Hat Ansible* (Red Hat Ansible Automation Platform);
- *Реестр контейнеров* (Container Registry);
- *Сеть* (Network);
- *Система управления версиями* (Source Control);
- *Токен доступа OpenShift или Kubernetes API* (OpenShift or Kubernetes API Bearer Token);
- *Токен персонального доступа GitLab* (GitLab Personal Access Token).

ALD Pro

Важно

Эта часть документации находится в стадии разработки.

API-токен Ansible Galaxy/Automation Hub

Полномочия этого типа предоставляют доступ к репозиториям коллекций Ansible, таким, как Ansible Galaxy[136], *Глобальная библиотека контента* и *Управление контентом*.

Они необходимы для загрузки и установки коллекций, которые требуются для проектов организации, при создании и синхронизации этих проектов. Для использования полномочия требуется выполнение следующих условий:

- наличие в корневом каталоге проекта файла `requirements.yml` со списком зависимостей;
- *связь* созданного полномочия с организацией.

Для создания полномочий этого типа предоставьте следующие данные:

- URL репозитория.

  Важно

  URL репозитория должен завершаться символом `/`.

  По умолчанию коллекции загружаются из репозитория `published`. В этом случае допускается указать сокращенный URL:

  ```
  https://hub.example.com/api/galaxy/
  ```

  Чтобы получить доступ к репозиторию, отличному от `published`, приведите URL к следующему виду:

[136] https://galaxy.ansible.com/

```
https://hub.example.com/api/galaxy/content/<repository>/
```

Здесь <repository> – название репозитория в реестре коллекций.

- Токен доступа, созданный пользователем на стороне используемого реестра контента Ansible.
- Адрес сервера Keycloak для выдачи токена (при использовании SSO).

Чтобы использовать несколько репозиториев одного и того же реестра, создайте для каждого репозитория отдельное полномочие.

TLS:

- требуется валидный сертификат на стороне репозитория коллекций;
- отключить проверку нельзя.

URL:

- задается при создании полномочий;
- по умолчанию при указании доменного имени используется протокол `https`, при необходимости его можно изменить в поле URL.

Astra Automation Platform

Важно

Эта часть документации находится в стадии разработки.

AWS Secrets Manager lookup

Полномочия этого типа предоставляют доступ к секретам, хранящимся в Amazon Web Services Secrets Manager.

Для создания полномочий этого типа предоставьте следующие данные:

- AWS Access Key – ключ доступа к AWS Secrets Manager.
- AWS Secret Key – пароль для доступа к AWS Secrets Manager.

TLS:

- требуется валидный сертификат на стороне AWS;
- отключить проверку нельзя.

URL: нельзя изменить.

Bitbucket Data Center HTTP Access Token

Важно

Эта часть документации находится в стадии разработки.

Centrify Vault Credential Provider Lookup

Полномочия этого типа предоставляют доступ к системе управления секретами Centrify Vault Credential Provider[137].

TLS:

- требуется валидный сертификат на стороне Centrify Vault Credential Provider;

[137] https://developer.delinea.com/docs/vault-functionality

- отключить проверку нельзя.

URL:

- задается при создании полномочий;
- по умолчанию при указании доменного имени используется протокол `https`, при необходимости его можно изменить в поле URL.

CyberArk Central Credential Provider Lookup

Полномочия этого типа предоставляют доступ к системе управления секретами CyberArk Central Credential Provider[138]. Для использования данной интеграции требуется запущенный веб-сервис хранения секретов CyberArk Central Credential Provider.

TLS:

- требуется валидный сертификат на стороне CyberArc Central Credential Provider;
- проверку можно отключить.

URL:

- задается при создании полномочий;
- по умолчанию при указании доменного имени используется протокол `https`, при необходимости его можно изменить в поле URL.

CyberArk Conjur Secrets Manager Lookup

Полномочия этого типа предоставляют доступ к системе управления секретами CyberArk Conjur Secrets Manager[139].

TLS:

- требуется валидный сертификат на стороне CyberArk Conjur Secrets Manager;
- отключить проверку нельзя.

URL:

- задается при создании полномочий;
- по умолчанию при указании доменного имени используется протокол `https`, при необходимости его можно изменить в поле URL.

GitHub App Installation Access Token

Важно

Эта часть документации находится в стадии разработки.

Google Compute Engine

Полномочия этого типа предоставляют доступ к списку управляемых узлов в Google Compute Engine[140] с реквизитами сервисной учетной записи.

Для создания полномочий этого типа необходимо предоставить следующие данные о сервисной учетной записи Google Compute Cloud:

- адрес электронной почты;
- закрытый ключ RSA, связанный с указанным адресом электронной почты.

[138] https://docs.cyberark.com/credential-providers/latest/en/content/ccp/the-central%20-credential-provider.htm
[139] https://www.conjur.org/get-started/why-conjur/how-conjur-works/
[140] https://cloud.google.com/compute/vm-manager/docs/os-inventory/os-inventory-management

Совет

Для автоматического заполнения реквизитов сервисной учетной записи Google Compute Cloud можно использовать файл в формате JSON. Пошаговые инструкции по созданию такого файла см. в документации Google Compute Cloud[141].

TLS:

- требуется валидный сертификат на стороне Google Compute Cloud;
- отключить проверку нельзя.

URL: для этого типа полномочий всегда используется протокол `https`.

HashiCorp Vault Secret Lookup

Полномочия этого типа предоставляют доступ к учетным данным, хранящимся в хранилище секретов HashiCorp Vault[142].

Для создания полномочий этого типа необходимо предоставить следующие данные:

- URL сервера HashiCorp Vault.
- Учетные данные для доступа к серверу HashiCorp Vault.
- Номер версии API.

Подробности об использовании хранилища секретов HashiCorp Vault см. в *соответствующей секции*.

TLS:

- требуется валидный сертификат на стороне сервера HashiCorp Vault;
- отключить проверку нельзя.

URL:

- задается при создании полномочий;
- по умолчанию при указании доменного имени используется протокол `https`, при необходимости его можно изменить в поле URL.

HashiCorp Vault Signed SSH

Полномочия этого типа предоставляют доступ к приватным ключам SSH, хранящимся в HashiCorp Vault[143].

Для создания полномочий этого типа необходимо предоставить следующие данные:

- URL сервера HashiCorp Vault;
- учетные данные для доступа к серверу HashiCorp Vault;
- номер версии API.

Подробности об использовании хранилища секретов HashiCorp Vault см. в *соответствующей секции*.

TLS:

- требуется валидный сертификат на стороне сервера HashiCorp Vault;
- отключить проверку нельзя.

URL: задается при создании полномочий.

[141] https://cloud.google.com/iam/docs/keys-create-delete
[142] https://developer.hashicorp.com/vault/docs/what-is-vault
[143] https://developer.hashicorp.com/vault/docs/secrets/ssh/signed-ssh-certificates

Insights

Полномочия этого типа предоставляют доступ к списку управляемых узлов в Red Hat Insights[144].

TLS:

- требуется валидный сертификат на стороне сервера Red Hat Insights;
- отключить проверку нельзя.

URL: для этого типа полномочий всегда используется протокол `https`.

Microsoft Azure Key Vault

Полномочия этого типа предоставляют доступ к секретам, хранящимся в Microsoft Azure Key Vault[145].

TLS:

- требуется валидный сертификат на стороне Microsoft Azure Key Vault;
- отключить проверку нельзя.

URL:

- задается при создании полномочий;
- по умолчанию при указании доменного имени используется протокол `https`, при необходимости его можно изменить в поле URL.

OpenStack

Полномочия этого типа предоставляют доступ к списку управляемых узлов в OpenStack[146].

TLS:

- требуется валидный сертификат на стороне сервера OpenStack;
- отключить проверку нельзя.

URL:

- задается при создании полномочий;
- по умолчанию при указании доменного имени используется протокол `https`, при необходимости его можно изменить в поле URL.

Red Hat Satellite 6

Полномочия этого типа предоставляют доступ к списку управляемых узлов в Red Hat Satellite 6[147].

TLS:

- требуется валидный сертификат на стороне сервера Red Hat Satellite 6;
- отключить проверку нельзя.

URL:

- задается при создании полномочий;
- по умолчанию при указании доменного имени используется протокол `https`, при необходимости его можно изменить в поле URL.

[144] https://docs.redhat.com/en/documentation/red_hat_insights/1-latest
[145] https://azure.microsoft.com/ru-ru/products/key-vault
[146] https://docs.openstack.org/2024.1/
[147] https://docs.redhat.com/en/documentation/red_hat_satellite/6.15

Thycotic DevOps Secrets Vault

Полномочия этого типа предоставляют доступ к системе управления секретами Thycotic DevOps Secrets Vault.

TLS:

- требуется валидный сертификат на стороне сервера Thycotic DevOps Secrets Vault;
- отключить проверку нельзя.

URL: задается при создании полномочий.

Thycotic Secret Server

Полномочия этого типа предоставляют доступ к серверу секретов Thycotic Secret Server.

TLS:

- требуется валидный сертификат на стороне Thycotic Secret Server;
- отключить проверку нельзя.

URL:

- задается при создании полномочий;
- по умолчанию при указании доменного имени используется протокол `https`, при необходимости его можно изменить в поле URL.

Vault

Полномочия этого типа предоставляют доступ к секретам, защищенным с помощью Ansible Vault.

Если необходим доступ к нескольким хранилищам, необходимо предоставить соответствующие им идентификаторы Vault.

TLS: не используется.

URL: не используется.

VMware vCenter

Полномочия этого типа предоставляют доступ к списку управляемых узлов в VMware vCenter[148].

TLS:

- требуется валидный сертификат на стороне сервера VMware vCenter;
- проверку можно отключить в настройках коллекции `community.vmware.vmware_guest`.

URL:

- задается при создании полномочий;
- по умолчанию при указании доменного имени используется протокол `https`, при необходимости его можно изменить в поле URL.

Web-сервисы Amazon

Полномочия этого типа предоставляют доступ к списку управляемых узлов в Web-сервисах Amazon[149].

Для создания полномочий этого типа необходимо предоставить следующие данные Web-сервисов Amazon:

- публичный ключ доступа;

[148] https://docs.vmware.com/en/VMware-vSphere/index.html
[149] https://aws.amazon.com/documentation-overview/?nc2=h_ql_doc_do

- секретный ключ.

TLS:

- требуется валидный сертификат на стороне AWS;
- отключить проверку нельзя.

URL: для этого типа полномочий всегда используется протокол `https`.

Виртуализация Red Hat

Полномочия этого типа предоставляют доступ к расширению (plug-in) инвентаря `oVirt4.ру`, которым управляют с помощью Виртуализации Red Hat[150].

TLS:

- требуется валидный сертификат на стороне сервера Red Hat Virtualization;
- отключить проверку нельзя.

URL:

- задается при создании полномочий;
- по умолчанию при указании доменного имени используется протокол `https`, при необходимости его можно изменить в поле URL.

Диспетчер ресурсов Microsoft Azure

Полномочия этого типа предоставляют доступ к списку управляемых узлов в Диспетчере ресурсов Microsoft Azure[151].

TLS:

- требуется валидный сертификат на стороне Microsoft Azure;
- отключить проверку нельзя.

URL: при указании доменного имени всегда используется протокол `https`.

Конфигурация terraform backend

Полномочия этого типа предоставляют доступ к удаленному Terraform backend.

TLS:

- требуется валидный сертификат на стороне Terraform backend;
- отключить проверку нельзя.

URL:

- задается при создании полномочий;
- по умолчанию при указании доменного имени используется протокол `https`, при необходимости его можно изменить в поле URL.

Личный токен доступа к GitFlic

Используя полномочия этого типа, Automation Controller через webhook получает доступ к нужным типам событий, происходящих в репозиториях GitFlic[152]. Контроллер реагирует на них, запуская задания и потоки заданий.

TLS:

- требуется валидный сертификат на стороне платформы Astra Automation;

[150] https://access.redhat.com/products/red-hat-virtualization
[151] https://learn.microsoft.com/ru-ru/azure/azure-resource-manager/management/overview
[152] https://gitflic.ru

- отключить проверку нельзя.

URL: URL обратного вызова автоматически создаются платформой при создании полномочий.

Личный токен доступа к GitHub

Используя полномочия этого типа, Automation Controller через webhook получает доступ к нужным типам событий, происходящих в репозиториях GitHub[153]. Контроллер реагирует на них, запуская задания и потоки заданий.

Пошаговые инструкции по созданию токена см. в документации GitHub[154].

TLS:

- требуется валидный сертификат на стороне платформы Astra Automation;
- отключить проверку нельзя.

URL:

- URL обратного вызова автоматически создаются платформой при создании полномочий;
- для этого типа полномочий всегда используется протокол `https`.

Машина

Полномочия этого типа предоставляют доступ к управляемым узлам по SSH. Поддерживается аутентификация по паролю и ключам SSH, созданным с применением различных алгоритмов.

При настройке полномочия можно выбрать один из следующих способов повышения привилегий:

- sudo;
- su;
- pbrun;
- pfexec;
- dzdo;
- pmrun;
- runas;
- enable;
- doas;
- ksu;
- machinectl;
- sesu.

Использование этой настройки эквивалентно выполнению команды `ansible-playbook` с параметром `--become-method=<method>`.

TLS: валидный сертификат не требуется.

URL: не используется.

[153] https://github.com

[154] https://docs.github.com/en/authentication/keeping-your-account-and-data-secure/managing-your-personal-access-tokens

Открытый ключ GPG

Полномочия этого типа используются для проверки подписи содержимого, полученного из *внешних источников*.

TLS: не используется.

URL: не используется.

Платформа автоматизации Red Hat Ansible

Полномочия этого типа предоставляют доступ к Платформе автоматизации Red Hat Ansible[155].

TLS:

- требуется валидный сертификат;
- проверку можно отключить.

URL:

- задается при создании полномочий;
- по умолчанию при указании доменного имени используется протокол `https`, при необходимости его можно изменить в поле URL.

Реестр контейнеров

Полномочия этого типа предоставляют доступ к различным хранилищам образов, например:

- *Управление контентом*;
- Docker[156];
- Quay[157];
- GitLab[158].

TLS:

- требуется валидный сертификат на стороне реестра контейнеров;
- отключить проверку нельзя.

URL: при указании доменного имени всегда используется протокол `https`.

Сеть

Полномочия этого типа используются для управления сетевым оборудованием. К настоящему времени, согласно документации Ansible[159], он устарел и не рекомендован к использованию. Вместо него рекомендуется использовать тип *Машина* (Machine).

TLS:

- требуется валидный сертификат на стороне сетевого оборудования;
- отключить проверку нельзя.

URL: не используется.

[155] https://docs.redhat.com/en/documentation/red_hat_ansible_automation_platform/2.4
[156] https://www.docker.com/
[157] https://quay.io/
[158] https://docs.gitlab.com/user/packages/container_registry/
[159] https://docs.ansible.com/ansible/devel/network/getting_started/network_differences.html#multiple-communication-protocols

Система управления версиями

Полномочия этого типа предоставляют доступ к системам контроля версий на основе Git и Subversion.

Поддерживается доступ с помощью ключа SSH или с использованием имени пользователя и пароля.

TLS:

- требуется валидный сертификат на стороне *VCS*;
- отключить проверку нельзя.

URL: не используется.

Токен доступа OpenShift или Kubernetes API

Полномочия этого типа используются для создания групп с доступом в контейнеры OpenShift или Kubernetes.

Для обеспечения доступа необходимо предоставить следующие данные:

- адрес точки доступа OpenShift или Kubernetes API для аутентификации;
- токен.

TLS:

- требуется валидный сертификат на стороне кластера OpenShift или Kubernetes;
- отключить проверку можно в UI.

URL:

- задается при создании полномочий;
- по умолчанию при указании доменного имени используется протокол `https`, при необходимости его можно изменить в поле URL.

Токен персонального доступа GitLab

Используя полномочия этого типа, Automation Controller через webhook получает доступ к нужным типам событий, происходящих в репозиториях GitLab[160]. Контроллер реагирует на них, запуская задания и потоки заданий.

Пошаговые инструкции по созданию токена см. в документации GitLab[161].

TLS:

- требуется валидный сертификат на стороне платформы Astra Automation;
- отключить проверку нельзя.

URL:

- URL обратного вызова автоматически создаются платформой при создании полномочий;
- по умолчанию при указании доменного имени используется протокол `https`, при необходимости его можно изменить в поле URL.

Собственные типы полномочий

Automation Controller позволяет создавать собственные типы полномочий.

При создании собственного типа полномочий используются следующие понятия:

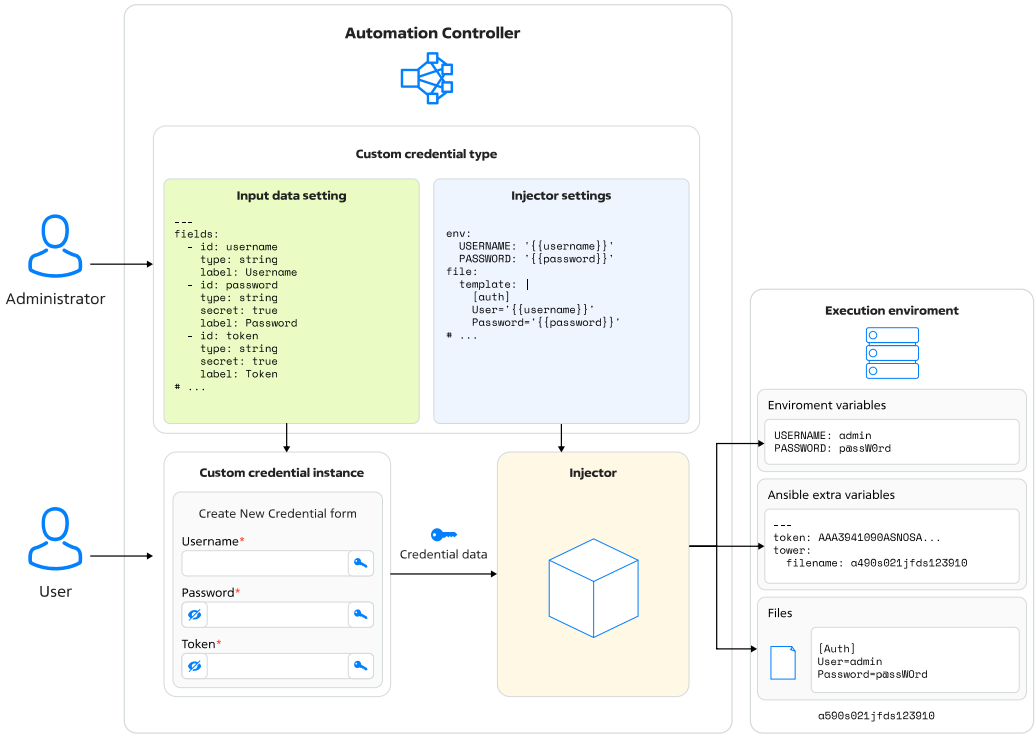
[160] https://gitlab.com
[161] https://docs.gitlab.com/ee/user/profile/personal_access_tokens.html

- *Входные данные* – данные, которые вводит пользователь при создании полномочия. В качестве входных данных могут использоваться имена пользователей, пароли, приватные ключи SSH и GPG, токены и тому подобное.
- *Инжектор* – механизм, выполняющий преобразование данных полномочия в нужный формат.
- *Инжекция* – процесс передачи данных полномочия в *среду исполнения* при выполнении задания.

Создание собственного типа полномочий имеет следующие особенности:

- Каждый собственный тип полномочий должен содержать описание полей входных данных и настройки инжектора.
- Контроллер не проверяет наличие коллизий названий переменных окружения, дополнительных переменных Ansible и пространств имен.
- Следует избегать использования названий переменных, начинающихся с `ANSIBLE_`, поскольку они зарезервированы.

Работа контроллера с собственными типами полномочий показана на схеме:



1. При создании собственного типа полномочий администратор задает настройки входных данных и инжектора.
2. Пользователи создают необходимые экземпляры полномочий собственного типа.
3. При запуске заданий введенные пользователями данные полномочия передаются в инжектор.
4. Инжектор передает данные полномочия в среду исполнения.

   Возможны следующие способы передачи данных полномочия в среду исполнения:

   - переменная окружения;
   - дополнительная переменная (extra vars) Ansible;
   - текстовый файл, в том числе с возможностью генерации содержимого с использованием шаблонизатора Jinja.

   Способ передачи данных определяется настройками инжектора. Одни и те же данные полномочия могут быть переданы в среду исполнения сразу несколькими способами,

например, в переменной окружения, дополнительной переменной Ansible и содержимом текстового файла.

Настройка входных данных

Настройки входных данных типа полномочий определяют параметры полей ввода, используемых при вводе данных полномочия.

При создании поля необходимо указать его идентификатор и тип. Также можно указать дополнительные атрибуты, влияющие на поведение поля формы и обработку введенного пользователем значения.

Типовые настройки входных данных состоят из обязательного блока `fields` и опционального блока `required`.

В блоке `fields` содержится список записей с описанием полей ввода данных полномочия. Каждая запись может состоять из следующих полей:

- `id` – идентификатор поля. Значение, указанное в этом поле, используется для передачи данных в инжектор.
- `type` – тип поля:
  - `string` – строка;
  - `boolean` – логическое.
- `default` – значение по умолчанию.
- `format` – дополнительная проверка корректности значения, указанного в поле.

  Поддерживаются следующие значения:
  - `ssh_private_key` – приватный ключ SSH;
  - `url` – URI.
- `label` – название поля при заполнении сведений о полномочии через графический интерфейс.
- `help_text` – подсказка, которая выводится рядом с полем при заполнении сведений о полномочии через графический интерфейс.
- `secret` – если равно `true`, вводимое значение является секретом.

  Для защиты секретов в полях ввода используется маскировка символов, а в таблице базы данных Automation Controller – защитное преобразование.
- `multiline` – логическое значение для полей типа `string`. При значении `true` поле позволяет вводить многострочный текст.
- `choices` – массив доступных значений для полей типа `string`. Если массив не пуст, при заполнении поля через графический интерфейс значение нужно выбрать из списка, а не вводить вручную.

В блоке `required` содержится список идентификаторов полей, обязательных для заполнения.

Конфигурация инжектора

Для доступа к значению, полученному из входных данных, следует в двойных фигурных скобках указать идентификатор соответствующего поля, например:

```
---
extra_vars:
  auth_token: '{{ auth_token }}'
```

Дополнительной переменной Ansible `auth_token` присваивается значение поля входных данных с идентификатором `auth_token`.

Настройки инжектора могут содержать следующие блоки:

- `file` – шаблон, на основе которого генерируется содержимое временного файла.

  Имя файла генерируется случайным образом. Оно хранится в переменной Ansible `tower.filename`.

  Если используется несколько полей для загрузки файлов, их следует описать в отдельных переменных, названия которых начинаются с префикса `template.`. Для получения имени файла следует обратиться к соответствующему значению словаря `tower.filename`, например:

  ```
  ---
  file:
    template.cert_file: '{{ tls_cert }}'
    template.key_file: '{{ tls_key }}'
  env:
    TLS_CERT_FILE_NAME: '{{ tower.filename.cert_file }}'
    TLS_KEY_FILE_NAME: '{{ tower.filename.key_file }}'
  ```

  Здесь для хранения сертификата и соответствующего ему ключа создаются два временных файла, имена которых сохраняются в переменных `tower.filename.cert_file` и `tower.filename.key_file` соответственно.

- `env` – список названий переменных окружения и соответствующих им значений.
- `extra_vars` – список названий дополнительных переменных Ansible и соответствующих им значений.

  Подробности о порядке присвоения значений переменным см. в документе *Переменные*.

Примеры

Пусть для работы с некоторым сервисом необходимы следующие учетные данные:

- Название учетной записи пользователя – строка, хранится в открытом виде.

  Значение передается в среду исполнения через переменную окружения `API_USERNAME`.

- Токен – строка, хранится в зашифрованном виде.

  Значение передается в среду исполнения через переменную окружения `API_TOKEN` в следующем формате:

  ```
  Bearer-Token: <token>
  ```

- Уровень доступа – строка, может принимать значения `low`, `medium` и `high`.

  Значение по умолчанию: `low`.

  Значение передается в среду исполнения через дополнительную переменную Ansible `api_access_level`.

- Приватный ключ SSH для защиты соединения.

  Значение передается в среду исполнения в виде файла, имя которого хранится в переменной окружения `API_PRIVATE_SSH_KEY`.

Для хранения указанных данных в Automation Controller следует использовать поля со следующими свойствами:

Параметр	Идентификатор	Тип	Обязательное
Название учетной записи	`username`	`string`	Да
Токен	`token`	`string`	Да
Уровень доступа	`access_level`	`string`	Да
Ключ SSH	`ssh_key`	`string`	Нет

Настройки собственного типа полномочий в этом случае можно задать следующим образом:

Список 1: Настройки входных данных

```
---
fields:
  - id: username
    type: string
    label: Username

  - id: token
    type: string
    label: Token
    secret: true
    help_text: A string of 32 charactes long.

  - id: access_level
    type: string
    label: Access level
    choices:
      - low
      - medium
      - high
    default: low

  - id: ssh_key
    type: string
    label: SSH private key
    format: ssh_private_key
    secret: true

required:
  - username
  - token
  - access_level
```

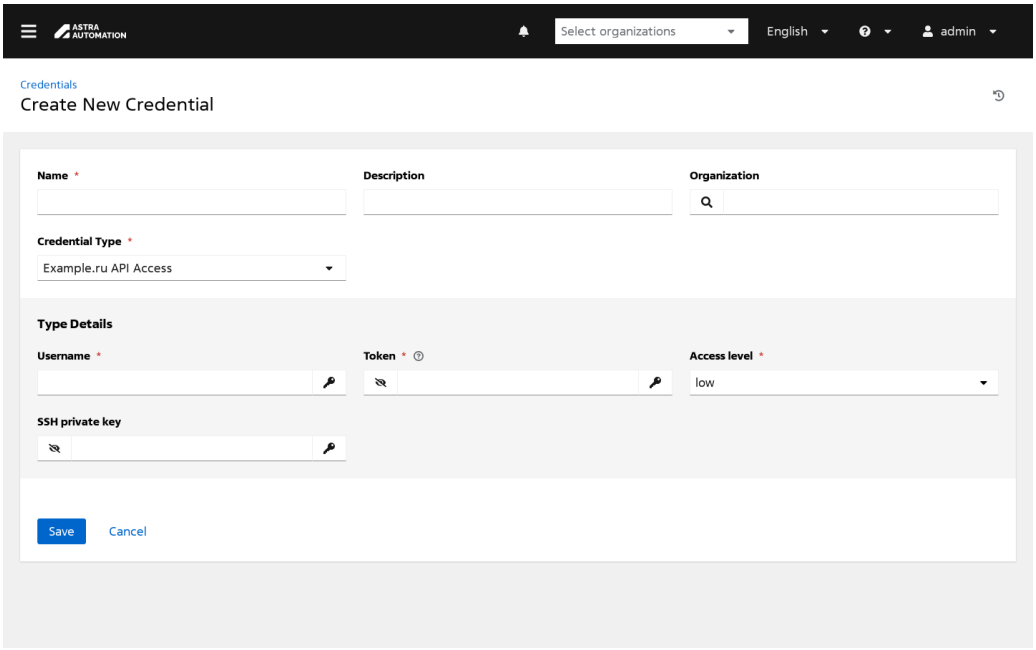
Список 2: Настройки инжектора

```
---
env:
  API_USERNAME: '{{ username }}'
  API_TOKEN: 'Bearer-Token:: {{ token }}'
  API_PRIVATE_SSH_KEY: '{{ tower.filename.private_ssh_key }}'
extra_vars:
  api_access_level: '{{ access_level }}'
file:
  template.private_ssh_key: '{{ ssh_key }}'
```

Примечание

Два двоеточия в строке с параметром API_TOKEN необходимы для соблюдения корректности синтаксиса YAML. При передаче данных строка будет преобразована в указанный выше формат с одним двоеточием.

В графическом интерфейсе Automation Controller форма для создания полномочий этого типа имеет следующий вид:



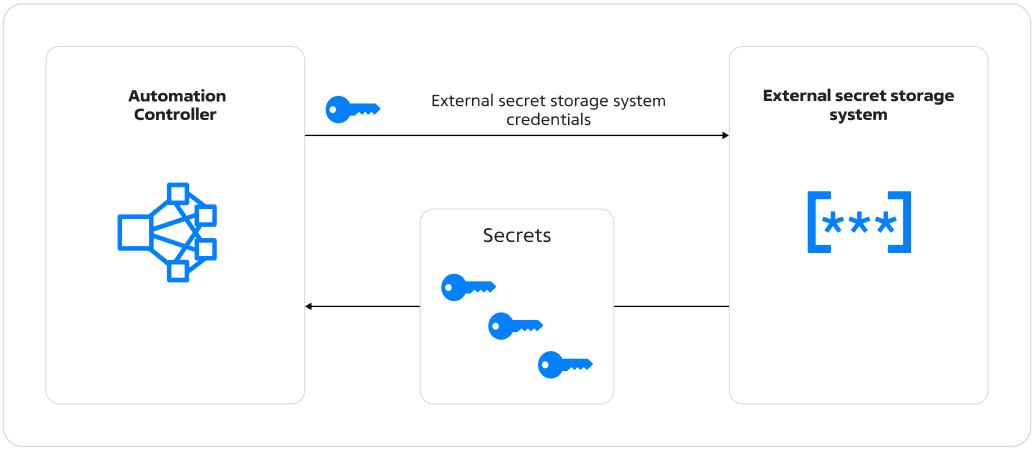
Внешние системы управления секретами

Внешней системой управления секретами называется сторонний сервис, используемый для хранения секретов и управления ими.

Automation Controller может использовать секреты, хранящиеся в следующих системах управления секретами:

- AWS Secrets Manager Lookup;
- Centrify Vault Credential Provider Lookup;
- CyberArk Central Credential Provider Lookup;
- CyberArk Conjur Secrets Manager Lookup;
- HashiCorp Vault;
- Microsoft Azure Key Vault;
- Thycotic DevOps Secrets Vault;
- Thycotic Secret Server.

В общем виде схема работы Automation Controller с секретами, хранящимися во внешней системе управления секретами, показана на схеме:



1. В Automation Controller создается полномочие для доступа к внешней системе управления секретами.

2. В Automation Controller создаются полномочия необходимых типов, используемые в заданиях. При этом вместо ввода значения секретов в соответствующих полях указывается созданное ранее полномочие для доступа к внешней системе управления секретами.
3. При запуске задания Automation Controller обращается к внешней системе управления секретами для получения значений необходимых секретов.

Ниже рассматриваются особенности использования различных внешних систем управления секретами.

AWS Secrets Manager Lookup

Важно

Эта часть документации находится в стадии разработки.

Centrify Vault Credential Provider Lookup

Важно

Эта часть документации находится в стадии разработки.

CyberArk Central Credential Provider Lookup

Важно

Эта часть документации находится в стадии разработки.

CyberArk Conjur Credential Provider Lookup

Важно

Эта часть документации находится в стадии разработки.

HashiCorp Vault

Для доступа к секретам Automation Controller может использовать API HashiCorp Vault версий v1 и v2.

Для аутентификации в HashiCorp Vault доступны следующие методы и средства:

- название учетной записи пользователя и пароль[162];
- AppRole[163];
- Kubernetes[164];
- сертификат TSL[165].
- LDAP[166] (только для HashiCorp Vault редакции Enterprise).

Данные аутентификации необходимо указывать при создании полномочий соответствующих типов:

[162] https://developer.hashicorp.com/vault/docs/auth/userpass
[163] https://developer.hashicorp.com/vault/docs/auth/approle
[164] https://developer.hashicorp.com/vault/docs/auth/kubernetes
[165] https://developer.hashicorp.com/vault/docs/auth/cert
[166] https://developer.hashicorp.com/vault/docs/auth/ldap

- *HashiCorp Vault Secret Lookup*;
- *HashiCorp Vault Signed SSH*.

При аутентификации через AppRole для доступа к HashiCorp Vault можно использовать *собственный тип полномочий* со следующими свойствами:

- Настройка входных данных:

```
---
fields:
  - id: vault_server
    type: string
    label: URL for Vault Server
  - id: vault_role_id
    type: string
    label: Vault AppRole ID
  - id: vault_secret_id
    type: string
    label: Vault Secret ID
    secret: true
required:
  - vault_server
  - vault_role_id
  - vault_secret_id
```

- Конфигурация инжектора:

```
---
extra_vars:
  ansible_hashi_vault_addr: '{{ vault_server }}'
  ansible_hashi_vault_role_id: '{{ vault_role_id }}'
  ansible_hashi_vault_secret_id: '{{ vault_secret_id }}'
  ansible_hashi_vault_auth_method: approle
```

Microsoft Azure Key Vault

Важно

Эта часть документации находится в стадии разработки.

Thycotic DevOps Secrets Vault

Важно

Эта часть документации находится в стадии разработки.

Thycotic Secret Server

Важно

Эта часть документации находится в стадии разработки.

11.1.3 Графический интерфейс

Раздел **Автоматизация процессов** (Automation Execution) является центральной точкой управления всеми процессами автоматизации на платформе Astra Automation. Он предоставляет функциональность для планирования, запуска и мониторинга заданий, управления шаблонами и расписаниями, а также для организации проектов и инфраструктуры, на которой выполняется автоматизация.

В этом разделе администраторы могут выполнять следующие действия:

- создавать и запускать задания;
- управлять шаблонами заданий и рабочих процессов;
- настраивать расписания запуска;
- синхронизировать проекты с системами контроля версий;
- контролировать целевую инфраструктуру и ресурсы;
- выполнять административные действия (очистка истории, настройка уведомлений, утверждения рабочих процессов).

Ниже описаны основные разделы интерфейса **Автоматизация процессов**.

Задания (Jobs)

Раздел **Задания** (Jobs) содержит таблицу со списком запущенных и выполненных заданий автоматизации. В разделе можно выполнять следующие действия:

- фильтрация записей по различным критериям;
- запуск заданий напрямую из таблицы;
- переход к журналам выполнения заданий и результатам каждой задачи.

Каждый элемент содержит информацию о статусе выполнения, времени запуска, типе задания и связанных ресурсах. Этот раздел удобен для ежедневного контроля и оперативного реагирования на события.

Шаблоны

Раздел **Шаблоны** (Templates) включает:

- *шаблоны заданий* (Job Templates);
- *шаблоны рабочих процессов* (Workflow Templates).

Шаблон задания определяет параметры запуска конкретного процесса автоматизации (например, сценарий и переменные). Шаблоны рабочих процессов позволяют объединять несколько заданий в *DAG* с использования визуализации, пауз и утверждений.

Расписания

Раздел **Расписания** (Schedules) используется для задания правил периодического запуска заданий и шаблонов. Он предоставляет следующие возможности:

- создание расписаний для автоматического выполнения задач по времени, дате или дням недели;
- управление исключениями и графиком повторений;
- фильтрация записей, запуск и редактирование свойств заданий.

Проекты

Раздел **Проекты** (Projects) используется для управления хранением и обеспечением доступа к репозиториям со сценариями, ролями и коллекциями. Каждый проект может быть синхронизирован с *VCS* или обновляться вручную.

Инфраструктура

Раздел **Инфраструктура** (Infrastructure) объединяет управление объектами, необходимыми для выполнения автоматизации:

- **Представление топологии** (Topology) – визуализированный обзор инфраструктурных компонентов и их связей;

- **Инвентарные списки** (Inventories) и **Управляемые узлы** (Hosts) – целевые узлы, на которых выполняется автоматизация;
- **Группы исполняющих узлов** (Instance Groups) и **Исполняющие узлы** (Instances) – элементы кластера, распределяющие нагрузку;
- **Среды исполнения** (Execution Environments) – контейнеризованные среды с предустановленными зависимостями для устойчивого и повторяемого выполнения задач;
- **Полномочия** (Credentials) и **Типы полномочий** (Credential Types) – управление доступом к инфраструктуре и внешним сервисам.

Администрирование

Раздел **Администрирование** (Administration) предназначен для администрирования платформы автоматизации. Он предоставляет следующие возможности:

- **Поток активности** (Activity Stream) – аудит изменений и событий в системе с возможностью фильтрации и просмотра подробностей;
- **Согласования потоков заданий** (Workflow Approvals) – обработка запросов на утверждение заданий в рабочих процессах;
- **Источники уведомлений** (Notifiers) – настройка способов отправки оповещений;
- **Служебные задания** (Management Jobs) – запуск системных задач: очистка истории, истекших токенов, пользовательских сессий и истории выполнения заданий.

Задания

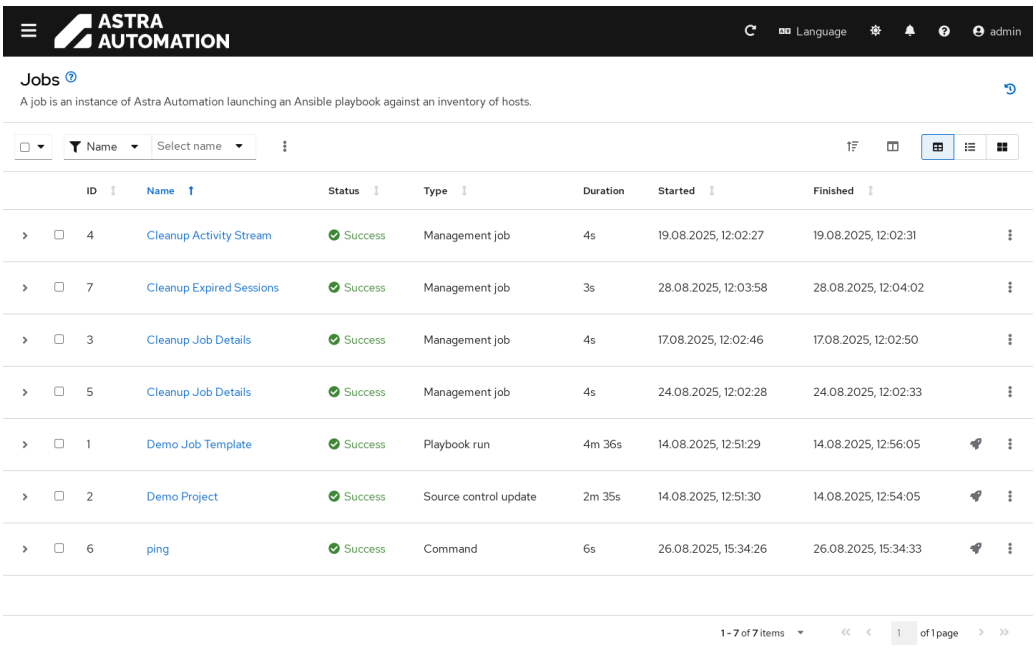
Окно **Задания** (Jobs) предоставляет доступ к журналу выполняемых процессов и позволяет выполнять следующие операции:

- отслеживание текущего состояния рабочих процессов и отдельных задач;
- анализ результатов выполнения сценариев и потоков заданий;
- просмотр журналов выполнения, включая вывод Ansible и код завершения;
- повторный запуск задания при необходимости;
- диагностика ошибок автоматизации на уровне исполнения сценариев.

Для перехода к окну **Задания** (Jobs) выберите на панели навигации *Автоматизация процессов* ▸ *Задания* (*Automation Execution* ▸ *Jobs*).

Таблица заданий

Внешний вид окна **Задания** (Jobs) представлен на схеме:



На панели инструментов размещаются кнопка для выбора всех записей в таблице, поле поиска, кнопка *Создать проект* (Create project) для создания проекта, кнопка ⋮ для вызова действий над выделенными записями, поле сортировки записей, кнопка *Настроить столбцы* (Manage columns) для настройки отображения столбцов, кнопки выбора формата таблицы.

Таблица проектов состоит из следующих столбцов, включая те, отображение которых может быть включено с помощью кнопки *Настроить столбцы* (Manage columns):

- Переключатель подробности вывода основных сведений о задании.

  При нажатии на переключатель выводится дополнительная информация о задании:

  Примечание

  Набор полей зависит от типа задания.

- Флаги для выбора нескольких записей.
- **Идентификатор** (ID) – уникальный идентификатор задания.
- **Название** (Name) – название задания, при нажатии на которое происходит переход в окно просмотра журнала выполнения задания или окно просмотра графа потока заданий.
- **Статус** (Status) – текущий статус выполнения задания:
  - **Успешно** (Successful) – не было ошибок;
  - **Ошибка** (Error) – возникли ошибки, которые были успешно обработаны;
  - **Сбой** (Failed) – возникли необрабатываемые ошибки, выполнение задания прервано;
  - **Отменено** (Canceled) – задание отменено пользователем;
  - **Ожидание** (Pending) – задание находится в очереди;
  - **Выполнение** (Running) – задание выполняется.
- **Тип** (Type) – тип задания.
- **Продолжительность** (Duration) – время, потраченное на выполнение задания.
- **Начато** (Started) – время запуска задания.
- **Закончено** (Finished) – время завершения, отмены или прерывания задания.

- **Источник** (Source) – тип *источника*, на основе которого создано задание синхронизации проекта или инвентарного списка.
- **Запущенный** (Launched by) – ссылка на профиль пользователя, запустившего задание.
- **Расписание** (Schedule) – ссылка на расписание, по которому выполняется запуск задания.
- **Шаблон задания** (Job template) – ссылка на *шаблон задания*.
- **Шаблон потока заданий** (Workflow job template) –ссылка на шаблон потока заданий.
- **Исходное задание потока заданий** (Source workflow job) – ссылка на поток заданий, который инициировал выполнение текущего задания.
- **Инвентарный список** (Inventory) – ссылка на инвентарный список.
- **Проект** (Project) – ссылка на *проект*.
- **Среда исполнения** (Execution environment) – ссылка на среду исполнения, используемую для выполнения задания.
- **Полномочия** (Credentials) – полномочия, используемые для доступа к узлам.
- **Метки** (Labels) – произвольные ярлыки, присвоенные заданию для удобной группировки и поиска.
- **Объяснение** (Explanation) – дополнительное текстовое описание причины запуска задания (часто используется при запуске через вебхук или API).
- **Фрагмент задания** (Job slice) – номер среза и *общее количество срезов*, используемых для выполнения задания.
- **Родитель фрагмента задания** (Job slice parent) – показывает, связано ли задание с механизмом Job Slicing (разделение задания на несколько параллельных частей):
  - **Истина** (True) – текущее задание является родительским, из него были созданы срезы;
  - **Ложь** (False) – задание не связано с разделением на срезы;
- Кнопки для быстрого вызова часто выполняемых действий:
  - перезапуск задания;
  - отмена задания;
  - удаление задания.

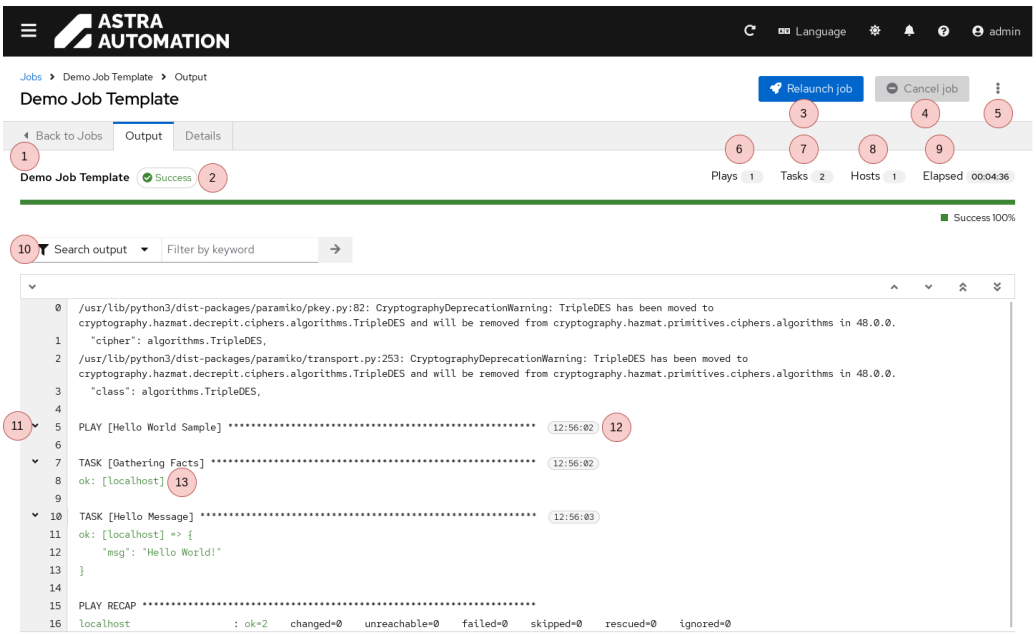
Просмотр вывода Ansible

Для просмотра вывода Ansible нажмите на ссылку с названием задания в таблице заданий. Для заданий в статусе «Выполнение» (Running) используется вывод в реальном времени.

На снимке экрана представлены основные компоненты вкладки **Вывод** (Output).

Примечание

Состав компонентов зависит от типа задания.



Здесь:

1. Название задания.
2. Статус выполнения задания.
3. Кнопка перезапуска задания.
4. Кнопка отмены задания.
5. Кнопки для быстрого вызова часто выполняемых действий:
   - Кнопка сохранения на локальный компьютер файла с выводом Ansible.
   - Кнопка удаления задания из истории.
6. Количество выполненных сценариев.
7. Общее количество выполненных задач (на всех узлах).
8. Количество управляемых узлов.
9. Общее время выполнения задания.
10. Поле выбора заданий потока.
11. Поле для настройки фильтров.

    Поле настройки фильтров позволяет отфильтровать строки в выводе Ansible по различным критериям.
12. Кнопки сворачивания / разворачивания части вывода. Позволяют скрыть часть вывода для более удобного просмотра оставшихся строк.
13. Вывод Ansible.
14. Событие.

    При нажатии на событие открывается окно дополнительных сведений о задаче для конкретного узла, содержащее следующие вкладки:

    - **Подробности** (Details) – подробная информация об узле и задаче.
    - **Data** – журнал выполнения задачи. Доступны кнопки для переключения формата журнала:
      - **JSON**.
      - **YAML**.

- **Вывод** (Output) – консольный вывод выполненных команд.

  Данная вкладка доступна только при наличии соответствующих данных.

- **Стандартная ошибка** (Standard Error) – консольный вывод ошибок выполненных команд.

  Данная вкладка доступна только при наличии ошибок.

Удаление

Чтобы удалить сведения о заданиях, выполните следующие действия:

1. В таблице установите флаги в строках с заданиями, сведения о которых необходимо удалить.
2. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите пункт *Удалить задания* (Delete jobs).
3. Подтвердите удаление.

Шаблоны

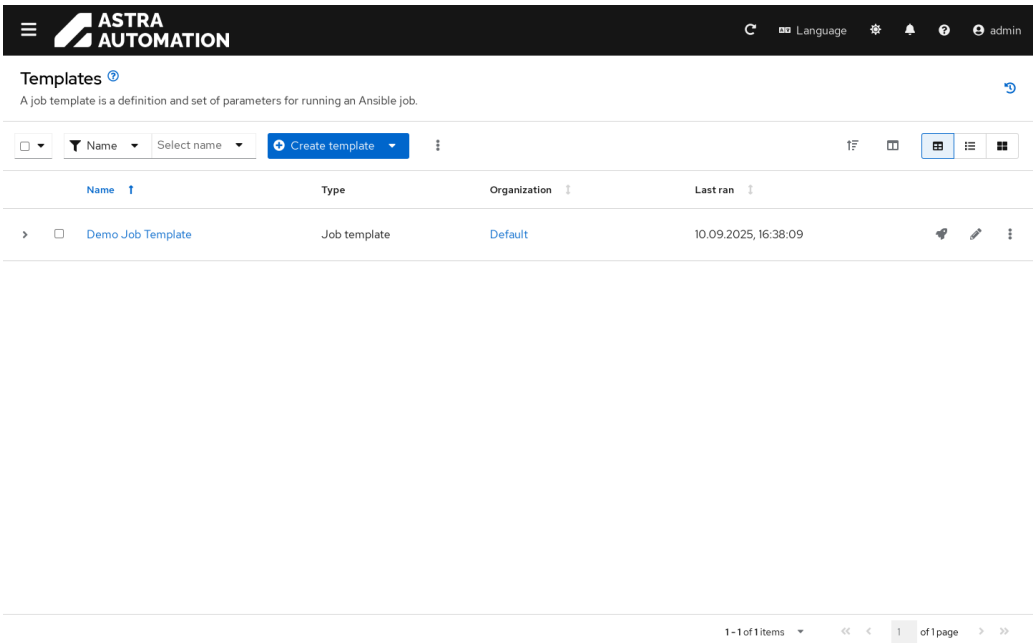
Окно **Шаблоны** (Templates) служит центральным местом для проектирования и управления сценариями с помощью следующих операций:

- создание и редактирование *шаблонов заданий*;
- проектирование и настройка *шаблонов потоков заданий*;
- настройка параметров запуска, полномочий, инвентарных списков и окружения исполнения.

Для перехода к окну **Шаблоны** (Templates) выберите на панели навигации *Автоматизация процессов* ▸ *Шаблоны* (*Automation Execution* ▸ *Templates*).

Таблица шаблонов

Внешний вид окна **Шаблоны** (Templates) представлен на схеме:



На панели инструментов размещаются кнопка для выбора всех записей в таблице, поле поиска, кнопка *Создать проект* (Create project) для создания проекта, кнопка ⋮ для вызова действий над выделенными записями, поле сортировки записей по алфавиту, кнопка *Настроить столбцы* (Manage columns) для настройки отображения столбцов, кнопки выбора формата таблицы.

Таблица проектов состоит из следующих столбцов, включая те, отображение которые может быть включено с помощью кнопки *Настроить столбцы* (Manage columns):

- Флаги для выбора нескольких записей.
- **Название** (Name) – ссылка для перехода в окно просмотра подробностей о шаблоне.
- **Деятельность** (Activity) – статус последнего запущенного задания с использованием этого шаблона.
- **Описание** (Description) – описание шаблона.
- **Тип** (Type) – тип шаблона.
- **Дата создания** (Created) – дата и время создания шаблона.
- **Последнее изменение** (Last modified) – дата и время последнего изменения шаблона.
- **Организация** (Organization) – ссылка на окно просмотра свойств *организации*, которой принадлежит *проект*, связанный с шаблоном.
- **Инвентарный список** (Inventory) – ссылка на инвентарный список.
- **Среда исполнения** (Execution Environment) – ссылка на среду исполнения, используемую для запуска шаблона.
- **Проект** (Project) – ссылка на *проект*.
- **Полномочия** (Credentials) – полномочия, используемые для доступа к узлам.
- **Метки** (Labels) – дополнительные метки, описывающие шаблон, например, dev или `test`. Метки можно использовать для группировки и фильтрации шаблонов и выполненных заданий.
- **Последний запуск** (Last Ran) – дата и время последнего запуска задания или шаблона потока заданий.
- кнопки для быстрого вызова часто выполняемых действий:
  - переход в окно визуализатора графа шаблона потока заданий;
  - запуск задания или потока заданий на основе шаблона;
  - переход в окно редактирования свойств шаблона;
  - создание копии шаблона;
  - удаление шаблона.

Просмотр

Для получения подробных сведений о шаблоне нажмите на его название в таблице шаблонов. Окно просмотра сведений о шаблоне состоит из следующих вкладок:

- **Подробности** (Details) – общие сведения о шаблоне;
- **Доступ пользователей** (User Access) – таблица пользователей и назначенных им *ролей*;
- **Командный доступ** (Team Access) – таблица команд и назначенных им *ролей*;
- **Расписания** (Schedules) – расписание запуска заданий на основе шаблона;
- **Задания** (Jobs) – таблица заданий, созданных на основе шаблона;
- **Опрос** (Survey) – таблица параметров, которые должны быть заданы при запуске заданий на основе шаблона;
- **Уведомления** (Notifications) – таблица уведомлений.

Настройка доступа

Во вкладках **Доступ пользователей** (User Access) и **Командный доступ** (Team Access) выводятся списки пользователей или команд и назначенных им *ролей*.

Назначение ролей отдельным пользователям

Чтобы настроить доступ к шаблону для отдельных пользователей, выполните следующие действия:

1. Во вкладке **Доступ пользователей** (User Access) нажмите кнопку *Добавить роль* (Add roles).
2. Выберите пользователей, которым хотите назначить роли, и нажмите кнопку *Далее* (Next).
3. Выберите роли, которые хотите назначить выбранным на предыдущем шаге пользователям.
4. Нажмите кнопку *Завершить* (Finish).

Назначение командных ролей

Чтобы настроить доступ к шаблону для всех участников одной или нескольких команд, выполните следующие действия:

1. Во вкладке **Командный доступ** (Team Access) нажмите кнопку *Добавить роль* (Add roles).
2. Выберите команды, участникам которых хотите назначить роли, и нажмите кнопку *Далее* (Next).
3. Выберите роли, которые хотите назначить выбранным на предыдущем шаге командам.
4. Нажмите кнопку *Завершить* (Finish).

Отзыв ролей

Чтобы отозвать пользовательские или командные роли, выполните следующие действия:

1. Во вкладке **Доступ пользователей** (User Access) или **Командный доступ** (Team Access) установите флаги в строках с пользователями или командами, соответственно, у которых следует отозвать роли.
2. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите пункт *Удалить роли* (Remove roles).
3. Подтвердите отзыв роли.

Примечание

Роли «Системный администратор» (System Administrator) и «Системный аудитор» (System Auditor) нельзя отозвать через настройку доступа на уровне проекта, однако, можно *изменить тип пользователя*.

Запуск

Для запуска задания или потока заданий выполните следующие действия:

1. В таблице шаблонов нажмите на названием шаблона, который необходимо запустить.
2. Во вкладке **Подробности** (Details) нажмите кнопку *Запустить шаблон* (Launch template).

3. Если в шаблоне настроен и включен опрос – в открывшемся окне **Запрос при запуске** (Prompt on Launch) заполните форму во вкладке **Опрос** (Survey) и нажмите кнопку *Далее* (Next).

   Заданные значения передаются в задание или поток заданий как переменные.

   Подробности о порядке присвоения значений переменным см. в документе *Переменные*.

4. Во вкладке **Обзор** (Review) проверьте корректность введенных данных. Если все значения указаны верно, нажмите кнопку *Завершить* (Finish).

Удаление

Чтобы удалить шаблон, выполните следующие действия:

1. В таблице установите флаги в строках с шаблонами, которые необходимо удалить.
2. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите пункт *Удалить шаблоны* (Delete templates).
3. Подтвердите удаление.

Шаблоны заданий

В этой секции рассматривается создание *шаблонов заданий* и запуск заданий на их основе через веб-интерфейс.

Создание шаблона заданий

Для создания шаблона задания выполните следующие действия:

1. В окне **Шаблоны** (Templates) нажмите кнопку *Создать шаблон* (Create template) и в открывшемся меню выберите **Создать шаблон задания** (Create job template).
2. Заполните форму **Создать шаблон задания** (Create job template):
   - **Название** (Name).

     Особенности заполнения поля:

     - В рамках одной организации не допускается создание шаблонов с одним и тем же названием.
     - Название шаблона не может состоять из одних пробельных символов (пробелы, табуляции и так далее).
     - Допускается использование символов кириллицы и специальных символов.
   - **Описание** (Description) – укажите описание шаблона, например, его назначение или особенности использования.
   - **Тип задания** (Job type) – выберите, будет ли выполнение задания вносить фактические изменения в конфигурацию управляемых узлов:
     - **Выполнить** (Run) – да;
     - **Проверить** (Check) – нет.

     Подробности см. в секции *Тип задания*.
   - **Инвентарный список** (Inventory) – выберите *инвентарный список*, управляемые узлы из которого настраиваются.
   - **Проект** (Project) – выберите *проект*.
   - **Сценарий автоматизации** (Playbook) – выберите файл сценария из состава проекта.

Примечание

Это поле становится доступным после заполнения поля **Проект** (Project).

- **Среда исполнения** (Execution environment) – выберите среду исполнения для запуска заданий.

Примечание

Это поле становится доступным после заполнения поля **Проект** (Project).

- **Полномочия** (Credentials) – укажите полномочия, которые следует использовать для доступа к управляемым узлам.

  Подробности см. в секции *Полномочия*.

- **Метки** (Labels).

  С помощью меток можно группировать задания, созданные на основе шаблона. Их удобно использовать для поиска необходимых записей журнала выполнения заданий или их логической группировки.

- **Ответвления** (Forks) – укажите количество параллельных процессов, используемых для запуска заданий на основе шаблона.

  Подробности см. в секции *Ответвления*.

- **Предел** (Limit) – укажите дополнительные фильтры названий управляемых узлов.

  Подробности см. в секции *Лимит на управляемые узлы*.

- **Уровень подробности** (Verbosity) – укажите степень детализации журнала выполнения заданий на основе шаблона.

  Подробности см. в секции *Степень подробности*.

- **Деление на фрагменты** (Job slicing) – количество частей, на которые следует разделить исходный инвентарный список.

  Подробности см. в секции *Деление на срезы*.

- **Тайм-аут** (Timeout) – укажите ограничение по времени (в секундах) на выполнение задания на основе шаблона.

  Подробности см. в секции *Таймаут*.

- **Показать изменения** (Show changes).

  Если эта настройка включена, при запуске задания на основе шаблона в журнал выводится список изменений, выполненных на управляемых узлах.

  Значение по умолчанию: Выкл (`Off`).

  Подробности см. в секции *Показ изменений*.

- **Группы исполняющих узлов** (Instance groups) – укажите группы узлов контроллера, которые следует использовать для запуска заданий на основе шаблона.

- **Теги заданий** (Job tags) – укажите одной строкой через запятую названия тегов, задачи с которыми следует выполнять при запуске заданий на основе шаблона. Прочие задачи будут пропущены.

  Подробности см. в секции *Теги*.

- **Пропускать теги** (Skip tags) – укажите одной строкой через запятую названия тегов, задачи с которыми следует пропустить при запуске заданий на основе шаблона. Прочие задачи будут выполнены.

  Подробности см. в секции *Теги*.

- **Дополнительные переменные** (Extra variables) – укажите дополнительные переменные, которые следует использовать при запуске заданий на основе шаблона.

  Подробности см. в секции *Переменные*.

- **Повышение привилегий** (Privilege escalation) – укажите, следует ли использовать повышение привилегий при выполнении заданий на основе шаблона.

  Подробности см. в секции *Повышение привилегий*.

- **Обратный вызов для предоставления ресурсов** (Provisioning callback).

  Если эта настройка включена:

  - при выполнении заданий на основе шаблона управляемые узлы могут связаться с контроллером и запросить обновление конфигурации;
  - становится доступным для заполнения поле **Ключ конфигурации узла** (Host Config Key).

- **Ключ конфигурации узла** (Host config key).

  URL, используемый управляемыми узлами для запроса в контроллере обновления конфигурации.

- **Включить веб-перехватчик (webhook)** (Enable webhook).

  Если эта настройка включена, разрешено создание заданий на основе шаблона путем запроса к Webhook. Также становятся доступны для заполнения следующие поля:

  - **Служба webhook** (Webhook service).

    Сервис хранения исходного кода, используемый для работы с Webhook.

    Возможные значения:

    * BitBucket Data Center;
    * GitHub;
    * GitLab.

  - **URL веб-прерывания (webhook)** (Webhook URL).

    URL для работы с Webhook.

    Значение в этом поле формируется автоматически при создании шаблона.

  - **Ключ webhook** (Webhook key).

    Ключ для работы с Webhook.

    Значение в этом поле формируется автоматически при создании шаблона.

  - **Полномочие на webhook** (Webhook credential).

    Полномочия, используемые для доступа к выбранному сервису хранения исходного кода.

- **Одновременные задания** (Concurrent jobs).

  Если эта настройка включена, на основе шаблона можно запустить одновременное выполнение нескольких заданий с одним и тем же инвентарным списком.

- **Разрешить хранение фактов** (Enable fact storage) – укажите, следует ли сохранить в кеш Automation Controller факты, полученные при выполнении заданий на основе шаблона.

  Подробности см. в секции *Хранилище фактов*.

- **Отменить резерв группы исполняющих узлов** (Prevent instance group fallback).

  Если эта настройка включена, для запуска заданий будут использоваться группы узлов, указанные в поле **Группы исполняющих узлов** (Instance groups). Если группы узлов контроллера не выбраны, используются глобальные группы узлов.

Подробности см. в секции *Запрет использования резервной группы исполняющих узлов*.

3. Нажмите кнопку *Создать шаблон задания* (Create job templates).

Запуск задания на основе шаблона

Чтобы запустить задание на основе шаблона, выполните следующие действия:

1. В окне **Шаблоны** (Templates) нажмите на название шаблона задания, который необходимо запустить.
2. Во вкладке **Подробности** (Details) нажмите кнопку *Запустить шаблон* (Launch template).
3. Если в настройках шаблона включен опрос, откроется окно **Запрос при запуске** (Prompt on Launch).
    1. Во вкладке **Опрос** (Survey) заполните значения полей.
    2. Нажмите кнопку *Далее* (Next).
    3. Во вкладке **Предварительный просмотр** (Review) проверьте настройки запуска задания. Если все значения заданы верно, нажмите кнопку *Завершить* (Finish).

Шаблоны потоков заданий

В этом документе рассматриваются следующие операции, связанные с *шаблонами потоков заданий*:

- создание;
- запуск.

Создание шаблонов потоков заданий

Для создания шаблона потока заданий выполните следующие действия:

1. В окне **Шаблоны** (Templates) нажмите кнопку *Создать шаблон* (Create template) и в открывшемся меню выберите **Создать шаблон потока заданий** (Create workflow job template).
2. Заполните форму **Создать шаблон потока заданий** (Create workflow job template):
    - **Название** (Name).

      Особенности заполнения поля:

        - В рамках одной организации не допускается создание шаблонов с одним и тем же названием.
        - Название шаблона не может состоять из одних пробельных символов (пробелы, табуляции и так далее).
        - Допускается использование символов кириллицы и специальных символов.
    - **Описание** (Description) – укажите описание шаблона, например, его назначение или особенности использования.
    - **Организация** (Organization) – укажите организацию, которой принадлежит шаблон потока заданий.
    - **Инвентарный список** (Inventory) – укажите инвентарный список.

      Важно

      Значение в этом поле переопределяет значения, заданные на уровне шаблонов заданий.

- **Предел** (Limit) – укажите дополнительные фильтры названий управляемых узлов.

  Подробности см. в секции *Лимит на управляемые узлы*.

- **Ветвь системы управления исходным кодом** (Source control branch) – если при запуске заданий следует использовать ветку, отличную от основной ветки проекта, укажите ее название в этом поле.

  Важно

  Название ветки, указанное в этом поле, применяется ко всем шаблонам заданий, используемым в потоке.

- **Метки** (Labels).

  С помощью меток можно группировать задания, созданные при выполнении потока. Их удобно использовать для поиска необходимых записей журнала выполнения заданий или их логической группировки.

- **Теги заданий** (Job tags) – укажите одной строкой через запятую названия тегов, задачи с которыми следует выполнять при запуске заданий на основе шаблона. Прочие задачи будут пропущены.

  Подробности см. в секции *Теги*.

- **Пропускать теги** (Skip tags) – укажите одной строкой через запятую названия тегов, задачи с которыми следует пропустить при запуске заданий на основе шаблона. Прочие задачи будут выполнены.

  Подробности см. в секции *Теги*.

- **Дополнительные переменные** (Extra variables) – укажите дополнительные переменные, которые следует использовать при запуске заданий потока.

  Подробности о порядке разрешения переменных см. в документе *aac_management_workflow_templates_extra_vars*.

- **Включить веб-перехватчик (webhook)** (Enable webhook).

  Если эта настройка включена, разрешено создание потока заданий на основе шаблона путем запроса к Webhook. Также становятся доступны для заполнения следующие поля:

  - **Служба webhook** (Webhok service).

    Сервис хранения исходного кода, используемый для работы с Webhook. Возможные значения:

    * BitBucket Data Center;
    * GitHub;
    * GitLab.

  - **URL веб-прерывания (webhook)** (Webhook URL)

    URL для работы с Webhook.

    Значение в этом поле формируется автоматически при создании шаблона.

  - **Ключ webhook** (Webhook Key)

    Ключ для работы с Webhook.

    Значение в этом поле формируется автоматически при создании шаблона.

- **Разрешить одновременные задания** (Enable concurrent jobs).

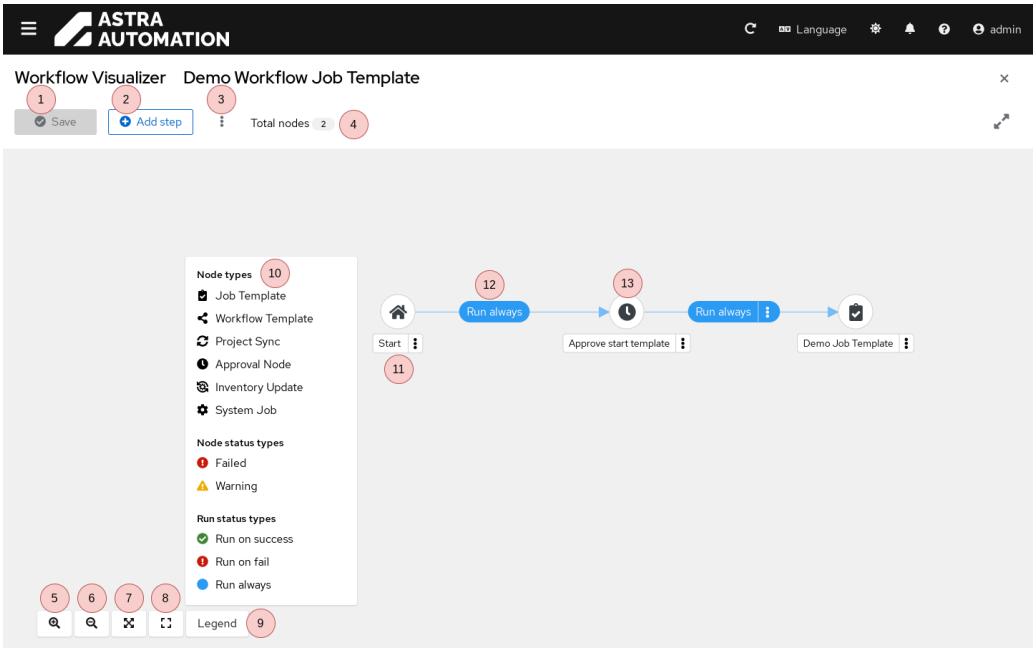
  Если эта настройка включена, на основе шаблона можно запустить одновременно несколько потоков заданий с одним и тем же инвентарным списком.

3. Нажмите кнопку *Создать шаблон потока заданий* (Create workflow job templates).

Визуализатор шаблонов потоков заданий

После создания шаблона потока заданий визуализатор запускается автоматически. Для его запуска для существующего шаблона потока заданий выполните следующие действия:

1. В таблице шаблонов нажмите на название шаблона потока заданий.
2. Нажмите кнопку *Посмотреть визуализатор потока заданий* (View workflow visualizer).



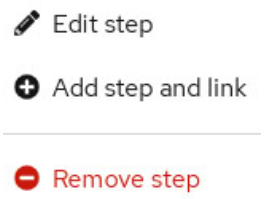
Окно визуализатора содержит следующие элементы:

1. Кнопка сохранения изменений.
2. Кнопка добавления шага.
3. Кнопка вызова дополнительных действий:
   - переход к документации;
   - запуска потока заданий;
   - удаление всех шагов.
4. Общее количество узлов. При расчете значения узел «Начать» (Start) не учитывается.
5. Кнопка увеличения масштаба.
6. Кнопка уменьшения масштаба.
7. Кнопка изменения масштаба под размер экрана.
8. Кнопка сброса настроек масштаба.
9. Переключатель отображения легенды.
10. Панель легенды.
11. Узел «Начать» (Start).
12. Ссылка на следующий узел.

    Цвет ссылки указывает на условие, при котором выполняется переход:
    - красный – ошибки при выполнении задания или отказ в согласовании;
    - зеленый – успешное выполнение задания или согласование;
    - синий – безусловный переход.
13. Индикатор типа узла. В данном случае – «Согласование» (Approval).

При нажатии на кнопку ⋮ рядом с названием узла или условием ссылки появляется контекстное меню.

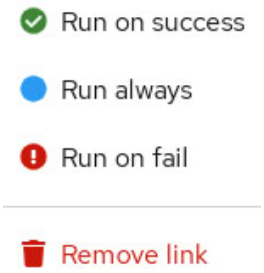
Контекстное меню узла:



Контекстное меню узла содержит кнопки для выполнения следующих действий:

- редактирование узла;
- создание нового узла и связь с ним;
- удаление узла.

Контекстное меню ссылки:



Контекстное меню ссылки содержит кнопки для выполнения следующих действий:

- изменение условия ссылки;
- удаление ссылки.

Начало работы

Чтобы начать работу с графом нового шаблона потока заданий, выполните следующие действия:

1. Нажмите кнопку **Добавить шаг** (Add step).
2. В диалоговом окне **Добавить шаг** (Add step) выберите тип узла.
3. Заполните поля, специфичные для выбранного типа узла. В зависимости от типа узла могут быть доступны дополнительные настройки.
4. Нажмите кнопку *Далее* (Next).
5. Во вкладке **Обзор** (Review) проверьте настройки узла и нажмите кнопку *Завершить* (Finish).

Первый добавленный в граф узел связывается с узлом «Начать» (Start), который имеет следующие особенности:

- его нельзя изменить или удалить;
- тип ссылки на другие узлы – только «Всегда выполнять» (Run always).

Создание узлов

Чтобы добавить в граф новый узел, выполните следующие действия:

1. Вызовите контекстное меню узла.
2. Нажмите кнопку *Добавить шаг и ссылку* (Add step and link). Это приводит к открытию диалогового окна **Добавить шаг** (Add step).

3. В поле **Статус** (Status) выберите условие перехода к создаваемому узлу:
   - **Запускать при успехе** (Run on success) – согласование или успешное выполнение задания;
   - **Запускать при неудаче** (Run on failure) – отказ в согласовании или ошибки при выполнении задания;
   - **Запускать всегда** (Always run) – переход выполняется в любом случае.
4. Нажмите кнопку *Далее* (Next).
5. В поле **Тип узла** (Node type) выберите тип создаваемого узла.
6. Заполните поля, соответствующие выбранному типу узла:
   - **Согласование** (Approval).

     Узлы этого типа используются для создания согласований – запросов к пользователю на подтверждение перехода к следующему заданию в потоке.

     Укажите параметры согласования.

     – **Название** (Name) – название узла.

       Указанное название выводится в таблице согласований.

     – **Описание** (Description) – описание узла.

     – **Таймаут** (Timeout) – время на согласование.

       Если в полях для минут и секунд указаны значения `0`, время на согласование не ограничено.

   - **Синхронизация источников инвентаризации** (Inventory Source Sync).

     Узлы этого типа создают задание синхронизации инвентарного списка с источником.

     – **Источник инвентаризации** (Inventory source) – выберите источник сведений об управляемых узлах.

   - **Шаблон задания** (Job Template).

     Узлы этого типа используются для запуска заданий на основе выбранного шаблона задания.

     Если в настройках шаблона задания включен опрос, при нажатии на кнопку *Далее* (Next) происходит переход к шагу опроса.

     Подробности об опросах см. в документе *Опросы*.

     – **Шаблон задания** (Job template) – выберите шаблон задания.

   - **Синхронизация проекта** (Project Sync).

     Узлы этого типа создают задание синхронизации кода проекта с источником.

     – **Проект** (Project) – выберите проект.

   - **Шаблон потока заданий** (Workflow Job Template).

     Узлы этого типа запускают поток заданий на основе выбранного шаблона потока заданий.

     – **Шаблон потока заданий** (Workflow job template) – выберите шаблон потока заданий.

   - **Служебное задание** (Management Job).

     Узлы этого типа создают служебное задание.

     При выборе заданий «Cleanup Activity Stream» и «Cleanup Job Details» появляется дополнительное поле для заполнения – **Количество дней, в течение которых будут храниться данные** (Days of data to be retained). Все записи старше указанного периода будут удалены.

Значение по умолчанию: 30.

Подробности о служебных заданиях см. в секции *Служебные задания*.

7. Заполните поля **Преобразование** (Convergence) и **Псевдоним узла** (Node alias):
   - **Преобразование** (Convergence) – укажите количество условий, которые должны быть выполнены для перехода к этому узлу потока заданий.

     Подробности см. в секции *Конвергенция*.
   - **Псевдоним узла** (Node alias) – если в этом поле указано какое-либо значение, при просмотре графа потока заданий оно выводится вместо названия ресурса, используемого для создания узла.
8. Нажмите кнопку *Далее* (Next).
9. Во вкладке **Обзор** (Review) проверьте настройки узла и нажмите кнопку *Завершить* (Finish).

Удаление узлов и ссылок

Чтобы удалить узел или ссылку, выполните следующие действия:

1. Вызовите контекстное меню узла или ссылку.
2. Нажмите кнопку удаления.
3. Подтвердите удаление.

Запуск потока заданий

Чтобы запустить поток заданий, выполните следующие действия:

1. В таблице шаблонов нажмите на название шаблона потоков заданий, который необходимо запустить.
2. Во вкладке **Подробности** (Details) нажмите кнопку *Запустить шаблон* (Launch template).

Опросы

Для управления *опросами* Automation Controller предоставляет интуитивно понятный пользовательский интерфейс.

При просмотре подробных сведений о шаблоне задания или шаблоне потока заданий во вкладке **Опрос** (Survey) выводится список вопросов, задаваемых пользователю при запуске заданий или потока заданий.

Примечание

В этом документе для краткости шаблоны заданий и шаблоны потоков заданий именуются общим термином «шаблоны».

Таблица вопросов состоит из следующих колонок:

- Флаги для выбора нескольких записей.
- **Название** (Name) – текст вопроса, задаваемого пользователю.
- **Тип** (Type).
- **По умолчанию** (Default).

  Значение по умолчанию. Если ответ имеет тип «Пароль» (Password), в этой колонке выводится значение ЗАШИФРОВАНО (ENCRYPTED).
- Кнопка для быстрого перехода в окно редактирования вопроса.

- Кнопка для удаления вопроса.

Переключение состояния

Чтобы при запуске задания или потока заданий пользователю были заданы вопросы, связанные с шаблоном, необходимо включить соответствующий опрос. Для изменения статуса опроса выполните следующие действия:

1. В таблице шаблонов нажмите на названием шаблона.
2. Выберите вкладку **Опрос** (Survey).
3. Переведите переключатель состояния опроса в необходимое положение.

Создание вопросов

Чтобы добавить вопросы в опросник, выполните следующие действия:

1. В таблице шаблонов нажмите на названием шаблона.
2. Во вкладке **Опрос** (Survey) нажмите кнопку *Создать вопрос для опроса* (Create survey question).
3. Заполните форму **Добавить вопрос** (Add Question):
   - **Вопрос** (Question) – укажите текст вопроса, который выводится в форме при запуске задания или потока заданий.
   - **Описание** (Description) – укажите справочную информацию, которая поможет пользователю принять решение при ответе на вопрос.
   - **Имя переменной ответа** (Answer variable name) – укажите имя переменной Ansible, значение которой пользователь задает при ответе на вопрос.

     Требования к имени переменной:

     – уникальность на уровне шаблона;

     – может состоять из символов латинского алфавита, цифр и знака _;

     – не может содержать пробельные символы.
   - **Тип ответа** (Answer type) – выберите тип ответа, который может ввести пользователь.

     Подробности о поддерживаемых типах ответов см. в документе *Опросы*.
   - **Требуемый** (Required) – если эта настройка включена, ответ пользователя на вопрос обязателен.
4. В зависимости от выбранного типа ответа заполните соответствующие поля:
   - **Текст** (Text).

     Совет

     Для ввода конфиденциальных данных рекомендуется использовать ответ типа «Пароль» (Password).

     Поддерживаются следующие опциональные поля:

     – **Минимальная длина** (Minimum length) – минимальное количество символов в строке.

       Значение по умолчанию: `0`.

     – **Максимальная длина** (Maximum length) – максимальное количество символов в строке.

       Значение по умолчанию: `1024`.

  - **Ответ по умолчанию** (Default answer)

- **Область ввода текста** (Textarea).

  Поддерживаются следующие опциональные поля:

  - **Минимальная длина** (Minimum length) – минимальное количество символов во введенном тексте.

    Значение по умолчанию: `0`.

  - **Максимальная длина** (Maximum length) – максимальное количество символов во введенном тексте.

    Значение по умолчанию: `1024`.

  - **Ответ по умолчанию** (Default answer)

- **Пароль** (Password).

  Поддерживаются следующие опциональные поля:

  - **Минимальная длина** (Minimum length) – минимальное количество символов в строке.

    Значение по умолчанию: `0`.

  - **Максимальная длина** (Maximum length) – максимальное количество символов в строке.

    Значение по умолчанию: `1024`.

  - **Ответ по умолчанию** (Default answer)

- **Множественный выбор (одиночный выбор)** (Multiple Choice (single select)).

  Чтобы создать элемент списка, введите значение в поле **Варианты выбора** (Muptiple Choice Options) и нажмите `Enter`.

  Чтобы сделать один из элементов списка значением по умолчанию, включите флаг в соответствующей строке.

- **Множественный выбор (множественное выделение)** (Multiple Choice (multiple select)).

  Чтобы создать элемент списка, введите значение в поле **Варианты выбора** (Multiple Choice Options) и нажмите `Enter`.

  Чтобы сделать пункты выбранными по умолчанию, включите флаги в соответствующих строках.

- **Целое число** (Integer).

  Поддерживаются следующие опциональные поля:

  - **Минимум** (Minimum) – минимально допустимое значение.

    Значение по умолчанию: `0`.

  - **Максимум** (Maximum) – максимально допустимое значение.

    Значение по умолчанию: `1024`.

  - **Ответ по умолчанию** (Default answer).

- **Плавающий** (Float).

  Поддерживаются следующие опциональные поля:

  - **Минимум** (Minimum) – минимально допустимое значение.

    Значение по умолчанию: `0`.

  - **Максимум** (Maximum) – максимально допустимое значение.

    Значение по умолчанию: `1024`.

  - **Ответ по умолчанию** (Default answer).

5. Нажмите кнопку *Создать вопрос для опроса* (Create survey question).

Удаление вопросов

Чтобы удалить вопросы, выполните следующие действия:

1. В таблице шаблонов включите флаги в строке вопросов, которые необходимо удалить.
2. Нажмите кнопку ⋮ и в открывшемся меню выберите пункт *Удалить вопросы опроса* (Delete survey questions).
3. Подтвердите удаление.

Расписания

Окно **Расписания** (Schedules) позволяет автоматизировать запуск сценариев по расписанию с помощью следующих операций:

- создание и редактирование расписаний запуска сценариев;
- определение циклов выполнения: разовый запуск, периодичность и так далее;
- контроль предстоящих и завершенных автоматических выполнений.

Для перехода к окну **Расписания** (Schedules) выберите на панели навигации *Автоматизация процессов* ▸ *Расписания* (*Automation Execution* ▸ *Schedules*).

Таблица расписаний

Внешний вид окна **Расписания** (Schedules) представлен на схеме:

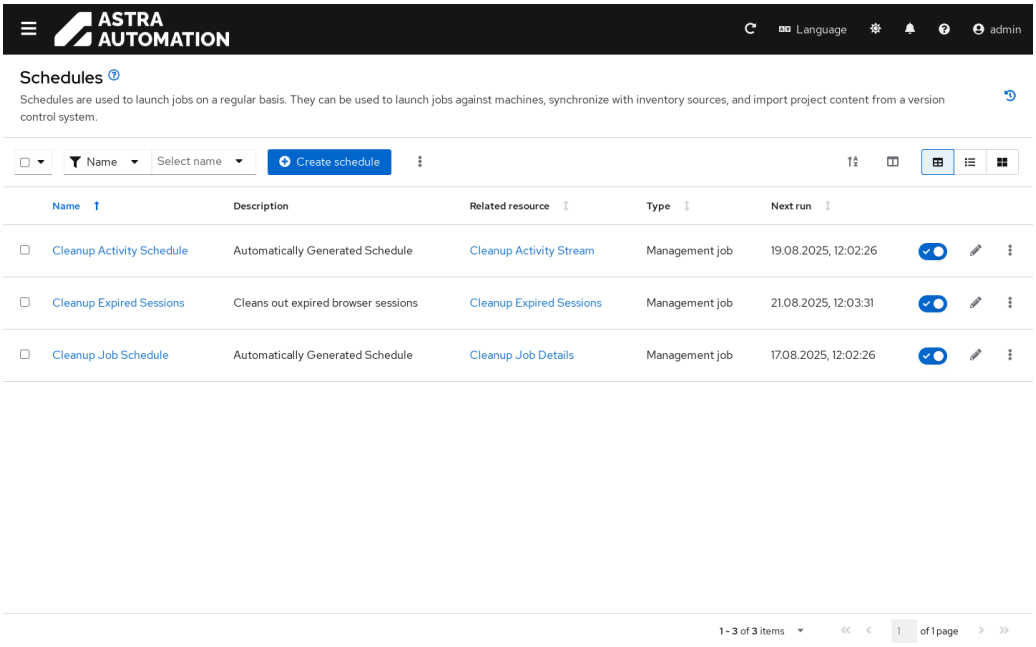


Таблица с расписаниями запуска заданий состоит из следующих столбцов:

- флаги для выбора нескольких расписаний;
- **Название** (Name) – ссылка на расписание, по которому выполняется запуск задания;
- **Описание** (Description) – описание расписания;
- **Связанный ресурс** (Related resource) – ссылка на ресурс, связанный с заданием;
- **Тип** (Type) – тип задания;
- **Следующий запуск** (Next Run) – дата и время ближайшего запуска задания;
- кнопки для быстрого вызова часто выполняемых действий:
  - переключатель включения и отключения расписания;

- редактирование;
- удаление.

Просмотр

Для получения подробных сведений о расписании нажмите на его название. Окно просмотра сведений о расписании состоит из одной вкладки **Подробности** (Details), в которой отображаются общие сведения о расписании: название, описание, дата первого запуска задания, дата следующего запуска задания и так далее.

Добавление нового расписания

Расписания можно создать:

- в окне **Расписания** (Schedules);
- в окне конкретного ресурса во вкладке **Расписания** (Schedules).

Результат будет одинаковым, различается только точка входа в процесс создания расписания.

Чтобы создать новое расписание в окне **Расписания** (Schedules), выполните следующие действия:

1. На панели инструментов нажмите кнопку *Создать расписание* (Create schedule).
2. В выпадающем списке **Тип ресурса** (Resource type) выберите тип ресурса, к которому относится расписание:
   - **Шаблон задания** (Job template) – в выпадающем списке **Шаблон задания** (Job template) выберите необходимый шаблон задания.
   - **Шаблон потока заданий** (Workflow job template) – в выпадающем списке **Шаблон потока заданий** (Workflow job template) выберите необходимый шаблон потока заданий.
   - **Источник инвентаризации** (Inventory source):
     - в выпадающем списке **инвентарный список** (Inventory) выберите необходимое описание инвентаря;
     - в выпадающем списке **Источник инвентаризации** (Inventory source) выберите необходимый источник.
   - **Синхронизация проекта** (Project sync) – в выпадающем списке **Проект** (Project) выберите необходимый проект.
   - **Шаблон служебных заданий** (Management job template) – в выпадающем списке **Шаблон служебных заданий** (Management job template) выберите необходимый шаблон служебных заданий.
3. Для типов ресурсов **Шаблон задания** (Job template), **Синхронизация проекта** (Project sync) и **Шаблон служебных заданий** (Management job template) заполните следующие поля:
   - **Название расписания** (Schedule name) – название.
   - (Опционально) **Описание** (Description) – описание.
   - **Дата/время начала** (Start date/time) – дата и время первого запуска.
   - **Часовой пояс** (Time zone) – часовой пояс. Дата и время первого запуска должны соответствовать этому часовому поясу.

     При создании расписания отображаются сведения, позволяющие просматривать настройки расписания и список запланированных событий в выбранном часовом поясе.

Важно

Значения времени переводятся в значения для временной зоны UTC и фиксируются для этой зоны. Поэтому если время задано для локальной временной зоны, то при переходе с летнего периода на зимний или обратно, локальное время запуска изменится в тех регионах, где предусмотрены такие переходы. Чтобы избежать нежелательных смещений, рекомендуется сразу указывать время для временной зоны UTC.

4. Нажмите кнопку *Далее* (Next) для перехода к следующему шагу создания расписания.
5. Заполните форму **Определить правила** (Define rules):
   - **Частота** (Frequency) – частота запуска задания. Задает единицу измерения для поля **Интервал** (Interval).
   - **Интервал** (Interval) – интервал повторения правила. Задает период повторения расписания, зависящий от единицы измерения, указанной в поле **Частота** (Frequency).
   - **Начало недели** (Week start) – день, с которого должна начинаться неделя.
   - **Будни** (Weekdays) – дни недели, в которые будет запускаться задание.
   - **Месяцы** (Months) – месяцы, в которые будет запускаться задание.
   - **Номер недели (недель) в году** (Annual week(s) number) – номера недель, в которые будет запускаться задание.
   - **Минута(ы) часа** (Minute(s) of hour) – минуты, в которые будет запускаться задание.
   - **Час дня** (Hour of day) – часы, в которые будет запускаться задание.
   - **День(дни) месяца** (Month day(s)) – номера дней месяца, в которые будет запускаться задание.
   - **День(дни) года** (Day(s) of year) – номера дней года, в которые будет запускаться задание.
   - **Вхождения** (Occurrences) – номер даты из набора, сформированного по правилу расписания. Если правило дает единственный результат, использование этого поля не имеет смысла.
   - **Тип завершения расписания** (Schedule ending type) – поле, определяющее, когда необходимо прекратить запускать задание.
6. Нажмите кнопку *Сохранить правило* (Save rule). Отобразится таблица сводки правил расписания, которая состоит из следующих столбцов:
   - кнопка переключения формата отображения времени;
   - **Следующие временные метки срабатываний** (Next occurrence timestamps) – набор дат запуска задания, соответствующих заданному правилу;
   - **RRule** – строка сгенерированного правила повторения в формате iCalendar RFC[167];
   - кнопки для быстрого вызова часто выполняемых действий:
     - редактирование;
     - удаление.
7. (Опционально) Нажмите кнопку *Добавить правило* (Add rule) для добавления нового правила.
8. Нажмите кнопку *Далее* (Next) для перехода к следующему шагу создания расписания.

[167] https://datatracker.ietf.org/doc/html/rfc5545

9. (Опционально) Нажмите кнопку *Создать исключение* (Create exception) для создания исключения из расписания. Исключения убирают даты из общего набора, полученного правилами. При создании исключений необходимо заполнить те же поля, что и для правил.
10. Нажмите кнопку *Далее* (Next) для перехода к следующему шагу создания расписания.
11. В окне **Обзор** (Review) проверьте данные создаваемого расписания и нажмите кнопку *Завершить* (Finish) для завершения создания расписания.

Редактирование основных сведений о расписании

Для изменения основных сведений о расписании выполните следующие действия:

1. В окне **Расписания** (Schedules) нажмите на название необходимого расписания.
2. Нажмите кнопку *Редактировать расписание* (Edit schedules).
3. Измените сведения о расписании.
4. Нажмите кнопку *Завершить* (Finish).

Удаление расписания

Для удаления расписания выполните следующие действия:

1. В окне **Расписания** (Schedules) включите флаги напротив удаляемых расписаний.
2. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите **Удалить расписание** (Delete schedules).
3. Подтвердите удаление.

Проекты

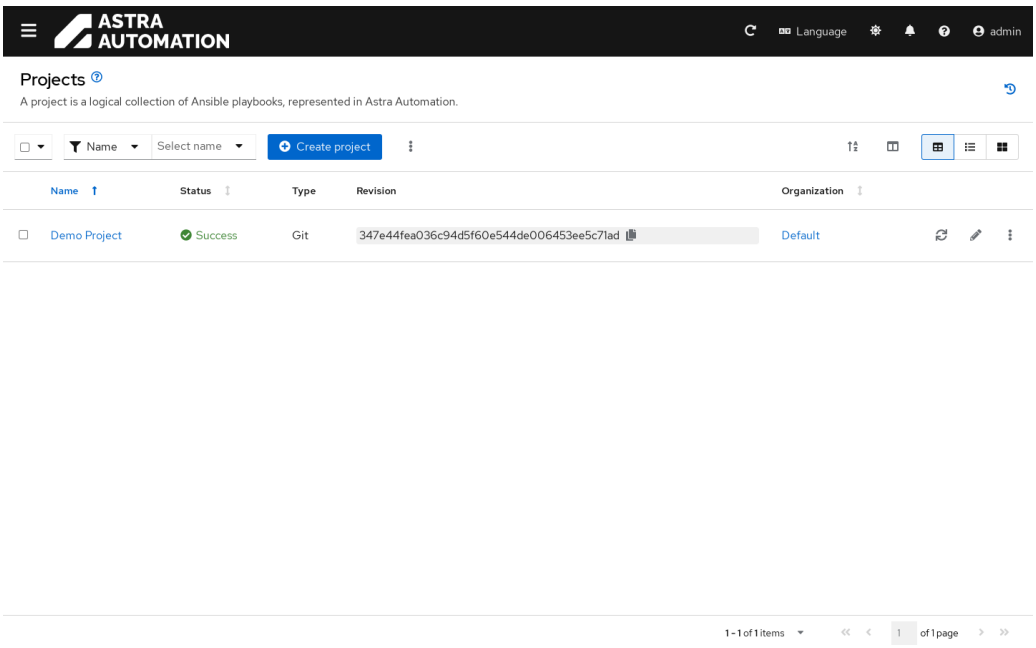
Окно **Проекты** (Projects) предназначено для управления *источниками кода* с помощью следующих операций:

- подключение репозиториев Git, SVN и других источников сценариев;
- синхронизация контента вручную или автоматически;
- отслеживание версий кода, используемых платформой;
- обеспечение контроля соответствия инфраструктурному репозиторию.

Для перехода к окну **Проекты** (Projects) выберите на панели навигации *Автоматизация процессов* ▸ *Проекты* (*Automation Execution* ▸ *Projects*).

Таблица проектов

Внешний вид окна **Проекты** (Projects) представлен на схеме:



На панели инструментов размещаются кнопка для выбора всех записей в таблице, поле поиска, кнопка *Создать проект* (Create project) для создания проекта, кнопка ⋮ для вызова действий над выделенными записями, поле сортировки записей, кнопка *Настроить столбцы* (Manage columns) для настройки отображения столбцов, кнопки выбора формата таблицы.

Таблица проектов состоит из следующих столбцов, включая те, отображение которые может быть включено с помощью кнопки *Настроить столбцы* (Manage columns):

- флаги для выбора нескольких проектов;
- **Название** (Name) – название проекта;
- **Описание** (Description) – описание проекта;
- **Статус** (Status) – текущее состояние проекта;
- **Тип** (Type) – тип источника кода проекта;
- **Пересмотр** (Revision) – версия кода проекта;
- **Организация** (Organization) – организация, которой принадлежит проект;
- **Дата создания** (Created) – дата и время создания проекта;
- **Последнее изменение** (Last modified) – дата и время последнего изменения проекта;
- **Последний раз использовался** (Last used) – дата и время последнего использования проекта;
- **Среда по умолчанию** (Default environment) – среда исполнения, назначенная проекту и используемая при запуске заданий из этого проекта, если иная среда исполнения не указана на уровне шаблона задания или рабочего процесса;
- кнопки для быстрого вызова часто выполняемых действий:
  - синхронизация кода проекта с источником;
  - переход в окно редактирования свойств проекта;
  - создание копии проекта;
  - удаление проекта.

Просмотр

Для получения подробных сведений о проекте нажмите на его название в таблице проектов.

Окно сведений о проекте состоит из следующих вкладок:

- **Подробности** (Details) – общие сведения о проекте;
- **Расписания** (Schedules) – таблица заданий по обновлению кода проекта из *источника*;
- **Шаблоны заданий** (Job Templates) – таблица *шаблонов заданий*, использующих код проекта;
- **Доступ пользователей** (User Access) – таблица пользователей и назначенных им *ролей*;
- **Командный доступ** (Team Access) – таблица команд и назначенных им *ролей*;
- **Уведомления** (Notifications) – таблица уведомлений.

> **Примечание**
>
> Эта вкладка недоступна для локальных проектов.

Создание

Для создания проекта выполните следующие действия:

1. В окне **Проекты** (Projects) нажмите кнопку *Создать проект* (Create project).
2. Заполните форму **Создать проект** (Create project):
   - **Название** (Name) – укажите название проекта.

     Особенности заполнения поля:

     - в рамках одной организации не допускается создание проектов с одним и тем же названием;
     - название проекта не может состоять из одних пробельных символов (пробелы, табуляции и так далее);
     - допускается использование символов кириллицы и специальных символов.
   - **Описание** (Description) – укажите описание проекта, например, его назначение или особенности использования.
   - **Организация** (Organization) – выберите организацию, которой принадлежит проект. В проекте должна быть хотя бы одна организация. После создания проекта вы сможете добавить другие организации.
   - **Среда исполнения** (Execution environment) – выберите *среду исполнения*, используемую для запуска заданий, созданных на основе проекта.

     > **Примечание**
     >
     > Это поле становится доступным для редактирования после выбора организации. Среда исполнения, выбранная на уровне проекта, имеет более высокий приоритет, чем среды исполнения, выбранные на уровне контроллера и организации.

   - **Тип системы управления исходным кодом** (Source control type) – выберите тип источника кода проекта.

     Особенности работы контроллера с различными типами источников описаны в разделе *Источники кода*.

- **Полномочия для проверки подписи контента** (Content signature validation credential) – если для проверки аутентичности кода проекта следует использовать GPG, выберите соответствующие полномочия типа «Публичный ключ GPG».

3. Заполните поля, соответствующие выбранному источнику кода.

Manual

- **Базовый путь проекта** (Project base path)

  Значение в этом поле недоступно для изменения. Оно указывает на каталог, в котором выполняется поиск проекта.

- **Каталог сценариев автоматизации** (Playbook Directory) – выберите локальный каталог с кодом проекта.

Git

- **URL системы управления исходным кодом** (Source control URL) – ссылка на репозиторий Subversion, в котором хранится код проекта.
- **Ветвь/Тег/Коммит системы управления исходным кодом** (Source control branch/tag/commit) – если требуется использовать версию кода, отличную от последнего коммита в основной ветке, укажите в этом поле название необходимой ветки, название тега или хеш коммита.
- **Специальная ссылка системы управления исходным кодом (refspec)** (Source control refspec) – если требуется использовать специфичную версию кода, укажите в этом поле правила получения необходимых ссылок из репозитория Git.
- **Полномочие на систему управления исходным кодом** (Source control credential) – если для доступа к репозиторию требуется аутентификация, выберите в этом поле соответствующие полномочия типа *Система управления версиями (Source Control)*.
- **Опции** (Options) – настройки, отвечающие за поведение запуска:
  - **Стереть** (Clean)

    Если эта настройка включена, перед обновлением будут сброшены все локальные изменения кода.

  - **Удалить** (Delete)

    Если эта настройка включена, при обновлении локальная копия кода проекта будет удалена, после чего загружена заново.

    В зависимости от размера репозитория это может значительно увеличить время, необходимое для завершения обновления.

  - **Отслеживать подмодули** (Track submodules)

    Если эта настройка включена, при обновлении проекта будут также обновлены все его субмодули.

  - **Обновлять ревизию при запуске** (Update revision on launch)

    Если эта настройка включена, перед запуском заданий на основе проекта его код будет обновляться автоматически.

  - **Разрешить переопределение ветви** (Allow branch override)

    Если эта настройка включена, в шаблоне задания можно выбрать ветку или версию кода, отличные от указанных в свойствах проекта.

Subversion

- **URL системы управления исходным кодом** (Source control URL) – ссылка на репозиторий Subversion, в котором хранится код проекта.
- **Номер редакции** (Revision number) – если требуется использовать ревизию кода, отличную от последнего коммита в основной ветке, укажите в этом поле номер необходимой ревизии.
- **Полномочие на систему управления исходным кодом** (Source control credential) – если для доступа к репозиторию требуется аутентификация, выберите в этом поле соответствующие полномочия типа *Система управления версиями (Source Control)*.
- **Опции** (Options) – настройки, отвечающие за поведение запуска:
  - **Стереть** (Clean)

    Если эта настройка включена, перед обновлением будут сброшены все локальные изменения кода.
  - **Удалить** (Delete)

    Если эта настройка включена, при обновлении локальная копия кода проекта будет удалена, после чего загружена заново.

    В зависимости от размера репозитория это может значительно увеличить время, необходимое для завершения обновления.
  - **Обновлять ревизию при запуске** (Update revision on launch)

    Если эта настройка включена, перед запуском заданий на основе проекта его код будет обновляться автоматически.
  - **Разрешить переопределение ветви** (Allow branch override)

    Если эта настройка включена, в шаблоне задания можно выбрать ветку или версию кода, отличные от указанных в свойствах проекта.

Red Hat Insights

- **Учетные данные Insights** (Insights credential) – выберите соответствующее полномочие для использования Red Hat Insights.
- **Опции** (Options) – настройки, отвечающие за поведение запуска:
  - **Стереть** (Clean)

    Если эта настройка включена, перед обновлением будут сброшены все локальные изменения кода.
  - **Удалить** (Delete)

    Если эта настройка включена, при обновлении локальная копия кода проекта будет удалена, после чего загружена заново.

    В зависимости от размера репозитория это может значительно увеличить время, необходимое для завершения обновления.
  - **Обновлять ревизию при запуске** (Update revision on launch)

    Если эта настройка включена, перед запуском заданий на основе проекта его код будет обновляться автоматически.

Remote Archive

- **URL системы управления исходным кодом** (Source control URL) – ссылка на архив, в котором хранится код проекта.

- **Полномочие на систему управления исходным кодом** (Source control credential) – если для доступа к репозиторию требуется аутентификация, выберите в этом поле соответствующие полномочия типа *Система управления версиями (Source Control)*.
- **Опции** (Options) – настройки, отвечающие за поведение запуска:
  - **Стереть** (Clean)

    Если эта настройка включена, перед обновлением будут сброшены все локальные изменения кода.
  - **Удалить** (Delete)

    Если эта настройка включена, при обновлении локальная копия кода проекта будет удалена, после чего загружена заново.

    В зависимости от размера репозитория это может значительно увеличить время, необходимое для завершения обновления.
  - **Обновлять ревизию при запуске** (Update revision on launch)

    Если эта настройка включена, перед запуском заданий на основе проекта его код будет обновляться автоматически.
  - **Разрешить переопределение ветви** (Allow branch override)

    Если эта настройка включена, в шаблоне задания можно выбрать ветку или версию кода, отличные от указанных в свойствах проекта.

4. Нажмите кнопку *Создать проект* (Create project).

После создания проекта его настройки можно изменять с помощью различных групп параметров, описанных в последующих секциях.

Настройка доступа

Во вкладках **Доступ пользователей** (User Access) и **Командный доступ** (Team Access) выводятся списки пользователей или команд и назначенных им *ролей*.

Назначение ролей отдельным пользователям

Чтобы настроить доступ к проекту для отдельных пользователей, выполните следующие действия:

1. Во вкладке **Доступ пользователей** (User Access) нажмите кнопку *Добавить роль* (Add roles).
2. Выберите пользователей, которым хотите назначить роли, и нажмите кнопку *Далее* (Next).
3. Выберите роли, которые хотите назначить выбранным на предыдущем шаге пользователям.
4. Нажмите кнопку *Завершить* (Finish).

Назначение командных ролей

Чтобы настроить доступ к проекту для всех участников одной или нескольких команд, выполните следующие действия:

1. Во вкладке **Командный доступ** (Team Access) нажмите кнопку *Добавить роль* (Add roles).
2. Выберите команды, участникам которых хотите назначить роли, и нажмите кнопку *Далее* (Next).
3. Выберите роли, которые хотите назначить выбранным на предыдущем шаге командам.
4. Нажмите кнопку *Завершить* (Finish).

Отзыв ролей

Чтобы отозвать пользовательские или командные роли, выполните следующие действия:

1. Во вкладке **Доступ пользователей** (User Access) или **Командный доступ** (Team Access) установите флаги в строках с пользователями или командами, соответственно, у которых следует отозвать роли.
2. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите пункт *Удалить роли* (Remove roles).
3. Подтвердите отзыв роли.

Примечание

Роли «Системный администратор» (System Administrator) и «Системный аудитор» (System Auditor) нельзя отозвать через настройку доступа на уровне проекта, однако, можно *изменить тип пользователя*.

Удаление

Удаление проектов имеет следующие особенности:

- при удалении проекта связанные с ним шаблоны заданий и инвентарные списки не удаляются и могут быть связаны с новым проектом;
- при удалении проекта, использующего в качестве источника локальный каталог, файлы проекта не удаляются с диска.

Чтобы удалить проект, выполните следующие действия:

1. В таблице установите флаги в строках с проектами, которые необходимо удалить.
2. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите пункт *Удалить проекты* (Delete projects).
3. Подтвердите удаление.

Обновление

Если проект использует внешний источник кода, для поддержания локальной копии в актуальном состоянии необходимо обновлять проект. Automation Controller позволяет обновлять код проектов вручную или автоматически, при этом автоматическое обновление может выполняться при запуске связанных с проектом заданий или по расписанию.

Чтобы обновить проект вручную, во вкладке **Подробности** (Details) нажмите кнопку *Синхронизировать проект* (Sync project).

Совет

Для быстрого запуска обновления проекта можно использовать кнопку *Синхронизировать проект* (Sync project), доступную в строке проекта, который необходимо обновить.

Чтобы проект автоматически обновлялся перед запуском связанных с ним заданий, в настройках источника включите соответствующие флаги в блоке **Опции** (Options).

Чтобы проект обновлялся по расписанию, настройте его следуя инструкциям в документе *Расписания*.

Режимы просмотра

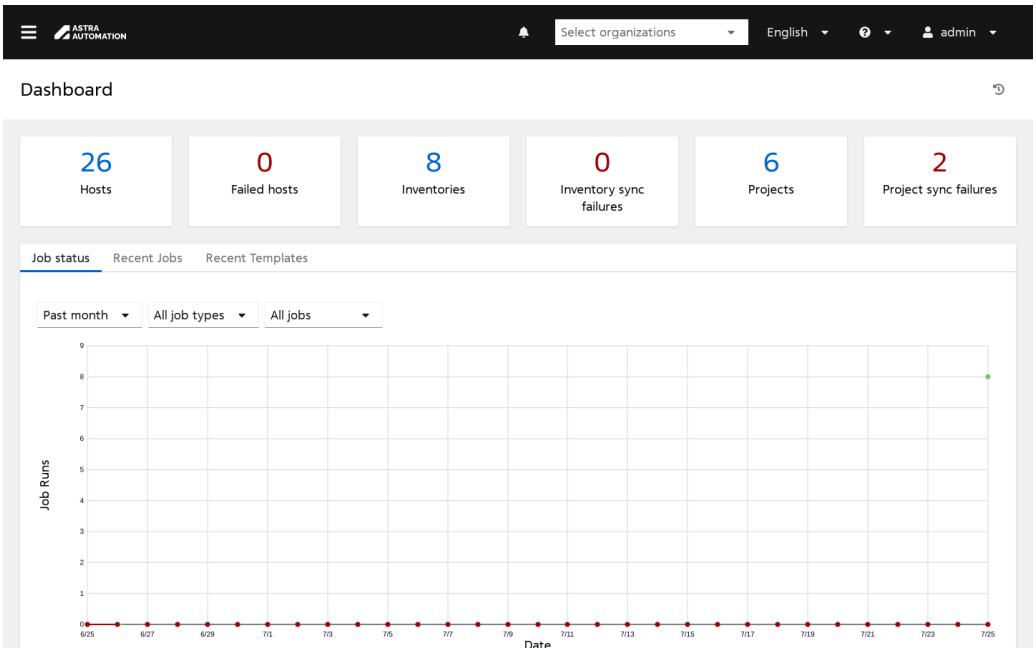
В этом разделе описывается просмотр сведений о состоянии заданий.

Информационная панель

Окно **Информационная панель** (Dashboard) используется для вывода сводной информации об управляемых узлах, статусах запущенных заданий, заданий синхронизации инвентаря и заданий синхронизации проектов. Для перехода к нему выберите на панели навигации *Режимы просмотра* ▸ *Информационная панель* (*Views* ▸ *Dashboard*).

Описание окна

Внешний вид окна **Информационная панель** (Dashboard) представлен на схеме:



Окно **Информационная панель** (Dashboard) состоит из панелей и группы вкладок.

Панели, слева направо:

- **Управляемые узлы** (Hosts) – общее количество управляемых узлов.

  При расчете значения учитываются в том числе управляемые узлы, находящиеся в *статусе* «Выкл» (Off).

- **Узлы с ошибками** (Failed Hosts) – количество узлов, для которых статус выполнения последнего связанного задания – «Отказ» (Failed).
- **Инвентарь** (Inventories) – количество *инвентарных списков*.
- **Ошибки синхронизации инвентаря** (Inventory sync failures) – количество отказов при синхронизации инвентарных списков.
- **Проекты** (Projects) – количество *проектов*.
- **Проекты с ошибками синхронизации** (Project sync failures) – количество отказов при синхронизации проектов.

Нажатие на любую панель приводит к переходу в соответствующее окно.

Во вкладках под панелями отображается следующая информация:

- **Статус задания** (Job Status) – график запуска заданий и статус их выполнения за последний месяц.

  График отображает количество запущенных заданий за каждые сутки. Сведения о количестве успешно выполненных заданий отображаются зеленым цветом, а о количестве заданий с ошибками или отказами – красным.

  Поддерживаются следующие настройки отображения:

  - Размер периода:

        * **Последний месяц** (Past month);
        * **Последние две недели** (Past two weeks);
        * **Последняя неделя** (Past week);
        * **Последние 24 часа** (Past 24 hours).
    - Тип заданий:
        * **Все типы заданий** (All job types);
        * **Синхронизация инвентаря** (Inventory sync);
        * **Обновлением SCM** (SCM update);
        * **Запуск playbook** (Playbook run).
    - Статус заданий – может принимать следующие значения:
        * **Все задания** (All jobs);
        * **Успешные задания** (Successful jobs);
        * **Задания с ошибками** (Failed jobs).

  При изменении значения любого из параметров график обновляется автоматически.
- **Последние задания** (Recent Jobs) – информация о последних запущенных заданиях.

  Таблица состоит из следующих колонок:
    - Переключатель подробности вывода основных сведений о задании.

      Набор отображаемых полей зависит от типа задания.
    - Флаги для выбора нескольких записей.
    - **Название** (Name) – ссылка для перехода в окно просмотра журнала выполнения задания или графа потока заданий.
    - **Статус** (Status)
        * **Успех** (Successful) – не было ошибок;
        * **Ошибка** (Error) – возникли ошибки, которые были успешно обработаны;
        * **Сбой** (Failed) – возникли необрабатываемые ошибки, выполнение задания прервано;
        * **Отменено** (Canceled) – задание отменено пользователем;
        * **Ожидание** (Pending) – задание находится в очереди;
        * **Выполнение** (Running) – задание выполняется.
    - **Время начала** (Start Time) – время запуска задания.
    - **Время завершения** (Finish Time) – время завершения, отмены или прерывания задания.
    - **Действия** (Actions) – кнопки для быстрого вызова часто выполняемых действий:
        * отмена задания;
        * перезапуск задания.
- **Последние шаблоны** (Recent Templates) – информация о последних использованных шаблонах заданий и шаблонах потоков заданий.

  Таблица состоит из следующих колонок:
    - Переключатель подробности вывода основных сведений о шаблоне.

      Набор отображаемых полей зависит от типа шаблона.
    - Флаги для выбора нескольких записей.

- **Название** (Name) – ссылка для перехода в окно просмотра подробных сведений о шаблоне задания или шаблоне потока заданий.
- **Тип** (Type) – тип шаблона.
- **Организация** (Organization) – ссылка для перехода в окно просмотра подробных сведений об организации, которой принадлежит шаблон.
- **Последний запуск** (Last Ran) – дата и время последнего запуска заданий на основе шаблона заданий или потока заданий.
- **Действия** (Actions) – кнопки для быстрого вызова часто выполняемых действий:
  * переход в окно редактирования графа шаблона потока заданий (эта кнопка доступна только для шаблонов потоков заданий);
  * запуск задания на основе шаблона задания или потока заданий на основе шаблона потока заданий;
  * переход в окно редактирования свойств шаблона;
  * копирование шаблона.

Метрики узлов

Окно **Метрики узлов** (Host Metrics) используется для вывода статистической информации о запуске заданий автоматизации на управляемых узлах. Для перехода к нему выберите на панели навигации *Режимы просмотра* ▸ *Метрики узлов* (*Views* ▸ *Host Metrics*).

Таблица метрик узлов

Внешний вид окна **Метрики узлов** (Host Metrics) представлен на схеме:

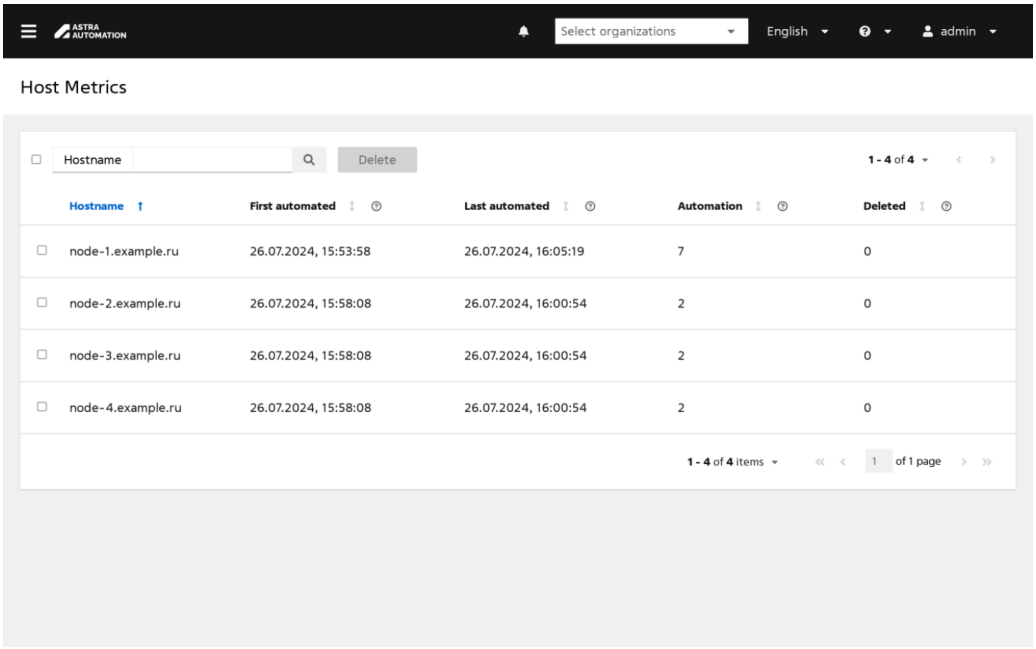


Таблица с информацией о выполнении заданий автоматизации на управляемых узлах состоит из следующих колонок:

- флаги для выбора нескольких записей;
- **Наименование узла** (Hostname) – название управляемого узла;
- **Первый автоматизированный** (First automated) – дата и время первого запуска задания автоматизации;
- **Последняя автоматизация** (Last automated) – дата и время последнего запуска задания автоматизации;

- **Автоматизация** (Automation) – общее количество выполненных заданий автоматизации;
- **Инвентарь** (Inventories) – количество инвентарных списков, в которые входит управляемый узел;
- **Удалено** (Deleted) – количество раз, которые управляемый узел был удален из списка управляемых узлов.

Поддерживается фильтрация и сортировка записей по различным параметрам.

Удаление записей

Чтобы удалить сводную информацию об определенных управляемых узлах, выполните следующие действия:

1. В таблице со статистической информацией включите флаги в строках с управляемыми узлами, сведения о которых следует удалить.
2. Нажмите кнопку *Удалить* (Delete).
3. Подтвердите удаление записей.

Расход ресурсов подписки

Важно

Эта часть документации находится в стадии разработки.

Инфраструктура

Раздел Инфраструктура (Infrastructure) предназначен для управления инфраструктурой плоскостей управления и исполнения. В нем доступны следующие функции:

- просмотр топологии (Topology);
- подготовка описания инвентаря (Inventories);
- подключение управляемых узлов (Hosts);
- управление исполняющими узами (Instances);
- настройка групп исполняющих узлов (Instance Groups);
- управление средой исполнения (Execution Environments);
- создание и настройка полномочий для получения доступа к внешним ресурсами или управления ими (Credentials);
- создание собственных типов полномочий (Credential Types).

Представление топологии

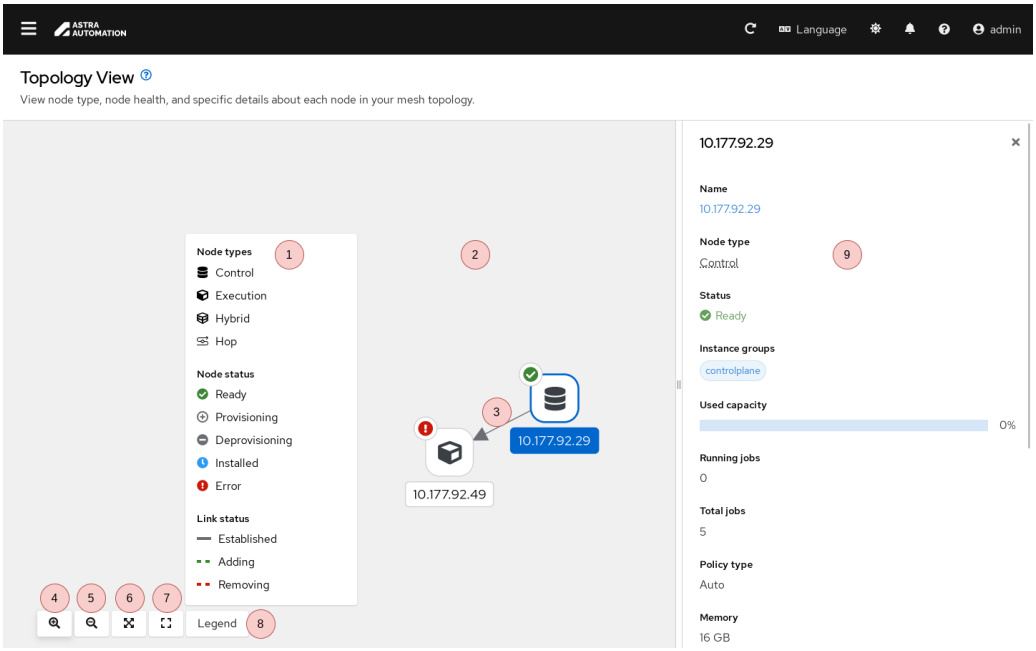
Окно **Представление топологии** (Topology View) предоставляет:

- графическое отображение взаимодействий контроллеров, узлов исполнения и механизмов распределения нагрузки;
- просмотр состояния компонентов кластера и ролей выполнения;
- возможность диагностики взаимосвязей и точек отказа в архитектуре.

Для перехода к окну **Представление топологии** (Topology View) выберите на панели навигации *Автоматизация процессов* ▸ *Инфраструктура* ▸ *Представление топологии* (*Automation Execution* ▸ *Infrastructure* ▸ *Topology View*).

Описание окна

Внешний вид окна **Представление топологии** (Topology View) представлен на схеме:



Окно **Представление топологии** (Topology View) состоит из следующих элементов:

1. Легенда – пояснение принятых элементов схемы.
2. Пространство для представления топологии.
3. Связи между узлами.
4. Кнопка увеличения масштаба.
5. Кнопки уменьшения масштаба.
6. Кнопка подстраивания схемы под размеры окна.
7. Кнопка сброса размера схемы в исходное значение.
8. Переключатель видимости панели легенды.
9. Панель сведений о выбранном узле. Содержимое панели зависит от типа выбранного узла.

Просмотр

Для просмотра сведений об узле выделите его.

На схеме выше представлена базовая топология платформы, состоящая из управляющего узла `10.177.92.29` и исполняющего узла `10.177.92.49`.

Подробности о типах узлов см. в документе *Особенности архитектуры*.

Инвентарные списки

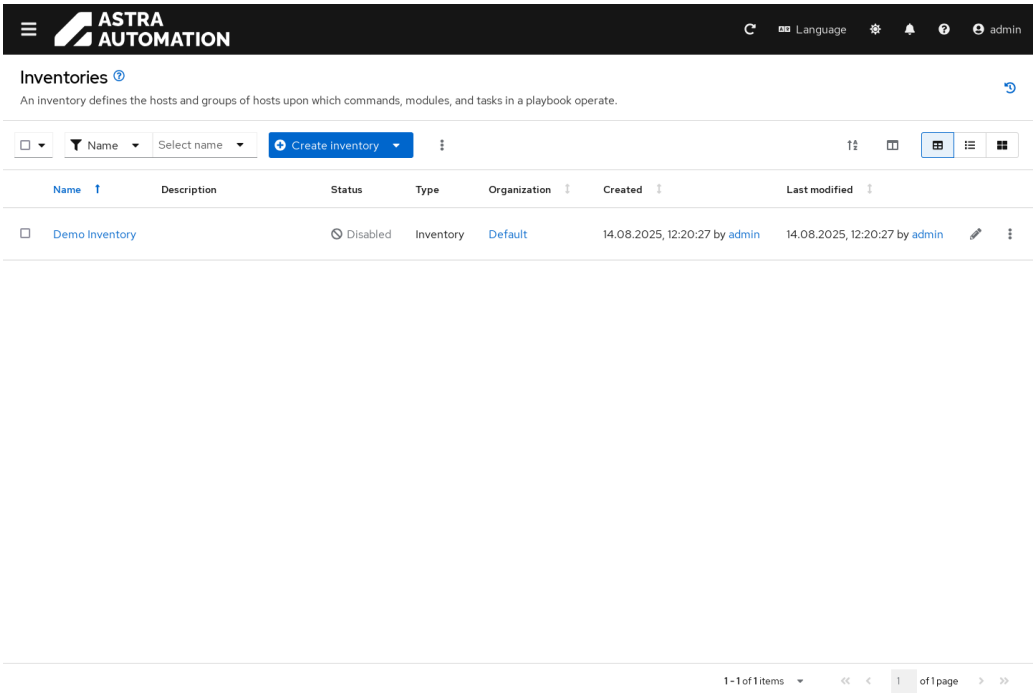
Окно **Инвентарные списки** (Inventories) предоставляет средства для управления группами узлов:

- создание и редактирование *инвентарных списков* и групп узлов;
- подключение динамических источников данных;
- управление переменными окружения и учетными данными для групп и узлов;
- выполнение синхронизации и проверка статусов ресурсов.

Для перехода к окну **Инвентарные списки** (Inventories) выберите на панели навигации *Автоматизация процессов* ▸ *Инфраструктура* ▸ *Инвентарные списки* (*Automation Execution* ▸ *Infrastructure* ▸ *Inventories*).

Таблица инвентарных списков

Внешний вид окна **Инвентарные списки** (Inventories) представлен на схеме:



На панели инструментов размещаются поле поиска, кнопка *Создать инвентарный список* (Create inventory) для создания инвентарного списка различных типов и кнопка ⋮ для вызова действий над выделенными записями.

Таблица инвентарных списков состоит из следующих столбцов:

- Флаги для выбора нескольких записей.
- **Название** (Name).
- **Описание** (Description).
- **Статус** (Status).

  Если для формирования инвентаря используется хотя бы один внешний источник, в этом поле выводится статус синхронизации. В противном случае выводится статус **Неактивный** (Disabled).

  Для умных и сборных инвентарных списков понятие синхронизации не используется.
- **Тип** (Type) – *тип инвентарного списка*.
- **Организация** (Organization) – организация-владелец.
- **Дата создания** (Created) – дата и время создания инвентарного списка.
- **Последнее изменение** (Last modified) – дата и время последнего изменения инвентарного списка.
- Кнопка перехода в окно изменения свойств инвентарного списка.
- Кнопка ⋮ – вызов меню дополнительных действий.

В этом документе приводятся инструкции, общие для инвентарных списков всех типов. Инструкции по управлению инвентарными списками различных типов описаны в соответствующих документах.

Просмотр

Для получения подробных сведений об инвентарном списке нажмите на ссылку с его названием в таблице инвентарных списков. Окно сведений об инвентарном списке состоит из следующих вкладок:

- **Подробности** (Details) – общие сведения об инвентарном списке.
- **Доступ пользователей** (User Access) – таблица пользователей и назначенных им ролей.
- **Командный доступ** (Team Access) – таблица команд и назначенных им ролей.
- **Группы** (Groups) – таблица групп узлов.

  Примечание

  Эта вкладка недоступна при просмотре сведений об умных инвентарных списках.

- **Узлы** (Hosts) – список управляемых узлов, входящих в инвентарный список.
- **Источники** (Sources) – список источников сведений об управляемых узлах.

  Примечание

  Эта вкладка доступна только при просмотре сведений об обычных инвентарных списках.

- **Задания** (Jobs) – список заданий, связанных с инвентарным списком.
- **Шаблоны заданий** (Job Templates) – список шаблонов заданий, связанных с инвентарным списком.

Настройка доступа

Во вкладке **Доступ пользователей** (User Access) или **Командный доступ** (Team Access) выводится таблица пользователей или команда, соответственно, и назначенных им *ролей*.

Назначение ролей отдельным пользователям

Чтобы настроить доступ к инвентарному списку для отдельных пользователей, выполните следующие действия:

1. Во вкладке **Доступ пользователей** (User Access) нажмите кнопку *Добавить роль* (Add roles).
2. Выберите пользователей, которым хотите назначить роли, и нажмите кнопку *Далее* (Next).
3. Выберите роли, которые хотите назначить выбранным на предыдущем шаге пользователям и нажмите кнопку *Далее* (Next).
4. Убедитесь в корректности настроенных доступов и нажмите кнопку *Завершить* (Finish).

Назначение командных ролей

Чтобы настроить доступ к инвентарному списку для всех участников одной или нескольких команд, выполните следующие действия:

1. Во вкладке **Командный доступ** (Team Access) нажмите кнопку *Добавить роль* (Add roles).
2. Выберите команды, участникам которых хотите назначить роли, и нажмите кнопку *Далее* (Next).

3. Выберите роли, которые хотите назначить выбранным на предыдущем шаге командам и нажмите кнопку *Далее* (Next).
4. Убедитесь в корректности настроенных доступов и нажмите кнопку *Завершить* (Finish).

Отзыв ролей

Чтобы отозвать пользовательские или командные роли, выполните следующие действия:

1. Во вкладке **Доступ пользователей** (User Access) или **Командный доступ** (Team Access) установите флаги в строках с пользователями или командами, соответственно, у которых следует отозвать роли.
2. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите пункт *Удалить роли* (Remove roles).
3. Подтвердите отзыв роли.

Примечание

Роли «Системный администратор» (System Administrator) и «Системный аудитор» (System Auditor) нельзя отозвать через настройку доступа на уровне инвентарного списка, однако, можно *изменить тип пользователя*.

Выполнение специальных команд (Ad Hoc)

Для выполнения специальных команд выполните следующие действия:

1. Во вкладке **Узлы** (Hosts) выберите управляемые узлы или во вкладке **Группы** (Groups) выберите группы управляемых узлов, на которых хотите выполнить специальную команду.

   Отфильтровать узлы по названию можно позже.
2. Нажмите кнопку *Выполнить команду* (Run Command).
3. В открывшемся окне заполните форму во вкладке **Подробности** (Details):
   - **Модуль** (Module) – выберите модуль Ansible, используемый для выполнения команды.

     Поддерживаются следующие модули (в алфавитном порядке):
     - apt[168];
     - apt_key[169];
     - apt_repository[170];
     - apt_rpm[171];
     - command[172];
     - group[173];
     - mount[174];
     - ping[175];

[168] https://docs.ansible.com/ansible/latest/collections/ansible/builtin/apt_module.html
[169] https://docs.ansible.com/ansible/latest/collections/ansible/builtin/apt_key_module.html
[170] https://docs.ansible.com/ansible/latest/collections/ansible/builtin/apt_repository_module.html
[171] https://docs.ansible.com/ansible/latest/collections/community/general/apt_rpm_module.html
[172] https://docs.ansible.com/ansible/latest/collections/ansible/builtin/command_module.html
[173] https://docs.ansible.com/ansible/latest/collections/ansible/builtin/group_module.html
[174] https://docs.ansible.com/ansible/latest/collections/ansible/posix/mount_module.html
[175] https://docs.ansible.com/ansible/latest/collections/ansible/builtin/ping_module.html

  - selinux[176];
  - service[177];
  - setup[178];
  - shell[179];
  - user[180];
  - win_group[181];
  - win_ping[182];
  - win_service[183];
  - win_updates[184];
  - win_user[185];
  - yum.

- **Аргументы** (Arguments) – укажите аргументы, с которыми должна быть выполнена команда из выбранного модуля.

  Примечание

  Для некоторых модулей заполнение этого поля обязательно.

- **Уровень подробности** (Verbosity) – выберите уровень детализации журнала выполнения задания.

  Поддерживаются следующие значения:

  - `0 (Нормальный) (0 Normal)`;
  - `1 (Подробный) (1 Verbose)`;
  - `2 (Более подробный) (2 More Verbose)`;
  - `3 (Отладка) (3 Debug)`;
  - `4 (Отладка подключения) (4 Connection Debug)`;
  - `5 (Отладка WinRM) (5 WinRM Debug)`.

  Значение по умолчанию: `0 (Нормальный) (0 Normal)`.

- **Предел** (Limit) – укажите шаблон названий управляемых узлов, на которых следует выполнить команду. В качестве разделителя названий узлов используется запятая.

  Если значение не указано, либо равно `all` или `*`, команда выполняется на всех узлах, входящих в инвентарный список.

  Подробности о шаблонах названий узлов см. в документации Ansible[186].

[176] https://docs.ansible.com/ansible/latest/collections/ansible/posix/selinux_module.html
[177] https://docs.ansible.com/ansible/latest/collections/ansible/builtin/service_module.html
[178] https://docs.ansible.com/ansible/latest/collections/ansible/builtin/setup_module.html
[179] https://docs.ansible.com/ansible/latest/collections/ansible/builtin/shell_module.html
[180] https://docs.ansible.com/ansible/latest/collections/ansible/builtin/user_module.html
[181] https://docs.ansible.com/ansible/latest/collections/ansible/windows/win_group_module.html
[182] https://docs.ansible.com/ansible/latest/collections/ansible/windows/win_ping_module.html
[183] https://docs.ansible.com/ansible/latest/collections/ansible/windows/win_service_module.html
[184] https://docs.ansible.com/ansible/latest/collections/ansible/windows/win_updates_module.html
[185] https://docs.ansible.com/ansible/latest/collections/ansible/windows/win_user_module.html
[186] https://docs.ansible.com/ansible/latest/inventory_guide/intro_patterns.html

- **Ответвления** (Forks) – укажите количество параллельных процессов, используемых для выполнения задания.

  Если для этого параметра задано значение меньше 1, используется значение по умолчанию, равное 5.

  Максимальное значение этого параметра задается в значении *системной настройки* **Максимальное количество ответвлений задания** (Maximum number of forks per job).

- **Показать изменения** (Show changes).

  Если эта настройка включена, в журнал выполнения задания выводятся изменения, сделанные на управляемых узлах при выполнении команды (если это поддерживается выбранным модулем или командой).

  Эквивалентно выполнению команды `ansible-playbook` с параметром `--diff`.

- **Повышение привилегий** (Privilege escalation).

  Если эта настройка включена, задание запускается с повышенными привилегиями.

  Эквивалентно запуску команды `ansible-playbook` с параметром `--become`.

- **Дополнительные переменные** (Extra variables) – укажите дополнительные переменные, значения которых следует использовать при выполнении задания.

  Подробности о порядке присвоения значений переменным см. в документе *Переменные*.

4. Нажмите кнопку *Далее* (Next).
5. Выберите *среду исполнения*, в которой следует запустить задание.
6. Нажмите кнопку *Далее* (Next).
7. Выберите полномочия, используемые для доступа к управляемым узлам, и нажмите кнопку *Далее* (Next).
8. Проверьте корректность введенных данных и нажмите кнопку *Завершить* (Run).

Удаление

Предупреждение

Особенности удаления инвентарных списков различного типа описаны в секции *Особенности удаления*.

Чтобы удалить инвентарные списки, выполните следующие действия:

1. В таблице инвентарных списков установите флаги в строках с инвентарными списками, которые следует удалить.
2. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите пункт *Удалить инвентарные списки* (Delete inventories).
3. Подтвердите удаление.

Обычный инвентарный список

В этой секции приводятся инструкции по работе с *обычными инвентарными списками* через веб-интерфейс.

Создание

Для создания *обычного инвентарного списка* выполните следующие действия:

1. В окне **Инвентарные списки** (Inventories) нажмите кнопку *Создать инвентарный список* (Create inventory) и в открывшемся меню выберите пункт **Создать инвентарный список** (Create inventory).
2. Заполните форму **Создать инвентарный список** (Create inventory):
   - **Название** (Name) – укажите название инвентарного списка.

     Особенности заполнения поля:

     - Название инвентарного списка должно быть уникальным на уровне организации.
     - Название инвентарного списка не может состоять из одних пробельных символов (пробелы, табуляции и так далее).
     - Допускается использование символов кириллицы и специальных символов.
   - **Описание** (Description) – укажите дополнительные данные об инвентарном списке, например, цель его создания или особенности использования.
   - **Организация** (Organization) – выберите организацию, которой принадлежит инвентарный список.
   - **Группы исполняющих узлов** (Instance groups) – группы узлов, используемые для запуска заданий, связанных с инвентарным списком.

     Подробности о порядке выбора узлов для запуска заданий см. в секции *Выбор группы узлов для запуска заданий*.
   - **Метки** (Labels).

     С помощью меток можно группировать задания, связанные с инвентарным списком. Также метки удобно использовать для поиска записей журнала выполнения заданий или их логической группировки.
   - **Переменные** (Variables) – укажите переменные, действующие на уровне инвентарного списка.

     Подробности о порядке присвоения значений переменным см. в документе *Переменные*.
   - **Отменить резерв группы исполняющих узлов** (Prevent instance group fallback).

     Если эта настройка включена, при запуске заданий, связанных с инвентарным списком, выбор предпочтительных групп узлов исполнения работает следующим образом:

     - Нельзя добавить группы, указанные как предпочтительные в свойствах организации.
     - Если в инвентарном списке список предпочтительных групп узлов пуст, используются глобальные настройки предпочтительных групп узлов.
3. Нажмите кнопку *Создать инвентарный список* (Create inventory).

После создания инвентарного списка его настройки можно изменять с помощью различных групп параметров, описанных в последующих секциях.

Управление группами узлов

Обычный инвентарный список имеет полную поддержку групп управляемых узлов и позволяет выполнять с ними следующие операции:

- создание;
- изменение;
- удаление;

- заполнение управляемыми узлами;
- построение иерархии.

Примечание

Группы узлов из источников создаются автоматически при синхронизации.

Для перехода к списку групп узлов и управления ими выполните следующие действия:

1. В таблице инвентарных списков нажмите на ссылку с названием обычного необходимого инвентарного списка.
2. Выберите вкладку **Группы** (Groups).

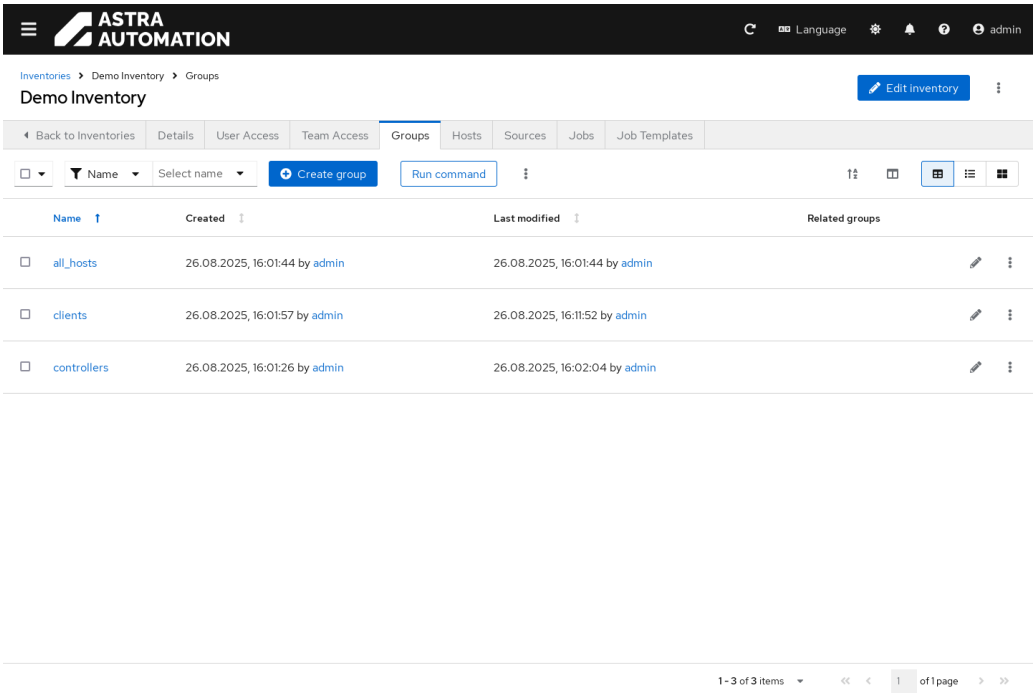


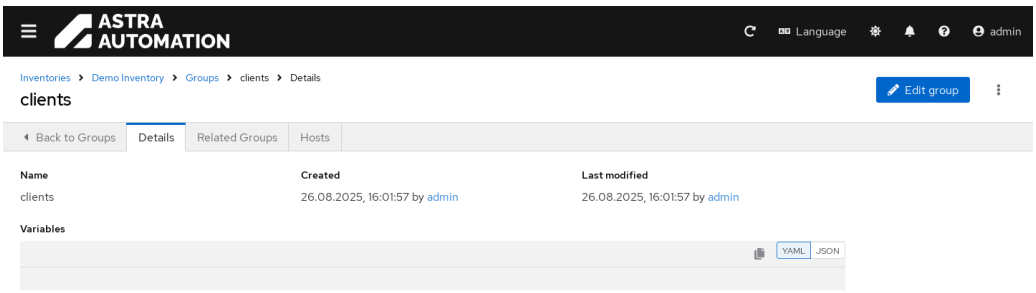
Таблица групп узлов обычного инвентарного списка состоит из следующих колонок:

- Флаги для выбора нескольких записей.
- **Название** (Name).
- **Дата создания** (Created) – дата и время создания группы узлов, а также ссылка на профиль выполнившего его пользователя.
- **Последнее изменение** (Last modified) – дата и время последнего изменения группы узлов, а также ссылка на профиль выполнившего его пользователя.
- **Связанные группы** (Related groups) – таблица дочерних групп узлов.

Просмотр информации о группах узлов

Для просмотра информации о группе управляемых узлов выполните следующие действия:

1. В таблице инвентарных списков нажмите на ссылку с названием необходимого обычного инвентарного списка.
2. Выберите вкладку **Группы** (Groups).
3. Нажмите на ссылку с названием необходимой группы узлов.



Окно просмотра информации о группе управляемых узлов состоит из следующих вкладок:

- **Подробности** (Details) – общие сведения о группе управляемых узлов:
  - **Название** (Name) – название группы.
  - **Дата создания** (Created) – дата и время создания группы узлов, а также ссылка на профиль выполнившего его пользователя.
  - **Последнее изменение** (Last modified) – дата и время последнего изменения группы узлов, а также ссылка на профиль выполнившего его пользователя.
  - **Переменные** (Variables) – переменные группы.

    Подробности о порядке присвоения значений переменным см. в документе *Переменные*.
- **Связанные группы** (Related groups) – таблица дочерних групп узлов.

  Таблица дочерних групп узлов состоит из следующих колонок:
  - флаги для выбора нескольких записей;
  - **Название** (Name) – ссылка на дочернюю группу;
  - **Дата создания** (Created) – дата и время создания группы узлов, а также ссылка на профиль выполнившего его пользователя.
  - **Последнее изменение** (Last modified) – дата и время последнего изменения группы узлов, а также ссылка на профиль выполнившего его пользователя.
  - Кнопка перехода в окно изменения свойств дочерней группы.
  - Кнопка ⋮ – вызов меню дополнительных действий.
- **Узлы** (Hosts) – таблица управляемых узлов, входящих в группу.

  Таблица управляемых узлов группы состоит из следующих колонок:
  - Флаги для выбора нескольких записей.
  - **Название** (Name) – ссылки для перехода в окно просмотра свойств управляемых узлов.
  - **Описание** (Description).
  - **Активность** (Activity).

Для управляемых узлов, сведения о которых добавлены из внешних источников, в этом поле выводится значение `imported`.

- **Дата создания** (Created) – дата и время создания узла, а также ссылка на профиль выполнившего его пользователя.
- **Последнее изменение** (Last modified) – дата и время последнего изменения узла, а также ссылка на профиль выполнившего его пользователя.
- Переключатель *статуса управляемого узла*.
- Кнопка перехода в окно изменения свойств узла.

Создание группы узлов

Чтобы создать группу управляемых узлов, выполните следующие действия:

1. Во вкладке **Группы** (Groups) нажмите кнопку *Создать группу* (Create group).
2. Заполните форму **Создать группу** (Create group):
   - **Название** (Name) – укажите название группы управляемых узлов.

     Предупреждение

     Название группы управляемых узлов должно быть уникально в пределах инвентарного списка.

   - **Описание** (Description) – укажите описание группы управляемых узлов.
   - **Переменные** (Variables) – укажите значения переменных, действующих на уровне группы управляемых узлов.

     Подробности о порядке присвоения значений переменным см. в документе *Переменные*.
3. Нажмите кнопку *Создать группу* (Create group).

Создание иерархии групп узлов

Чтобы создать иерархические отношения между группами управляемых узлов, выполните следующие действия:

1. Во вкладке **Группы** (Groups) нажмите на ссылку с названием группы узлов, которая должна быть родительской. Далее эта группа называется родительской группой.
2. В окне просмотра информации о родительской группе выберите вкладку **Связанные группы** (Related Groups).
3. Нажмите кнопку *Добавить группу* (Add group) и в открывшемся меню выберите соответствующий пункт:
   - **Добавить существующую группу** (Add existing group) – чтобы сделать дочерними уже существующие группы узлов;
   - **Создать группу** (Create group) – чтобы создать новую дочернюю группу узлов.
4. Если вы выбрали добавление существующих групп узлов, в открывшемся окне **Выберите группы** (Select groups) включите флаги в строках с названиями необходимых групп узлов и нажмите кнопку *Добавить группу* (Add group).
5. Если вы выбрали создание новой дочерней группы узлов, заполните форму **Создать группу** (Create new group) и нажмите кнопку *Создать группу* (Create group).

Удаление группы узлов

Чтобы удалить группы узлов, выполните следующие действия:

1. В таблице инвентарных списков нажмите на ссылку с названием обычного инвентарного списка, из которого необходимо удалить группы узлов.
2. Выберите вкладку **Группы** (Groups).
3. В таблице групп управляемых узлов установите флаги в строках с удаляемыми группами.
4. На панели инструментов нажмите кнопку ⋮ и в в открывшемся меню выберите пункт *Удалить группы* (Delete groups).
5. В диалоговом окне **Удалить группу** (Delete group) выполните следующие действия:
    1. Выберите действие, которое следует выполнить с дочерними группами и входящими в удаляемые группы узлами:
        - **Удалить все группы и узлы** (Delete all groups and hosts) – удалить;
        - **Добавить дочерние узлы и группы** (Promote child groups and hosts) – сохранить.
    2. Нажмите кнопку *Удалить* (Delete).

Управление источниками

Обычный инвентарный список поддерживает получение сведений об управляемых узлах из одного или нескольких внешних источников.

Примечание

Чтобы в контроллере появились сведения об управляемых узлах из источника, после его добавления необходимо запустить синхронизацию.

Для перехода к списку источников и управления ими выполните следующие действия:

1. В таблице инвентарных списков нажмите на ссылку с названием необходимого обычного инвентарного списка.
2. Выберите вкладку **Источники** (Sources).

Добавление источника

Чтобы добавить в обычный инвентарный список источник сведений об управляемых узлах, выполните следующие действия:

1. Во вкладке **Источники** (Sources) нажмите кнопку *Создать источник* (Create source).
2. Заполните форму **Создать источник** (Create source):
    - **Название** (Name) – укажите название источника сведений об управляемых узлах.
    - **Описание** (Description) – укажите описание источника, например, название связанного проекта.
    - **Среда исполнения** (Execution environment) – укажите среду исполнения, используемую для работы с источником.

      Указанное здесь значение имеет более высокий приоритет, чем среда исполнения, указанная на уровне организации, команды или инвентаря.
    - **Источник** (Source) – выберите тип источника сведений об управляемых узлах.
3. Заполните поля, соответствующие выбранному источнику.

Примечание

Поля **Проект** (Project) и **Инвентарный файл** (Inventory file) доступны только для источников типа «Взято из проекта» (Source from a Project). При этом значение в поле **Инвентарный файл** (Inventory file) можно выбрать только после выбора проекта.

- **Полномочия** (Credential) – если для доступа к источнику требуется аутентификация, выберите полномочие соответствующего типа в этом поле.
- **Проект** (Project) – выберите проект, служащий источником сведений об управляемых узлах.
- **Инвентарный файл** (Inventory file) – выберите файл инвентаря из состава проекта.

  При выборе значения `/ (корень проекта)` (`/ (project root)`) источниками сведений об управляемых узлах считаются все файлы инвентаря, которые контроллер найдет в проекте. Это значение рекомендуется выбирать только в том случае, когда инвентарь в проекте хранится в виде иерархической структуры из файлов и каталогов.
- **Уровень подробности** (Verbosity) – укажите уровень подробности вывода Ansible в журнал выполнения задания синхронизации инвентаря.

  Поддерживаются следующие значения:

  - `0 (Предупреждение)` (`0 (Warning)`) – выводится информация о событиях с уровнем важности не ниже «Предупреждение».
  - `1 (Информация)` (`1 (Info)`) – выводится информация обо всех событиях, происходящих при синхронизации инвентаря.
  - `2 (Отладка)` (`2 (Debug)`) – выводится вся доступная информация обо всех событиях, происходящих при обработке инвентаря.

  Значение по умолчанию: `1 (Информация)` (`1 (Info)`).
- **Фильтр по узлам** (Hosts filter) – укажите регулярное выражение, используемое для фильтрации узлов по названию. Этот фильтр применяется последним, после всех остальных фильтров, предоставляемых используемым расширением инвентаря.
- **Включенная переменная** (Enabled variable) – укажите название переменной из данных об управляемом узле, значение которой следует использовать для определения статуса узла после импорта.
- **Разрешенное значение** (Enabled value).

  Чтобы управляемый узел импортировался со статусом «Вкл» (On), должны выполниться два условия:

  - в свойствах управляемого узла есть переменная с названием, указанным в поле **Включенная переменная** (Enabled variable);
  - значение этой переменной совпадает с указанным в поле **Разрешенное значение** (Enabled value).

  Если такой переменной нет или ее значение отличается от указанного в этом поле, узел импортируется со статусом «Выкл» (Off).
- **Перезаписать** (Overwrite).

  Если эта настройка включена, синхронизация с источником работает следующим образом:

  - Импортированные ранее узлы и группы, удаленные в источнике, удаляются из инвентарного списка.

- Узлы и группы, не привязанные к источнику, перемещаются в первую найденную группу, созданную вручную. Если такой группы нет, все не привязанные к источнику узлы и группы остаются в группе по умолчанию (`all`).

Если эта настройка выключена, при синхронизации из инвентарного списка **не** удаляются узлы и группы, удаленные в источнике.

Предупреждение

Известная проблема

При удалении узла из источника инвентаря синхронизация не приводит к удалению его из инвентаря в контроллере. Поэтому такой узел необходимо удалить вручную из инвентаря контроллера.

- **Перезаписать переменные** (Overwrite variables).

  Если эта настройка включена, при синхронизации с источником значения переменных инвентаря, хранящихся в контроллере, удаляются и заменяются значениями из источника.

  Если эта настройка выключена, при синхронизации с источником хранящиеся в контроллере значения переменных объединяются со значениями в источнике.

Примечание

Для одноименных переменных значение из источника заменяет значение, хранящееся в контроллере.

- **Обновить при запуске** (Update on launch).

  Если эта настройка включена, при запуске связанного с инвентарем задания автоматически запускается задание синхронизации инвентаря с источником.

- **Время жизни кеша (секунды)** (Cache timeout, seconds).

Примечание

Эта настройка доступна только если включен флаг **Обновление при запуске** (Update on launch)

  Время, в течение которого с момента последней синхронизации данные об управляемых узлах считаются свежими.

  Эта настройка может быть полезна, если задания обновления инвентаря создаются быстрее, чем выполняются. Если данные в кеше свежие, задание синхронизации считается успешно выполненным без запуска.

  При значении `0` задания синхронизации запускаются всегда.

  Значение по умолчанию: `0`.

- **Исходные переменные** (Source variables).

  При использовании источника типа «Взято из проекта» (Sourced from a Project) переменные объединяются с переменными инвентаря.

  При использовании источников других типов в этом поле задаются настройки соответствующего расширения инвентаря.

  Ссылки на документацию расширений инвентаря см. в секции *Поддержка расширений*.

4. Нажмите кнопку *Создать источник* (Create source).

Синхронизация списка узлов с источником

Синхронизация используется для первичного импорта сведений об управляемых узлах в контроллер и для актуализации сведений об импортированных ранее узлах.

Чтобы запустить синхронизацию инвентарного списка с источниками, во вкладке **Источники** (Sources) выполните следующие действия:

- Если необходимо синхронизировать инвентарный список со всеми источниками, на панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите пункт *Запуск обновлений инвентаря* (Launch inventory updates).
- Если необходимо синхронизировать инвентарный список только с определенным источником, выполните следующие действия:
  1. В таблице со списком источников нажмите на ссылку с названием необходимого источника.
  2. В окне свойств источника нажмите кнопку *Запуск обновлений инвентарного спика* (Launch inventory update).
  3. Чтобы отслеживать ход выполнения задания синхронизации, нажмите на ссылку в поле **Статус последнего задания** (Last job status).

Настройка расписания синхронизации списка узлов с источником

Чтобы синхронизация с источником выполнялась по расписанию, во вкладке **Источники** (Sources) выполните следующие действия:

1. В таблице со списком источников нажмите на ссылку с названием необходимого источника.
2. Во вкладке **Расписания** (Schedules) нажмите кнопку *Создать расписание* (Create schedule).
3. Заполните форму **Создать расписание** (Create schedule).

   Описание полей формы доступно на странице *Расписания*.
4. Нажмите кнопку *Завершить* (Finish).

Удаление источника

Предупреждение

При удалении источника удаляются все импортированные из него управляемые узлы.

Чтобы удалить источник, выполните следующие действия:

1. В таблице инвентарных списков нажмите на ссылку с названием необходимого обычного инвентарного списка.
2. Выберите вкладку **Источники** (Sources).
3. В таблице источников нажмите на ссылку с названием удаляемого источника.
4. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите пункт *Удалить источник инвентаря* (Delete inventory source).
5. Подтвердите удаление источника.

Умный инвентарный список

В этой секции приводятся инструкции по работе с умными инвентарными списками через веб-интерфейс.

Важно

Эта часть документации находится в стадии разработки.

Сборный инвентарный список

В этой секции приводятся инструкции по работе со *сборными инвентарными списками* через веб-интерфейс.

Создание сборного инвентарного списка

Для создания сборного инвентаря выполните следующие действия:

1. В окне **Инвентарные списки** (Inventories) нажмите на кнопку *Добавить* (Add) и в открывшемся меню выберите пункт **Создать сборный инвентарь** (Create constructed inventory).
2. Заполните форму **Создать сборный инвентарь** (Create constructed inventory):
   - **Название** (Name) – укажите название инвентарного списка.

     Особенности заполнения поля:

     - Название инвентарного списка должно быть уникальным на уровне организации.
     - Название инвентарного списка не может состоять из одних пробельных символов (пробелы, табуляции и так далее).
     - Допускается использование символов кириллицы и специальных символов.
   - **Описание** (Description) – укажите дополнительные данные об инвентарном списке, например, цель его создания или особенности использования.
   - **Организация** (Organization) – выберите организацию, которой принадлежит инвентарный список.
   - **Группы исполняющих узлов** (Instance groups) – группы узлов, используемые для запуска заданий, связанных с инвентарным списком.

     Подробности о порядке выбора узлов для запуска заданий см. в секции *Выбор группы узлов для запуска заданий*.
   - **Входные инвентарные списки** (Input inventories) – выберите инвентарные списки, которые хотите использовать для получения сведений об управляемых узлах.
   - **Время жизни кеша (секунды)** (Cache timeout (seconds)).

     При синхронизации сборного инвентаря автоматически создается источник сведений об управляемых узлах. Эта настройка задает период времени, в течение которого данные в источнике считаются свежими.

     Эта настройка может быть полезна, если задания обновления инвентаря создаются быстрее чем выполняются. Если данные в кеше свежие, задание синхронизации считается успешно выполненным без запуска.

     При значении `0` задания синхронизации запускаются всегда.

     Значение по умолчанию: `0`.
   - **Уровень подробности** (Verbosity) – выберите уровень детализации журнала выполнения задания синхронизации инвентарного списка.

     Поддерживаются следующие значения:

     - `0 (Нормальный) (0 (Normal))`;
     - `1 (Подробный) (1 (Verbose))`;
     - `2 (Более подробный) (2 (More Verbose))`;

  - 3 (Отладка) (3 (Debug));
  - 4 (Отладка подключения) (4 (Connection Debug));
  - 5 (Отладка WinRM) (5 (WinRM Debug)).

  Значение по умолчанию: 0 (Нормальный) (0 (Normal)).

- **Предел** (Limit) – укажите выражение, используемое для фильтрации управляемых узлов.
- **Исходные переменные** (Source variables) – укажите параметры используемого расширения Ansible и привила его использования.

3. Нажмите кнопку *Создать инвентарный список* (Create inventory).
4. Во вкладке **Подробности** (Details) нажмите кнопку *Синхронизировать инвентарь* (Sync inventory).

Управляемые узлы

Окно **Управляемые узлы** (Hosts) обеспечивает детальную работу с управляемыми системами с помощью следующих операций:

- просмотр состояния узлов и истории выполнения на каждом узле;
- анализ статусов проверки доступности и фактов Ansible;
- запуск специальных команд (Ad-hoc) для быстрой диагностики и администрирования;
- устранение проблем соединения и проверка полномочий.

Для перехода к окну **Управляемые узлы** (Hosts) выберите на панели навигации *Автоматизация процессов* ▸ *Инфраструктура* ▸ *Управляемые узлы* (*Automation Execution* ▸ *Infrastructure* ▸ *Hosts*).

Таблица управляемых узлов

Внешний вид окна **Управляемые узлы** (Hosts) представлен на схеме:

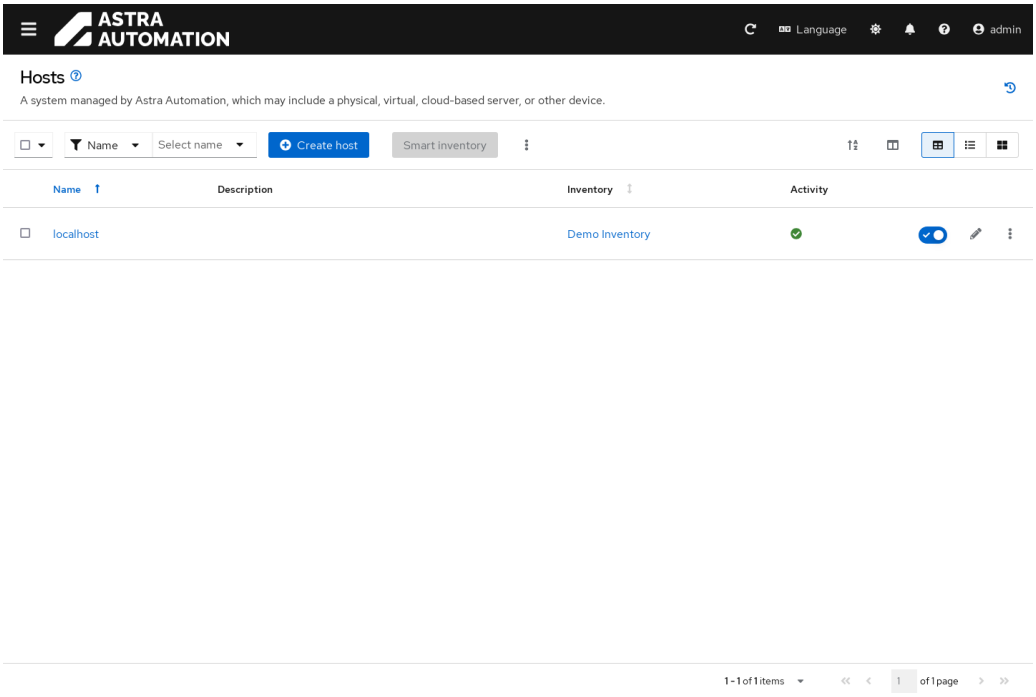


Таблица состоит из столбцов согласно настройке с помощью кнопки *Настроить столбцы* (Manage columns). Полный список столбцов:

- Флаги для выбора нескольких записей.
- **Название** (Name) – ссылка на окно просмотра свойств управляемого узла.

- **Описание** (Description).

  Для управляемых узлов, сведения о которых добавлены из внешних источников, в этом поле выводится значение `imported`.

- **Инвентарный список** (Inventory) – ссылка на окно просмотра свойств *инвентарного списка*, в который входит управляемый узел.
- **Деятельность** (Activity) – статус последнего запущенного задания с использованием этого инвентарного списка.
- Кнопки для быстрого вызова часто выполняемых действий:
  - переключатель *статуса управляемого узла*;
  - кнопка перехода в окно редактирования свойств управляемого узла;
  - кнопка ⋮ для вызова меню дополнительных действий над отдельной записью.

Просмотр

Для получения подробных сведений об управляемом узле нажмите на ссылку с его названием в таблице управляемых узлов. Окно сведений об управляемом узле состоит из следующих вкладок:

- **Подробности** (Details) – общие сведения об управляемом узел.
- **Факты** (Facts) – факты Ansible.

  Примечание

  Факты доступны для просмотра после сбора.

- **Группы** (Groups) – таблица групп узлов, в которые входит управляемый узел. Состоит из следующих столбцов:
  - флаги для выбора нескольких записей;
  - **Название** (Name) – ссылка на окно просмотра свойств группы узлов;
  - **Действия** (Actions) – кнопка перехода в окно редактирования свойств группы узлов.
- **Задания** (Jobs) – таблица заданий, связанных с управляемым узлом.

  Таблица состоит из столбцов согласно настройке с помощью кнопки *Настроить столбцы* (Manage columns). Полный список столбцов:

  - Переключатель подробности вывода основных сведений о задании.

    При нажатии на переключатель выводится дополнительная информация о задании:

    Примечание

    Набор полей зависит от типа задания.

  - Флаги для выбора нескольких записей.
  - **Идентификатор** (ID) – уникальный идентификатор задания.
  - **Название** (Name) – название задания, при нажатии на которое происходит переход в окно просмотра журнала выполнения задания или окно просмотра графа потока заданий.
  - **Статус** (Status) – текущий статус выполнения задания:
    - **Успешно** (Successful) – не было ошибок;

    * **Ошибка** (Error) – возникли ошибки, которые были успешно обработаны;
    * **Сбой** (Failed) – возникли необрабатываемые ошибки, выполнение задания прервано;
    * **Отменено** (Canceled) – задание отменено пользователем;
    * **Ожидание** (Pending) – задание находится в очереди;
    * **Выполнение** (Running) – задание выполняется.
  - **Тип** (Type) – тип задания.
  - **Начато** (Started) – время запуска задания.
  - **Закончено** (Finished) – время завершения, отмены или прерывания задания.
  - **Запущенный** (Launched by) – ссылка на профиль пользователя, запустившего задание.
  - **Расписание** (Schedule) – ссылка на расписание, по которому выполняется запуск задания.
  - **Шаблон задания** (Job template) – ссылка на *шаблон задания*.
  - **Шаблон потока заданий** (Workflow job template) –ссылка на шаблон потока заданий.
  - **Исходное задание потока заданий** (Source workflow job) – ссылка на поток заданий, который инициировал выполнение текущего задания.
  - **Инвентарный список** (Inventory) – ссылка на инвентарный список.
  - **Проект** (Project) – ссылка на *проект*.
  - **Среда исполнения** (Execution environment) – ссылка на среду исполнения, используемую для выполнения задания.
  - **Фрагмент задания** (Job slice) – номер среза и *общее количество срезов*, используемых для выполнения задания.
  - Кнопки для быстрого вызова часто выполняемых действий:
    * перезапуск задания;
    * отмена задания;
    * удаление задания.

Создание

Для создания записи об управляемом узле выполните следующие действия:

1. В окне **Управляемые узлы** (Hosts) нажмите кнопку *Создать узел* (Create host).
2. Заполните форму **Создать узел** (Create host):
   - **Название** (Name) – укажите название управляемого узла.
   - **Описание** (Description) – укажите дополнительные сведения об управляемом узле.
   - **Инвентарный список** (Inventory) – выберите *обычный инвентарный список*, в состав которого добавляется управляемый узел.
   - **Переменные** (Variables) – укажите переменные, действующие на уровне управляемого узла.

     Подробности о порядке присвоения значений переменным см. в документе *Переменные*.
3. Нажмите кнопку *Создать узел* (Create host).

Включение в группу

Управляемые узлы можно включать только в существующие группы в том же инвентарном списке, которому принадлежат сами узлы. Пошаговые инструкции по управлению группами управляемых узлов представлены в секции *Управление группами узлов*.

Чтобы включить управляемый узел в группы узлов, выполните следующие действия:

1. Во вкладке **Группы** (Groups) нажмите кнопку *Ассоциировать групп* (Associate groups).
2. В открывшемся окне **Выберите группы** (Select Groups) включите флаги в строках с теми группами узлов, в которые хотите включить управляемый узел.
3. Нажмите кнопку *Подтвердить* (Confirm).

Исключение из группы

Чтобы исключить управляемый узел из групп узлов, выполните следующие действия:

1. Во вкладке **Группы** (Groups) включите флаги для групп, из которых хотите исключить управляемый узел.
2. Нажмите кнопку *Отвязать* (Disassociate).
3. Подтвердите исключение узла из групп узлов.

Удаление

Чтобы удалить записи об управляемых узлах, выполните следующие действия:

1. В окне **Управляемые узлы** (Hosts) включите флаги напротив удаляемых записей.
2. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите **Удалить узлы** (Delete hosts).
3. Подтвердите удаление.

Группы исполняющих узлов

Окно **Группы исполняющих узлов** (Instance Groups) предназначено для управления наборами вычислительных ресурсов автоматизации с помощью следующих операций:

- распределение задач по узлам исполнения или контейнерным группам;
- настройка политики балансировки нагрузки и изоляции рабочих нагрузок;
- создание выделенных групп для повышения отказоустойчивости и производительности.

Для перехода к окну **Группы исполняющих узлов** (Instance Groups) выберите на панели навигации *Автоматизация процессов* ▸ *Инфраструктура* ▸ *Группы исполняющих узлов* (*Automation Execution* ▸ *Infrastructure* ▸ *Instance Groups*).

Таблица групп исполняющих узлов

Внешний вид окна **Группы исполняющих узлов** (Instance Groups) представлен на схеме:

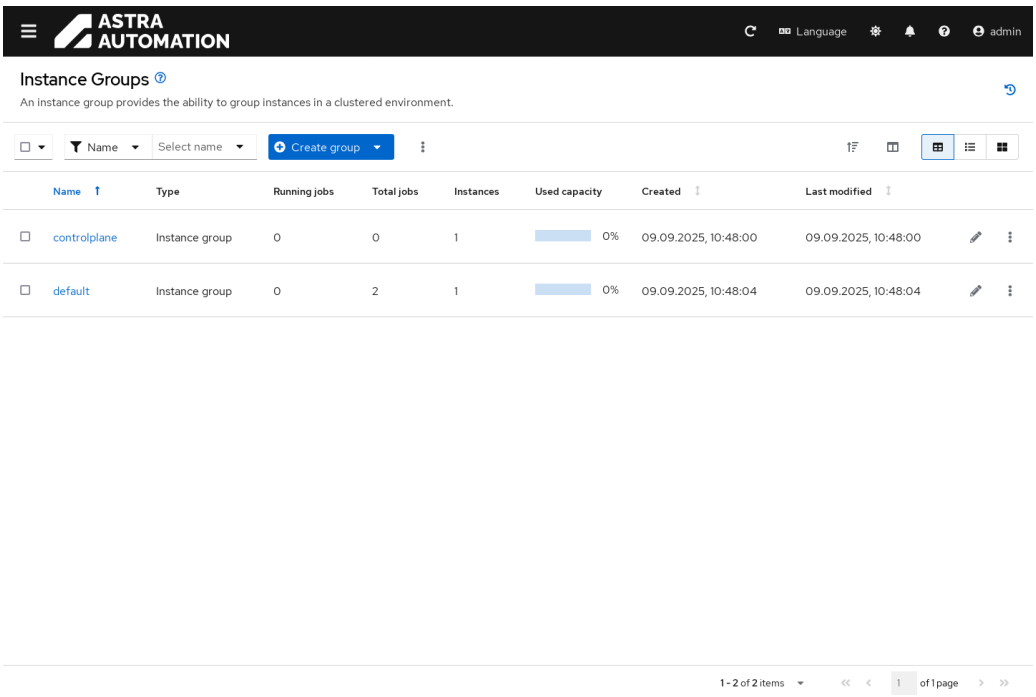


Таблица групп исполняющих узлов состоит из столбцов согласно настройке с помощью кнопки *Настроить столбцы* (Manage columns). Полный список столбцов:

- Флаги для выбора нескольких записей.
- **Название** (Name) – ссылка для перехода в окно просмотра подробных сведений о группе узлов или группе контейнеров.
- **Тип** (Type) – тип записи (группа узлов контроллера или группа контейнеров).
- **Выполняемые задания** (Running jobs) – количество заданий, выполняемых в текущий момент в группе исполняющих узлов или группе контейнеров (при использовании Kubernetes).
- **Общее количество заданий** (Total jobs) – количество заданий, выполненных когда-либо в группе исполняющих узлов или группе контейнеров (при использовании Kubernetes).
- **Экземпляры** (Instances) – количество узлов или контейнеров в группе.

> **Примечание**
>
> Один и тот же узел может входить в несколько групп узлов.

- **Используемая мощность** (Used capacity) – степень утилизации ресурсов группы узлов. Чем выше значение в этом столбце, тем выше нагрузка на узлы группы.
- **Дата создания** (Created) – дата и время создания записи.
- **Последнее изменение** (Last modified) – дата и время последнего изменения записи.
- Кнопка для перехода в окно изменения данных о группе.
- Кнопка ⋮ для вызова меню дополнительных действий над отдельной записью.

Просмотр группы узлов

Для получения подробных сведений о группе исполняющих узлов нажмите на ссылку с ее названием в таблице групп. Окно сведений о группе узлов контроллера состоит из следующих вкладок:

- **Подробности** (Details) – общие сведения о группе.

- **Экземпляры** (Instances) – таблица узлов группы и кнопки управления ими.

  Таблица состоит из столбцов согласно настройке с помощью кнопки *Настроить столбцы* (Manage columns). Полный список столбцов:

  - Флаги для выбора нескольких записей.
  - **Название** (Name) – ссылка для перехода в окно просмотра подробных сведений об узле.
  - **Статус** (Status) – текущий статус узла.

    Для проверки статуса узла выполните *проверку его работоспособности*.
  - **Тип узла** (Node type).

    Подробности о типах узлов см. в документе *Особенности архитектуры*.
  - **Регулировка производительности** (Capacity adjustment) – ползунок изменения максимального количества ветвлений для всех процессов, выполняемых на узле.

    Максимальное значение этого параметра зависит от количества ядер CPU и объема оперативной памяти на узле.
  - **Используемая мощность** (User capacity) – индикатор утилизации ресурсов узла. Чем выше значение в этом поле, тем больше нагрузка на узел.
  - **Выполняемые задания** (Running jobs) – количество заданий, выполняемых в текущий момент в группе исполняющих узлов или группе контейнеров (при использовании Kubernetes).
  - **Общее количество заданий** (Total jobs) – количество заданий, выполненных когда-либо в группе исполняющих узлов или группе контейнеров (при использовании Kubernetes).
  - **Память** (Memory) – объем оперативной памяти узла.
  - **Тип политики** (Policy type) – политика добавления экземпляра в группы.
  - **Дата создания** (Created) – дата и время создания записи.
  - **Последнее изменение** (Last modified) – дата и время последнего изменения записи.
  - Переключатель статуса узла.
- **Командный доступ** (Team Access) – таблица команд и назначенных им ролей на доступ к группе узлов.
- **Доступ пользователей** (User Access) – таблица пользователей и назначенных им ролей на доступ к группе узлов.
- **Задания** (Jobs) – таблица заданий, запущенных на узлах группы.

  Таблица состоит из столбцов согласно настройке с помощью кнопки *Настроить столбцы* (Manage columns). Полный список столбцов:

  - Переключатель подробности вывода основных сведений о задании.

    При нажатии на переключатель выводится дополнительная информация о задании:

    Примечание

    Набор полей зависит от типа задания.
  - Флаги для выбора нескольких записей.
  - **Идентификатор** (ID) – уникальный идентификатор задания.

- **Название** (Name) – название задания, при нажатии на которое происходит переход в окно просмотра журнала выполнения задания или окно просмотра графа потока заданий.
- **Статус** (Status) – текущий статус выполнения задания:
  - **Успешно** (Successful) – не было ошибок;
  - **Ошибка** (Error) – возникли ошибки, которые были успешно обработаны;
  - **Сбой** (Failed) – возникли необрабатываемые ошибки, выполнение задания прервано;
  - **Отменено** (Canceled) – задание отменено пользователем;
  - **Ожидание** (Pending) – задание находится в очереди;
  - **Выполнение** (Running) – задание выполняется.
- **Тип** (Type) – тип задания.
- **Начато** (Started) – время запуска задания.
- **Закончено** (Finished) – время завершения, отмены или прерывания задания.
- **Запущенный** (Launched by) – ссылка на профиль пользователя, запустившего задание.
- **Расписание** (Schedule) – ссылка на расписание, по которому выполняется запуск задания.
- **Шаблон задания** (Job template) – ссылка на *шаблон задания*.
- **Шаблон потока заданий** (Workflow job template) –ссылка на шаблон потока заданий.
- **Исходное задание потока заданий** (Source workflow job) – ссылка на поток заданий, который инициировал выполнение текущего задания.
- **Инвентарный список** (Inventory) – ссылка на инвентарный список.
- **Проект** (Project) – ссылка на *проект*.
- **Среда исполнения** (Execution environment) – ссылка на среду исполнения, используемую для выполнения задания.
- **Фрагмент задания** (Job slice) – номер среза и *общее количество срезов*, используемых для выполнения задания.
- Кнопки для быстрого вызова часто выполняемых действий:
  - перезапуск задания;
  - отмена задания;
  - удаление задания.

Создание группы узлов

Для создания группы узлов выполните следующие действия:

1. В окне **Группы исполняющих узлов** (Instance Groups) нажмите кнопку *Создать группу* (Create group).
2. В открывшемся меню выберите **Добавить группу исполняющих узлов** (Create instance group).
3. Заполните форму **Добавить группу исполняющих узлов** (Create instance group):
   - **Название** (Name) – укажите название группы узлов.

     Требования к названию:
     - название должно быть уникальным;
     - запрещено создание группы с названием `default`.

- **Политика минимального количества** (Policy instance minimum) – укажите минимальное количество узлов, которые должны быть включены в состав группы при подключении к кластеру новых узлов.

  Значение по умолчанию: `0`.

- **Политика процентного соотношения** (Policy instance percentage) – укажите минимальный процент узлов от общего количества, которые должны быть включены в состав группы при подключении к кластеру новых узлов.

  Значение по умолчанию: `0`.

- **Максимальное количество одновременных заданий** (Max concurrent jobs) – укажите максимальное количество заданий, которые могут быть одновременно запущены на узлах группы.

  При значении `0` количество заданий не ограничено.

  Значение по умолчанию: `0`.

- **Максимальное количество ответвленных процессов** (Max forks) – максимальное количество ответвленных процессов всех заданий, запущенных на узлах группы.

  При значении `0` количество ветвлений не ограничено.

  Значение по умолчанию: `0`.

4. Нажмите кнопку *Создать группу исполняющих узлов* (Create instance group).

Включение узлов в группу

Чтобы включить узлы в группу, выполните следующие действия:

1. В таблице групп узлов и контейнеров нажмите на название группы, в которую необходимо добавить узлы.
2. Выберите вкладку **Экземпляры** (Instances).
3. Нажмите кнопку *Ассоциировать исполняющий узел* (Associate instance).
4. В диалоговом окне **Выберите исполняющие узлы** (Select instances) включите флаги в строках с необходимыми узлами и нажмите кнопку *Подтвердить* (Confirm).

Отвязка узлов от группы

Чтобы исключить узлы из группы, выполните следующие действия:

1. В таблице групп узлов и контейнеров нажмите на название группы, узлы которой необходимо отвязать.
2. Выберите вкладку **Экземпляры** (Instances).
3. Включите флаги в строках с исключаемыми узлами.
4. Нажмите кнопку *Отсоединить исполняющий узел* (Disassociate instance).
5. Подтвердите исключение узлов из группы.

Проверка работоспособности узлов

Чтобы проверить работоспособность узлов группы, выполните следующие действия:

1. В таблице групп узлов и контейнеров нажмите на название группы, узлы которой необходимо проверить.
2. Выберите вкладку **Экземпляры** (Instances).
3. Включите флаги в строках с узлами, работоспособность которых хотите проверить.
4. Нажмите кнопку *Запустите проверку работоспособности* (Run health check).
5. Перезагрузите страницу.

Создание группы контейнеров

Для создания группы контейнеров выполните следующие действия:

1. В окне **Группы исполняющих узлов** (Instance Groups) нажмите кнопку *Создать группу* (Create group).
2. В открывшемся меню выберите **Создать группу контейнеров** (Create container group).
3. Заполните форму **Создать группу контейнеров** (Create container group):
   - **Название** (Name) – укажите название группы контейнеров.
   - **Полномочие** (Credential) – если для доступа к группе контейнеров требуется аутентификация, выберите соответствующее *полномочие* типа «Токен доступа OpenShift или Kubernetes API» (OpenShift or Kubernetes API Bearer Token).
   - **Максимальное количество одновременных заданий** (Max concurrent jobs) – максимальное количество заданий, которые могут быть одновременно запущены на группе контейнеров.

     При значении `0` количество заданий не ограничено.

     Значение по умолчанию: `0`.
   - **Максимальное количество ответвленных процессов** (Max forks) – максимальное количество ответвленных процессов всех заданий, запущенных на группе контейнеров.

     При значении `0` количество ветвлений не ограничено.

     Значение по умолчанию: `0`.
   - **Настроить спецификацию pod** (Customize pod spec). Если эта настройка включена, становятся доступными для изменения настройки спецификации подов. Укажите их в поле **Переопределенная спецификация подов** (Pod spec override).
4. Нажмите кнопку *Создать группу контейнеров* (Create container group).

Удаление группы узлов или группы контейнеров

Чтобы удалить группы узлов или группы контейнеров, выполните следующие действия:

1. В таблице групп исполняющих узлов установите флаги в строках с группами, которые следует удалить.
2. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите пункт *Удалить группы исполняющих узлов* (Delete instance groups).
3. Подтвердите удаление.

Узлы

Окно **Узлы** (Instances) используется для просмотра статуса узлов системы автоматизации процессов и запуска диагностических команд. В систему входят узлы плоскости управления (Control Plane, Controller) и плоскости исполнения (Execution Plane). Для перехода к нему выберите на панели навигации *Автоматизация процессов* ▸ *Инфраструктура* ▸ *Узлы* (*Automation Execution* ▸ *Infrastructure* ▸ *Instances*).

Таблица узлов

Внешний вид окна **Узлы** (Instances) представлен на схеме:

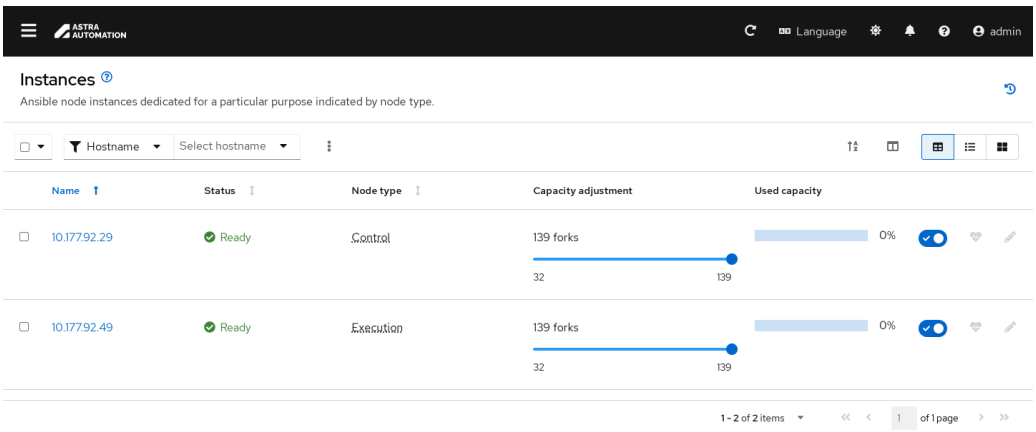


Таблица групп исполняющих узлов состоит из столбцов согласно настройке с помощью кнопки *Настроить столбцы* (Manage columns). Полный список столбцов:

- Переключатели вывода подробной информации об узлах.
- Флаги для выбора нескольких записей.
- **Название** (Name) – ссылка для перехода в окно просмотра подробных сведений об узле.
- **Статус** (Status) – текущий статус узла.

  Для проверки и обновления статуса узла выполните *проверку его работоспособности*.
- **Тип узла** (Node type) – тип узла.

  Подробности о типах узлов см. в документе *Особенности архитектуры*.
- **Регулировка производительности** (Capacity adjustment) – ползунок изменения максимального количества ветвлений для всех процессов, запущенных на узле.

  Максимальное значение этого параметра зависит от количества ядер CPU и объема оперативной памяти на узле.
- **Используемая мощность** (Used capacity) – индикатор утилизации ресурсов узла. Чем выше значение в этом поле, тем больше нагрузка на узел.
- **Выполняемые задания** (Running jobs) – количество заданий, выполняемых узлом в данный момент.
- **Общее количество заданий** (Total jobs) – общее число заданий, назначенных на узел (включая завершенные и ожидающие).
- **Память** (Memory) – объем оперативной памяти, доступной на узле.
- **Тип политики** (Policy type) – тип политики распределения заданий для узла.

  Этот параметр по умолчанию имеет значение **Автоматически** (Auto), которое невозможно изменить в графическом интерфейсе. В этом режиме система автоматически балансирует нагрузку и оптимизирует использование емкости узлов.

  Если значение **Инструкция** (Manual) – распределение выполняется вручную, например, через настройку групп узлов или указание конкретных исполнителей в конфигурации заданий.

  Значение столбца **Тип политики** (Policy type) зависит от состояния параметра **Управляется политикой** (Managed by policy), который включается в настройках узла.

Примечание

Изменение параметра **Управляется политикой** (Managed by policy) через графический интерфейс недоступно. Управление узлами выполняется автоматически платформой или вручную с помощью API и утилиты `aa-setup`.

- **Дата создания** (Created) – дата и время добавления узла в систему.
- **Последнее изменение** (Last modified) – дата и время последнего обновления конфигурации узла.
- Переключатель статуса узла.
- Кнопка для запуска проверки работоспособности узла.
- Кнопка для перехода в окно изменения данных об узле.

Проверка работоспособности узлов

Чтобы проверить работоспособность узлов, выполните следующие действия:

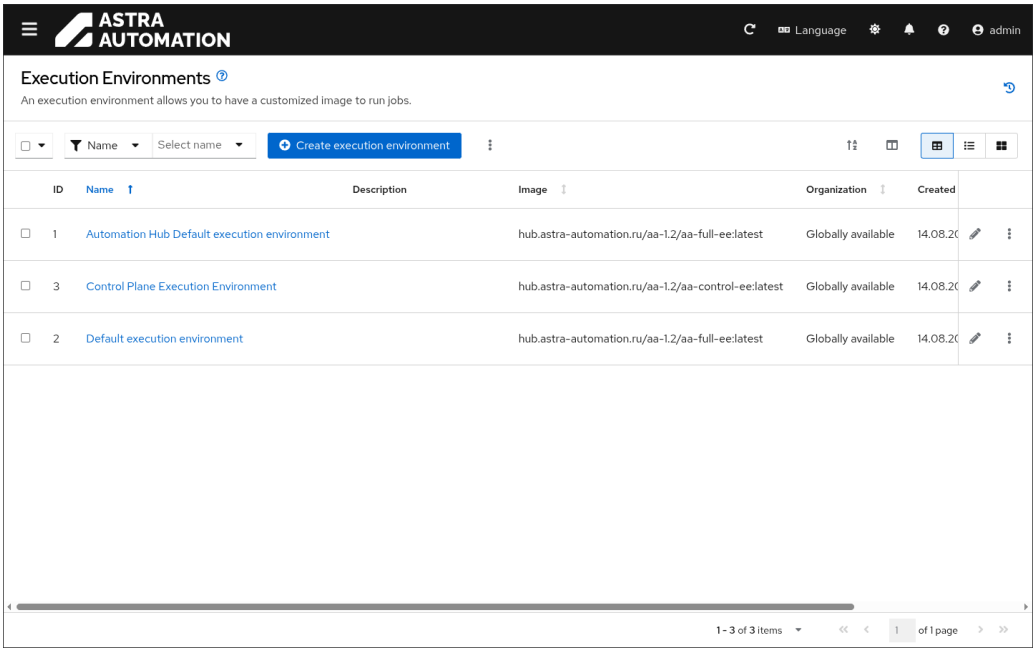
1. В таблице узлов включите флаги в строках с теми узлами, работоспособность которых хотите проверить.
2. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите пункт *Запустите проверку работоспособности* (Run health check).
3. Обновите страницу браузера.

Среды исполнения

Для перехода к списку образов сред исполнения (далее – образов), которые может использовать Automation Controller, выберите на панели навигации *Автоматизация процессов* ▸ *Инфраструктура* ▸ *Среды исполнения* (*Automation Execution* ▸ *Infrastructure* ▸ *Execution Environments*).

Таблица сред исполнения

Внешний вид окна **Среды исполнения** (Execution Environments) представлен на схеме:



На панели инструментов размещаются поле поиска, кнопка *Создать среду исполнения* (Create execution environment) и кнопка ⋮ для вызова действий над выделенными записями.

Таблица сред исполнения состоит из следующих столбцов:

- Флаги для выбора нескольких записей.
- **Идентификатор** (ID) – идентификатор образа в Automation Controller.
- **Название** (Name) – название образа в Automation Controller.

- **Описание** (Description) – краткое описание образа.
- **Образ** (Image) – ссылка на образ в реестре образов.
- **Организация** (Organization) – организация-владелец образа.

  Если образ доступен всем организациям, в этом столбце выводится строка «Доступный глобально» (Globally available).
- **Дата создания** (Created) – дата и время создания записи.
- **Последнее изменение** (Last modified) – дата и время последнего изменения записи.
- Кнопка перехода в окно изменения свойств образа.
- Кнопка ⋮ – вызов меню дополнительных действий.

Просмотр

Для получения подробных сведений об образе нажмите на ссылку с его названием. Окно просмотра сведений об образе состоит из следующих вкладок:

- **Подробности** (Details) – общие сведения об образ.
- **Шаблоны** (Templates) – *шаблоны заданий*, связанные с образом.

Создание

Для создания записи о среде исполнения выполните следующие действия:

1. В окне **Среды исполнения** (Execution Environments) нажмите кнопку *Создать среду исполнения* (Create execution environment).
2. Заполните форму **Создать среду исполнения** (Create execution environment):
   - **Название** (Name) – название образа в интерфейсе пользователя.

     Примечание

     Название должно быть уникальным на уровне контроллера.

   - **Образ** (Image) – ссылка на образ в реестре образов.
   - **Pull** – *параметры загрузки образа*.
   - **Описание** (Description) – краткое описание образа.
   - **Организация** (Organization) – *организация*, которой принадлежит образ.

     Чтобы образ могли использовать все организации, оставьте поле незаполненным.
   - **Registry credential** – если для доступа к реестру образов требуется авторизация, выберите в этом поле *полномочия* типа «Реестр контейнеров» (Container Registry).

     Инструкции по созданию полномочий этого типа приведены в секции *Реестр контейнеров*.
3. Нажмите кнопку *Создать среду исполнения* (Create execution environment).

Изменение

Для изменения записи о среде исполнения выполните следующие действия:

1. В окне **Среды исполнения** (Execution Environments) нажмите на названии образа, который необходимо редактировать.
2. Нажмите кнопку *Редактировать среду исполнения* (Edit execution environment).
3. Внесите необходимые изменения в форме **Редактировать <среда исполнения>** (Edit <execution environment>).

4. Нажмите кнопку *Сохранить среду исполнения* (Save execution environment).

Удаление

Для удаления сред исполнения выполните следующие действия:

1. В окне **Среды исполнения** (Execution Environments) включите флаги напротив удаляемых записей.
2. Нажмите кнопку ⋮ и в открывшемся меню выберите **Удалить среды исполнения** (Delete execution environments).
3. Подтвердите удаление.

Полномочия

Окно **Полномочия** (Credentials) используется для централизованного управления *учетными данными*, необходимыми для автоматизации, с помощью следующих операций:

- создание и хранение учетных данных для облаков, SSH, Vault и других систем;
- настройка доступа к внешним сервисам и репозиториям;
- разделение полномочий между командами согласно политике *RBAC*.

Для перехода к окну **Полномочия** (Credentials) выберите на панели навигации *Автоматизация процессов* ▸ *Инфраструктура* ▸ *Полномочия* (*Automation Execution* ▸ *Infrastructure* ▸ *Credentials*).

Таблица полномочий

Внешний вид окна **Полномочия** (Credentials) представлен на схеме:

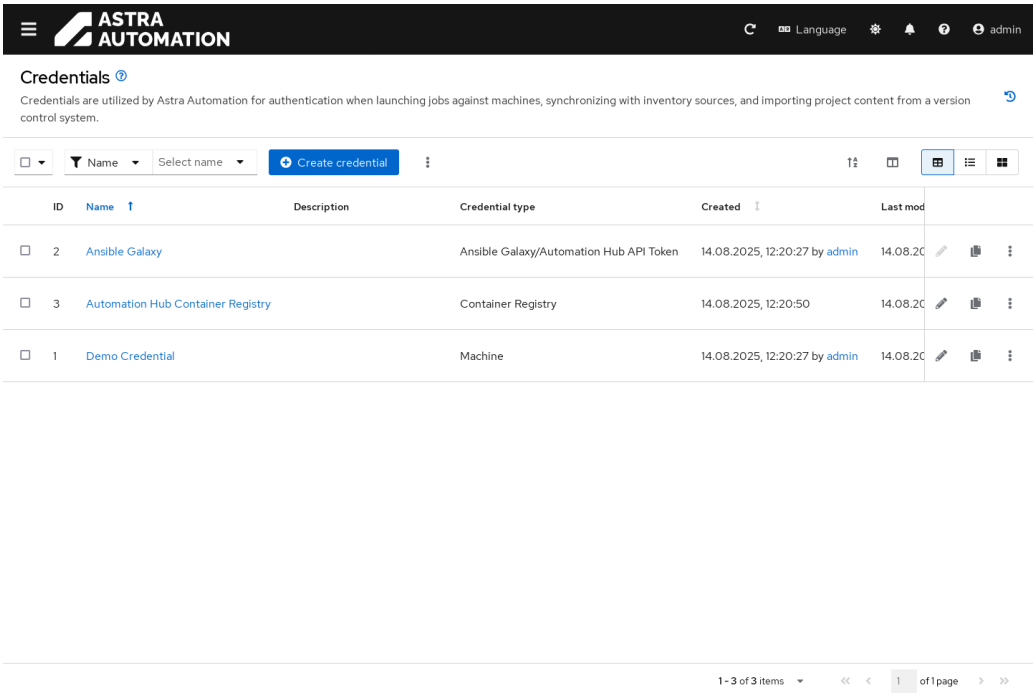


Таблица полномочий состоит из следующих столбцов:

- Флаги для выбора нескольких записей.
- **Идентификатор** (ID) – уникальный идентификатор записи таблицы.
- **Название** (Name) – ссылка с названием полномочия.

  При нажатии на ссылку происходит переход в окно просмотра подробной информации о полномочии.
- **Описание** (Description) – краткое описание полномочия.

- **Тип полномочий** (Credential type) – тип полномочий.
- **Дата создания** (Created) – дата и время создания записи.
- **Последнее изменение** (Last modified) – дата и время последнего изменения записи.
- Кнопка для перехода в окно изменения данных о полномочии.
- Кнопка для копирования полномочия.
- Кнопка ⋮ для вызова меню дополнительных действий над отдельной записью.

Просмотр сведений о полномочиях

Для получения подробных сведений о полномочии нажмите на ссылку с его названием в таблице полномочий.

Внешний вид окна со сведениями о полномочии зависит от их типа и может состоять из следующих вкладок:

- **Подробности** (Details) – общие сведения о полномочии;
- **Шаблоны заданий** (Job Templates) – таблица *шаблонов заданий*, использующих полномочие;
- **Командный доступ** (Team Access) – таблица команд и назначенных им ролей на доступ к полномочию;
- **Доступ пользователей** (User Access) – таблица пользователей и назначенных им ролей на доступ к записи полномочию.

Создание полномочия

Для создания полномочия выполните следующие действия:

1. В окне **Полномочия** (Credentials) нажмите кнопку *Создать полномочие* (Create credential).
2. Заполните форму **Создать полномочие** (Create credential).
   - **Название** (Name) – название полномочия.
   - **Описание** (Description) – дополнительная информация о полномочии, например, цель его создания или правила использования.
   - **Организация** (Organization) – организация, которой принадлежит полномочие.

     Если поле не заполнено, полномочие принадлежит всем организациям контроллера.
   - **Тип полномочий** (Credential type) – *тип полномочия*.

     При выборе значения в этом поле в форме появляются соответствующие поля.
3. Заполните поля, соответствующие выбранному типу полномочия.

   Особенности заполнения полей для каждого типа полномочия приведены в следующих секциях.
4. Нажмите кнопку *Создать полномочие* (Create credential).

ALD Pro

Важно

Эта часть документации находится в стадии разработки.

API-токен Ansible Galaxy/Automation Hub

При создании полномочия этого типа доступны следующие поля:

- **URL сервера Galaxy** (Galaxy Server URL) – URL репозитория в реестре коллекций Ansible.
- **URL сервера аутентификации** (Auth Server URL) – точка доступа API сервера Keycloak, если используется *SSO*.
- **Токен API** (API Token) – токен, используемый для аутентификации в реестре коллекций.

Подробности о полномочиях этого типа см. в секции *API-токен Ansible Galaxy/Automation Hub*.

Astra Automation Platform

Важно

Эта часть документации находится в стадии разработки.

AWS Secrets Manager lookup

Важно

Эта часть документации находится в стадии разработки.

Bitbucket Data Center HTTP Access Token

Важно

Эта часть документации находится в стадии разработки.

Centrify Vault Credential Provider Lookup

Важно

Эта часть документации находится в стадии разработки.

CyberArk Central Credential Provider Lookup

Важно

Эта часть документации находится в стадии разработки.

CyberArk Conjur Secrets Manager Lookup

Важно

Эта часть документации находится в стадии разработки.

GitHub App Installation Access Token Lookup

Важно

Эта часть документации находится в стадии разработки.

Google Compute Engine

При создании полномочия этого типа доступны следующие поля:

- **Файл JSON сервисной учетной записи** (Service account JSON file) – файл с реквизитами сервисной учетной записи Google Compute Cloud. При выборе файла JSON, содержащего необходимые сведения, остальные поля формы заполняются автоматически.

  Пошаговые инструкции по созданию такого файла см. в документации Google Compute Cloud[187].

- **Адрес электронной почты сервисной учетной записи** (Service Account Email Address) – адрес электронной почты, связанный с сервисной учетной записью Google Compute Engine.
- **Project** – идентификатор проекта в Google Compute Engine.
- **Закрытый ключ RSA** (RSA Private Key) – содержимое приватного ключа SSH, используемого для доступа к Google Compute Engine с реквизитами сервисной учетной записи.

Подробности об этом типе полномочий см. в подразделе *Google Compute Engine*.

HashiCorp Vault Secret Lookup

При создании полномочия этого типа доступны следующие поля:

- **URL сервера** (Server URL) – URL сервера HashiCorp Vault, используемого для хранения секретов.
- **Токен** (Token) – токен аутентификации на сервере HashiCorp Vault.
- **Сертификат CA** (CA Certificate) – сертификат, используемый для защиты подключения к серверу HashiCorp Vault.
- **AppRole role_id** – идентификатор AppRole, используемой для аутентификации на сервере HashiCorp Vault.
- **AppRole secret_id** – секрет AppRole, используемые для аутентификации на сервере HashiCorp Vault.
- **Клиентский сертификат** (Client Certificate) – сертификат в формате PEM, используемый при аутентификации TLS. Должен включать в себя в том числе все необходимые промежуточные сертификаты, необходимые для аутентификации в HashiCorp Vault.
- **Ключ клиентского сертификата** (Client Certificate Key) – приватный ключ в формате PEM, используемый при аутентификации TLS.
- **Роль аутентификации TLS** (TLS Authentication Role) – название роли или сертификата, используемых при аутентификации TLS в HashiCorp Vault.
- **Название пространства имен (только для Vault Enterprise)** (Namespace name (Vault Enterprise only)) – название пространства имен в HashiCorp Vault.

  Примечание

  Для использования значения, указанного в этом поле, требуется Hashicorp Vault редакции Enterprise.

- **Роль Kubernetes** (Kubernetes role) – название роли, используемой для аутентификации Kubernetes.
- **Имя пользователя** (Username) – название учетной записи пользователя HashiCorp Vault.

[187] https://cloud.google.com/iam/docs/keys-create-delete

- **Пароль** (Password) – пароль пользователя HashiCorp Vault.
- **Путь к авторизации** (Path to Auth) – значение в этом поле следует указывать, если путь к авторизации на сервере HashiCorp Vault отличается от значения `/approle`, используемого по умолчанию.
- **Версия API** (API Version) – версия API HashiCorp Vault.
  - v1 – статичный поиск;
  - v2 – поиск с версионированием.

Подробности об этом типе полномочий см. в секции *HashiCorp Vault Secret Lookup*.

HashiCorp Vault Signed SSH

Важно

Эта часть документации находится в стадии разработки.

Insights

Важно

Эта часть документации находится в стадии разработки.

Microsoft Azure Key Vault

Важно

Эта часть документации находится в стадии разработки.

OpenStack

При создании полномочия этого типа доступны следующие поля:

- **Имя пользователя** (Username) – название учетной записи пользователя OpenStack.
- **Пароль (ключ API)** (Password (API Key)) – пароль или ключ API OpenStack.
- **Узел (URL для аутентификации)** (Host (Authentication URL)) – URL, используемый для аутентификации в OpenStack, например:

  ```
  https://openstack.example.com/auth/
  ```

- **Проект (имя арендатора)** (Project (Tenant Name)) – название или идентификатор проекта (тенанта) OpenStack. Как правило, совпадает с именем пользователя OpenStack.
- **Проект (доменное имя)** (Project (Domain Name)) – доменное имя проекта (тенанта) OpenStack.
- **Доменное имя** (Domain Name) – FQDN сервера OpenStack.

  Заполнение этого поля требуется только при использовании сервера аутентификации Keystone v3.
- **Название региона** (Region Name) – название региона OpenStack.
- **Проверка SSL** (Verify SSL) – если эта настройка включена, при подключении к OpenStack проверяется корректность сертификата SSL.

  Значение по умолчанию: выключено.

Red Hat Satellite 6

Важно

Эта часть документации находится в стадии разработки.

Thycotic DevOps Secrets Vault

Важно

Эта часть документации находится в стадии разработки.

Thycotic Secret Server

Важно

Эта часть документации находится в стадии разработки.

Vault

Важно

Эта часть документации находится в стадии разработки.

VMware vCenter

Важно

Эта часть документации находится в стадии разработки.

Web-сервисы Amazon

Важно

Эта часть документации находится в стадии разработки.

Виртуализация Red Hat

Важно

Эта часть документации находится в стадии разработки.

Диспетчер ресурсов Microsoft Azure

Важно

Эта часть документации находится в стадии разработки.

Конфигурация terraform backend

При создании полномочия этого типа доступны следующие поля:

- **Backend configuration** – конфигурация Terraform backend в формате языка конфигурации Hashicorp.
- **Google Cloud Platform account credentials** – учетные данные Google Cloud Platform[188] в формате JSON.

Личный токен доступа к GitFlic

Важно

Эта часть документации находится в стадии разработки.

Личный токен доступа к GitHub

При создании полномочия этого типа укажите в поле **Токен** (Token) значение токена, созданного в профиле GitHub.

Подробности о полномочиях этого типа см. в секции *Личный токен доступа к GitHub*.

Машина

При создании полномочия этого типа доступны следующие поля:

- **Имя пользователя** (Username) – название учетной записи пользователя, с реквизитами которого контроллер должен подключаться к управляемому узлу.
- **Пароль** (Password) – пароль пользователя, с реквизитами которого контроллер должен подключаться к управляемому узлу.
- **Закрытый ключ SSH** (SSH Private Key) – содержимое приватного ключа SSH.
- **Подписанный сертификат SSH** (Signed SSH Certificate) – содержимое сертификата, используемого для защиты подключения SSH.
- **Парольная фраза закрытого ключа** (Private Key Passphrase) – если приватный ключ SSH защищен паролем, укажите его в этом поле.
- **Способ повышения привилегий** (Privilege Escalation Method) – метод повышения привилегий.
- **Имя пользователя с повышенными привилегиями** (Privilege Escalation Username) – если для повышения привилегий команды следует выполнять от имени пользователя, отличающегося от указанного в поле **Имя пользователя** (Username), укажите его в этом поле.
- **Пароль с повышенными привилегиями** (Privilege Escalation Password) – пароль пользователя, название учетного записи которого указано в поле **Имя пользователя с повышенными привилегиями** (Privilege Escalation Username).

Подробности о полномочиях этого типа см. в секции *Машина*.

Открытый ключ GPG

При создании полномочия этого типа укажите в поле **Открытый ключ GPG** (GPG Public Key) содержимое публичного ключа GPG, который Automation Controller должен использовать для проверки подписи содержимого.

Подробности о полномочиях этого типа см. в секции *Открытый ключ GPG*.

Инструкции по созданию ключей GPG и их использованию для подписи содержимого Ansible см. в документе *Ansible Sign*.

[188] https://cloud.google.com

Платформа автоматизации Red Hat Ansible

Важно

Эта часть документации находится в стадии разработки.

Реестр контейнеров

Важно

Эта часть документации находится в стадии разработки.

Сеть

Важно

Эта часть документации находится в стадии разработки.

Система управления версиями

При создании полномочия этого типа доступны следующие поля:

- **Имя пользователя** (Username) – название учетной записи пользователя, с реквизитами которого контроллер должен подключаться к источнику кода.
- **Пароль** (Password) – пароль пользователя, с реквизитами которого контроллер должен подключаться к источнику кода.
- **Закрытый ключ SCM** (SCM Private Key) – содержимое приватного ключа SSH, используемого для подключения к источнику кода.
- **Парольная фраза закрытого ключа** (Private Key Passphrase) – если приватный ключ SSH защищен паролем, укажите его в этом поле.

Подробности о полномочиях этого типа см. в секции *Система управления версиями*.

Токен доступа OpenShift или Kubernetes API

Важно

Эта часть документации находится в стадии разработки.

Токен персонального доступа GitLab

При создании полномочия этого типа укажите в поле **Токен** (Token) значение токена, созданного в профиле GitLab.

Подробности о полномочиях этого типа см. в секции *Токен персонального доступа GitLab*.

Удаление полномочия

Чтобы удалить полномочия, выполните следующие действия:

1. В окне **Полномочия** (Credentials) включите флаги напротив удаляемых записей.
2. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите **Удалить полномочия** (Delete credentials).
3. Подтвердите удаление.

Типы полномочий

Окно **Типы полномочий** (Credential Types) используется для управления пользовательскими (нестандартными) типами *полномочий*. Для перехода к нему выберите на панели навигации *Автоматизация процессов* ▸ *Инфраструктура* ▸ *Типы полномочий* (*Automation Execution* ▸ *Infrastructure* ▸ *Credential Types*).

Таблица типов полномочий

Внешний вид окна **Типы полномочий** (Credential Types) представлен на схеме:

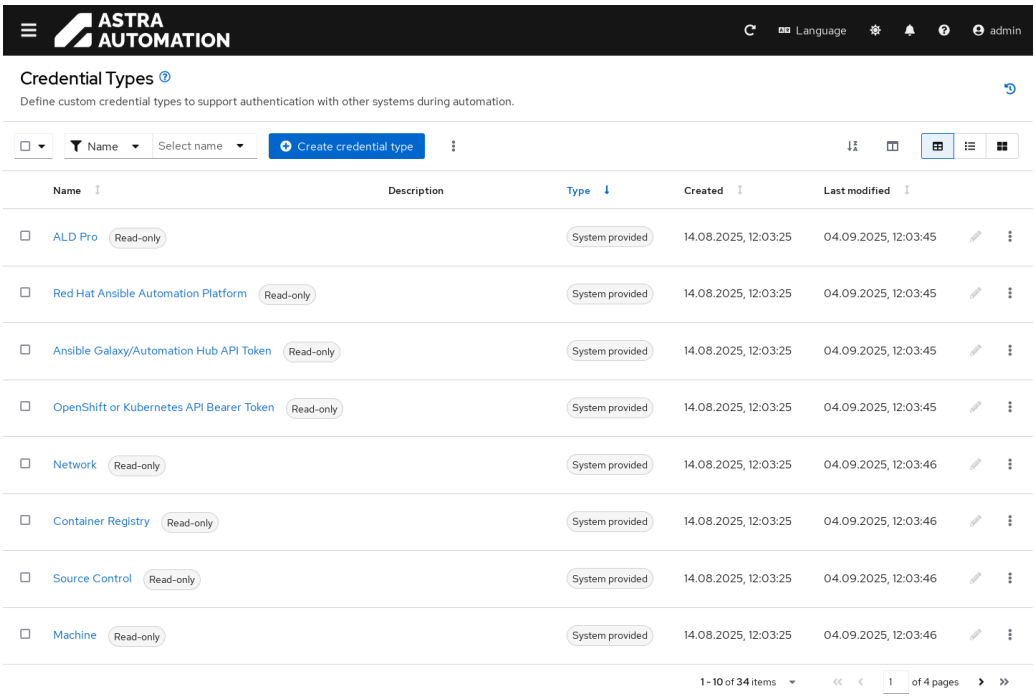


Таблица типов полномочий состоит из следующих колонок:

- Флаги для выбора нескольких записей.
- **Название** (Name) – ссылка с названием типа полномочий.

  При нажатии на ссылку происходит переход в окно просмотра подробной информации о типе полномочий.
- **Описание** (Description) – краткое описание типа полномочий.
- **Тип** (Type) – создатель полномочия.

  Возможные значения:

  - **System provided** – системный тип полномочий, недоступен для редактирования.
  - Пустое значение – пользовательский тип полномочий.
- **Дата создания** (Created) – дата и время создания записи.
- **Последнее изменение** (Last modified) – дата и время последнего изменения записи.
- Кнопка для перехода в окно изменения данных о типе полномочий.
- Кнопка ⋮ для вызова меню дополнительных действий над отдельной записью.

Примечание

Для управления пользовательскими типами полномочий необходимы привилегии администратора контроллера.

Создание

Для создания нового типа полномочий выполните следующие действия:

1. В окне **Типы полномочий** (Credential Types) нажмите кнопку *Создать тип полномочий* (Create credential type).
2. Заполните форму **Создать тип полномочий** (Create credential type):
   - **Название** (Name) – укажите название типа полномочий.

     Название типа полномочий должно быть уникальным на уровне контроллера.
   - **Описание** (Description) – укажите описание типа полномочий, например, их назначение или особенности использования.
   - **Входная настройка** (Input configuration) – укажите параметры входных данных, которые необходимо указать при использовании этого типа полномочий.
   - **Конфигурация внедрения данных** (Injector configuration) – укажите данные, которые передаются во внешнюю систему аутентификации при использовании полномочий этого типа.

   Примечание

   Особенности и примеры заполнения полей **Настройка входных данных** (Input configuration) и **Конфигурация инжектора** (Injector configuration) см. в секции *Собственные типы полномочий*.
3. Нажмите кнопку *Создать тип полномочий* (Create credential type).

Удаление

Чтобы удалить тип полномочий, выполните следующие действия:

1. *Удалите* все полномочия соответствующего типа.
2. В окне **Типы полномочий** (Credential Types) включите флаги напротив удаляемых записей.
3. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите **Удалить типы полномочий** (Delete credential types).
4. Подтвердите удаление.

Администрирование

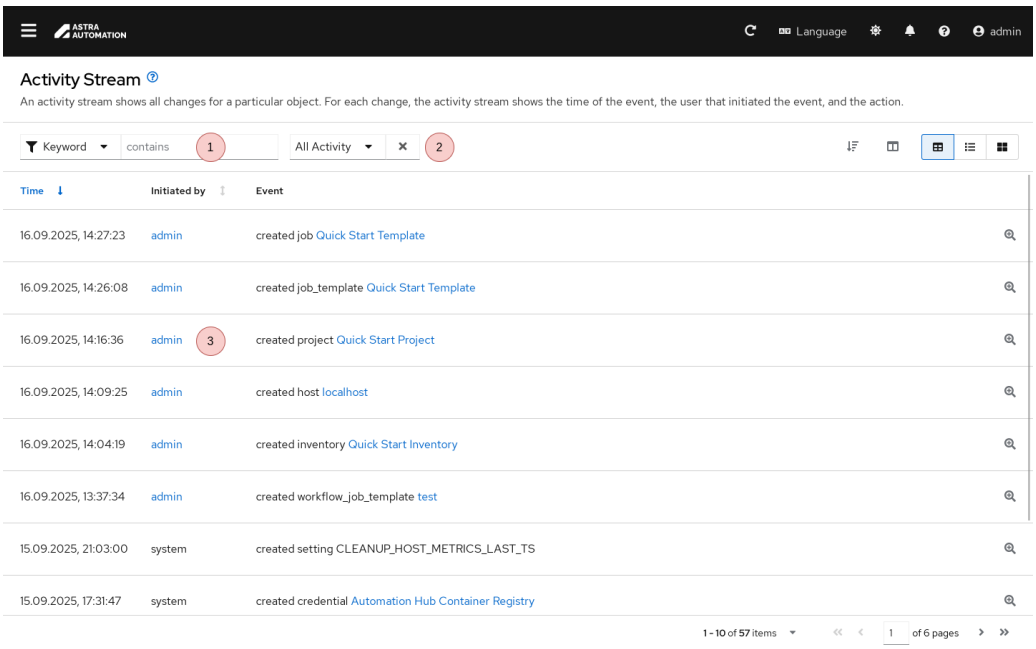
Раздел *Администрирование* (Administration) содержит подразделы, используемые для просмотра активности пользователей, согласования потоков заданий, управления способами отправки уведомлений и служебными заданиями.

Поток активности

Окно **Поток активности** (Activity Stream) используется для просмотра информации о действиях, выполненных внутри Automation Controller. Для перехода к нему выберите на панели навигации *Автоматизация процессов* ▸ *Администрирование* ▸ *Поток активности* (*Automation Execution* ▸ *Administration* ▸ *Activity Stream*).

Таблица событий

Внешний вид окна **Поток активности** (Activity Stream) представлен на схеме:



Окно **Поток активности** (Activity Stream) состоит из следующих основных элементов:

1. Поле поиска по строкам.

   Используется для фильтрации записей в таблице событий на основе вхождения в значение одного из полей указанной строки. Поддерживается создание сложных выражений фильтрации с использованием логических условий.

2. Поле фильтрации записей по типу связанных ресурсов или компонентов.

3. Таблица событий.

   Таблица событий состоит из следующих столбцов:

   - **Время** (Time) – дата и время события.
   - **Инициировано** (Initiated by) – инициатор события. Если инициатором события был один из пользователей контроллера, в этом столбце выводится ссылка на его профиль.
   - **Событие** (Event) – название события.

     Если событие связано с одним из ресурсов или компонентов контроллера (проект, шаблон, поток заданий и так далее), в этом поле выводится ссылка на него.

   - Кнопка просмотра подробной информации о событии.

     При нажатии на кнопку открывается диалоговое окно **Подробности о событии** (Event detail), в котором в дополнение ко времени, инициатору и типу события выводится список сделанных изменений.

Согласования потоков заданий

Окно **Согласования потоков заданий** (Workflow Approvals) используется для подтверждения или отмены действий, связанных с узлами типа «Согласование» (Approval) *шаблонов потоков заданий*. Для перехода к нему выберите на панели навигации *Автоматизация процессов* ▸ *Администрирование* ▸ *Согласования потоков заданий* (*Automation Execution* ▸ *Administration* ▸ *Workflow Approvals*).

Таблица согласования

Внешний вид окна **Согласования потоков заданий** (Workflow Approvals) представлен на схеме:

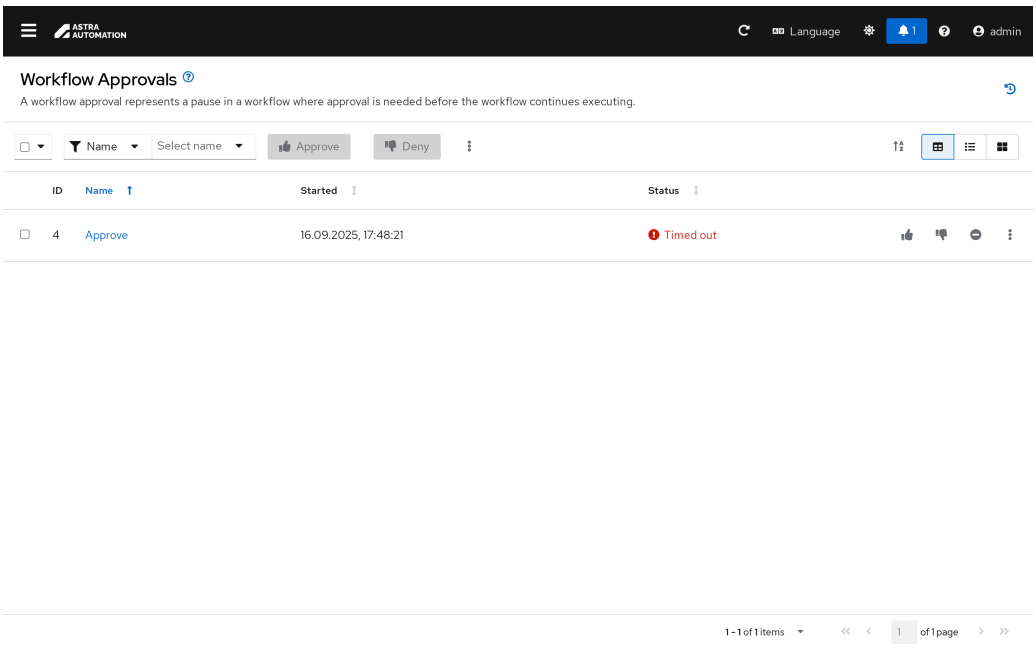


Таблица согласований состоит из следующих столбцов:

- Флаги для выбора нескольких записей.
- **Идентификатор** (ID) – уникальный идентификатор согласования.
- **Название** (Name) – ссылка на окно просмотра сведений о согласовании.
- **Начато** (Started) – дата и время запуска согласования.
- **Статус** (Status) – текущий статус согласования.

  Если время на согласование ограничено, для нерешенных согласований в этом столбце выводятся дата и время, до наступления которых необходимо принять решение.

  Для согласований, по которым решение принято, в этом столбце выводится их статус – «Согласовано» (Approved) и «Отказано» (Denied) соответственно.
- **Задание потока** (Workflow Job) – ссылка на окно просмотра сведений о задании потока, которое будет запущено при согласовании.
- Кнопки управления согласованиями:
  - принятие положительного решения по согласованию;
  - отказ в согласовании;
  - остановка выполнения потока заданий;
  - удаление согласования.

Просмотр

Для просмотра подробных сведений о согласовании нажмите на его название в таблице.

Окно подробных сведений о согласовании состоит из следующих вкладок:

- **Подробности** (Details), на которой выводится следующая информация:
  - **Идентификатор** (ID) – уникальный идентификатор согласования.
  - **Название** (Name) – название узла в графе потока заданий.
  - **Начато** (Started) – дата и время запуска согласования.
  - **Статус** (Status) – текущий статус согласования.
- **Подробности задания workflow** (Workflow Job Details), на которой выводится следующая информация:

- **Идентификатор** (ID) – уникальный идентификатор согласования.
- **Название** (Name) – ссылка на окно просмотра сведений о ходе выполнения согласования.
- **Статус** (Status) – текущий статус согласования.
- **Тип** (Type) – тип задания согласования.
- **Продолжительность** (Duration) – время, потраченное на согласование.
- **Начато** (Started) – дата и время запуска согласования.
- **Закончено** (Finished) – дата и время завершения выполнения.
- **Запущенный** (Launched by)– пользователь, инициировавший запуск.
- **Шаблон потока заданий** (Workflow job template) – ссылка на окно просмотра подробных сведений о шаблоне потока заданий.
- **Объяснение** (Explanation) – дополнительное описание, поясняющее причину завершения или ошибки.
- **Родитель фрагмента задания** (Job slice parent) – показывает, связано ли согласование с механизмом Job Slicing (разделение задания на несколько параллельных частей):
  - **Истина** (True) – текущее задание является родительским, из него были созданы срезы;
  - **Ложь** (False) – задание не связано с разделением на срезы;
- **Уровень подробности** (Verbosity) – уровень подробности вывода.
- **Дата создания** (Created) – дата и время создания согласования, а также ссылка на профиль связанного пользователя.
- **Последнее изменение** (Last modified) – дата и время последнего изменения согласования, а также ссылка на профиль связанного пользователя.
- **Дополнительные переменные** (Extra variables) – значения переменных Ansible, заданные на уровне потока заданий. Доступно отображение в формате YAML или JSON.

  Подробности о порядке присвоения значений переменным см. в документе *Переменные*.

Источники уведомлений

Окно **Источники уведомлений** (Notifiers) позволяет настроить Automation Controller на отправку уведомлений о различных событиях во внешние системы:

- Grafana;
- IRC-серверы;
- Mattermost;
- Pagerduty;
- Rocket.Chat;
- Slack;
- Twilio;
- Webhook;
- электронная почта.

Для перехода к окну **Источники уведомлений** (Notifiers) выберите на панели навигации *Автоматизация процессов* ▸ *Администрирование* ▸ *Источники уведомлений* (*Automation Execution* ▸ *Administration* ▸ *Notifiers*).

Таблица источников уведомлений

Внешний вид окна **Источники уведомлений** (Notifiers) представлен на схеме:

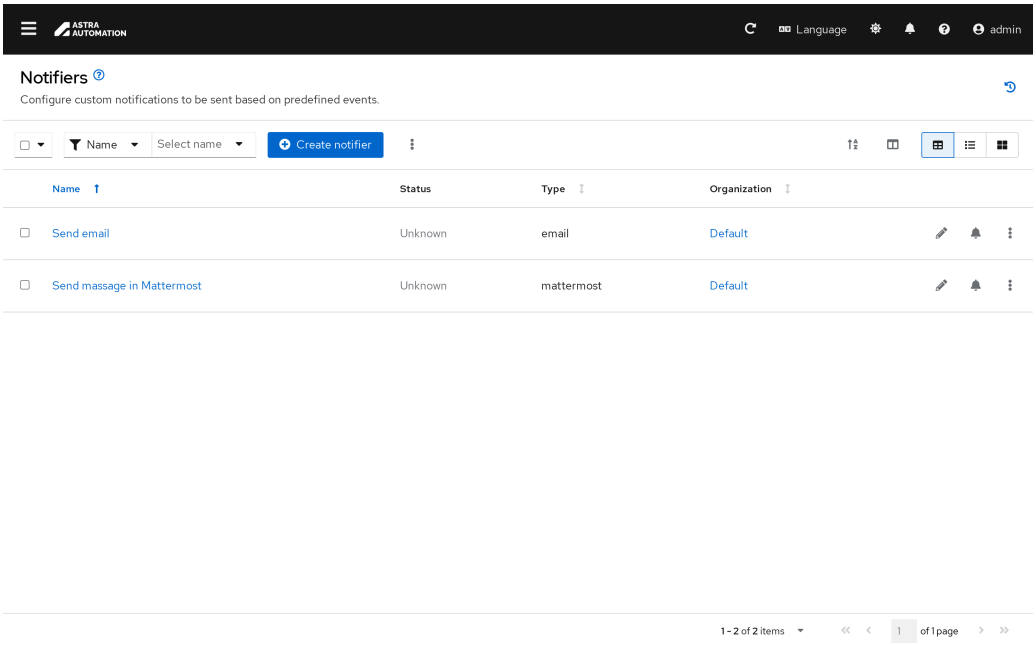


Таблица способов отправки уведомлений состоит из следующих столбцов:

- флаги для выбора нескольких записей;
- **Название** (Name);
- **Статус** (Status);
- **Тип** (Type) – тип внешней системы, в которую отправляются уведомления;
- **Организация** (Organization) – организация, которой принадлежит уведомление;
- кнопки для быстрого вызова часто выполняемых действий:
  - переход в окно редактирования свойств уведомления;
  - проверка корректности настроек уведомления;
  - копирование уведомления;
  - удаление уведомления.

Просмотр

Для получения подробных сведений об источнике уведомлений нажмите на его название в таблице.

Внешний вид окна со сведениями об источнике уведомлений зависит от типа внешней системы.

Создание

Для создания источника уведомлений выполните следующие действия:

1. В окне **Источники уведомлений** (Notifiers) нажмите кнопку *Создать источник уведомлений* (Create notifier).
2. Заполните форму **Создать источник уведомлений** (Create notifier):
   - **Название** (Name) – название источника уведомлений.
   - **Описание** (Description) – описание источника уведомлений.

- **Организация** (Organization) – организация, для которой настраивается источник уведомлений.
- **Тип** (Type) – тип создаваемого источника уведомлений.

  После выбора типа в форме отображаются соответствующие поля ввода.

Электронная почта (Email)

  - **Название учетной записи** (Username) – название учетной записи пользователя для доступа к почтовому серверу.
  - **Пароль** (Password) – пароль пользователя для доступа к почтовому серверу.
  - **Узел** (Host) – IP-адрес или *FQDN* почтового сервера.
  - **Список получателей** (Recipient list) – адреса электронной почты получателей, по одному на строку.
  - **Электронная почта отправителя** (Sender email) – адрес электронной почты, с которого следует отправлять письма.
  - **Порт** (Port) – номер порта для подключения к почтовому серверу.
  - **Тайм-аут** (Timeout) – период ожидания почтового сервера в секундах. Если в течение указанного времени контроллер не сможет связаться с почтовым сервером, отправка уведомления будет отменена.

    Минимальное значение – 1.

    Максимальное значение – 120.
  - **Использовать TLS** (Use TLS) – защита подключения к почтовому серверу с помощью TLS.
  - **Использовать SSL** (Use SSL) – защита подключения к почтовому серверу с помощью SSL.

Grafana

  - **URL для Grafana** (Grafana URL) – URL сервера Grafana без пути `/api/annotations/`. При создании уведомлений он добавляется к указанному URL автоматически.
  - **Ключ API Grafana** (Grafana API key) – ключ доступа к Grafana API.
  - **ID информационной панели** (ID of the dashboard) – идентификатор информационной панели Grafana для вывода уведомлений.
  - **ID панели** (ID of the panel) – идентификатор панели Grafana для вывода уведомлений.
  - **Теги для аннотации** (Tags for the annotation) – теги без команд, по одному на строку.
  - **Отключить проверку SSL** (Disable SSL verification) – должен ли контроллер проверять SSL-сертификат сервера Grafana.

IRC-сервер (IRC)

  - **Пароль IRC-сервера** (IRC server password) – пароль для подключения к серверу IRC.
  - **Порт IRC-сервера** (IRC server port) – номер порта для подключения к серверу IRC.

    Значение по умолчанию: 80.
  - **Адрес IRC-сервера** (IRC server address) – IP-адрес или *FQDN* сервера IRC.
  - **IRC ник** (IRC Nick) – название учетной записи пользователя для доступа к IRC.

- **Каналы или пользователи назначения** (Destination channels or users) – названия каналов и имена пользователей, которым должны быть отправлены сообщения с уведомлениями, по одному на строку.

  Важно

  При вводе названий каналов и имен пользователей не указывайте символы @ и # в начале строки.

- **Отключить проверку SSL** (Disable SSL verification) – должен ли контроллер проверять SSL-сертификат сервера IRC.

Mattermost

- **Целевой URL** (Target URL) – URL сервера Mattermost.
- **Название учетной записи** (Username) – название учетной записи пользователя для доступа к серверу Mattermost.
- **Канал** (Channel) – название канала Mattermost для отправки уведомлений.
- **URL значка** (Icon URL) – ссылка на иконку, используемую как аватар пользователя Mattermost, от имени которого отправляются сообщения.
- **Проверка SSL** (Verify SSL) – должен ли контроллер проверять сертификат SSL сервера Mattermost.

Pagerduty

- **Поддомен Pagerduty** (Pagerduty subdomain) – поддомен PagerDuty.
- **Токен API** (API token) – токен для подключения к PagerDuty.
- **Ключ API сервиса/интеграции** (API service/integration key) – ключ к сервису API или ключ интеграции.
- **Идентификатор клиента** (Client identifier) – идентификатор клиента, от имени которого контроллер будет отправлять уведомления в PagerDuty.

Rocket.Chat

- **Целевой URL** (Target URL) – *FQDN* для подключения к Rocket.Chat.
- **Название учетной записи** (Username) – название учетной записи пользователя для доступа к Rocket.Chat.
- **URL значка** (Icon URL) – ссылка на иконку, используемую как аватар пользователя, от имени которого отправляются сообщения.
- **Отключить проверку SSL** (Disable SSL verification) – должен ли контроллер проверять сертификат SSL сервера Rocket.Chat.

Slack

- **Токен** (Token) – токен, используемый для подключения к Slack.
- **Каналы назначения** (Destination channels) – названия каналов Slack для отправки в них уведомлений, по одному каналу на строку.

  Важно

  Название канала должно начинаться с символа #.

Чтобы ответить на определенное сообщение или запустить тред, укажите идентификатор родительского сообщения – строку из 16 цифр. После 10-й цифры добавьте символ `.`, например, `#dev-ops, 0000000555.111111`.

Подробности см. в документации Slack[189].

- **Цвет уведомления** (Notification color) – цвет уведомления в шестнадцатеричном формате, например, #12F или `#FF0000`.

Twilio

- **Идентификатор (SID) учетной записи** (Account SID) – идентификатор сеанса связи.
- **Токен учетной записи** (Account token) – токен для доступа к серверу Twilio.
- **Исходный номер телефона** (Source phone number) – номер телефона, с которого отправляются сообщения.

> **Важно**
>
> Допускается использование цифр и символа + в начале строки.

- **Номера SMS назначения** (Destination SMS numbers) – номера телефонов для отправки на них SMS с уведомлениями, по одному на строку.

> **Важно**
>
> Допускается использование цифр и символа + в начале строки.

Подробности см. в документации Twilio[190].

Webhook

- **Название учетной записи** (Username) – название учетной записи пользователя для доступа к сервису WebHook.
- **Базовый пароль для авторизации** (Basic auth password) – пароль пользователя для доступа к сервису WebHook.
- **Целевой URL** (Target URL) – URL WebHook.
- **Отключить проверку SSL** (Disable SSL verification) – должен ли контроллер проверять сертификат SSL сервера WebHook.
- **HTTP-заголовки** (HTTP Headers) – дополнительные заголовки HTTP в формате JSON, которые должны быть добавлены в запрос.
- **HTTP-метод** (HTTP Method) – метод HTTP, используемый для отправки запросов к WebHook. Поддерживаются методы `POST` и `PUT`.

- **Настроить сообщения** (Customize messages) – если этот переключатель включен, в форме отображаются дополнительные поля:

[189] https://api.slack.com/messaging/retrieving#individual_messages
[190] https://www.twilio.com/docs/messaging

Поле	Событие
Стартовое сообщение (Start message)	Запуск задания
Сообщение об успехе (Success message)	Успешное выполнение задания
Текст ошибки (Error message)	Ошибка при выполнении задания
Сообщение об утвержденном workflow (Workflow approved message)	Согласование этапа потока заданий
Сообщение об отклоненном workflow (Workflow denied message)	Отклонение этапа потока заданий
Ожидающее сообщение workflow (Workflow pending message)	Ожидание согласования этапа потока заданий
Сообщение о превышении времени ожидания workflow (Workflow timed out message)	Истечение времени ожидания согласования этапа потока заданий

Примечание

Подробности о синтаксисе шаблонов и доступных переменных см. в документе *Отправка уведомлений в Mattermost*.

3. Нажмите кнопку *Сохранить источник уведомлений* (Save notifier).

Удаление

Чтобы удалить источники уведомлений, выполните следующие действия:

1. В таблице включите флаги напротив удаляемых источников уведомлений.
2. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите **Удалить источники уведомлений** (Delete notifiers).
3. Подтвердите удаление.

Служебные задания

Окно **Служебные задания** (Management jobs) предоставляет интерфейс для управления *системными операциями платформы*:

- запуск заданий очистки журналов и временных данных;
- инициация синхронизации проектов и инвентарей;
- выполнение и обслуживание платформы без обращения к CLI.

Для перехода к окну **Служебные задания** (Management jobs) выберите на панели навигации *Автоматизация процессов* ▸ *Администрирование* ▸ *Служебные задания* (*Automation Execution* ▸ *Administration* ▸ *Management Jobs*).

Таблица служебных заданий

Внешний вид окна **Служебные задания** (Management jobs) представлен на схеме:

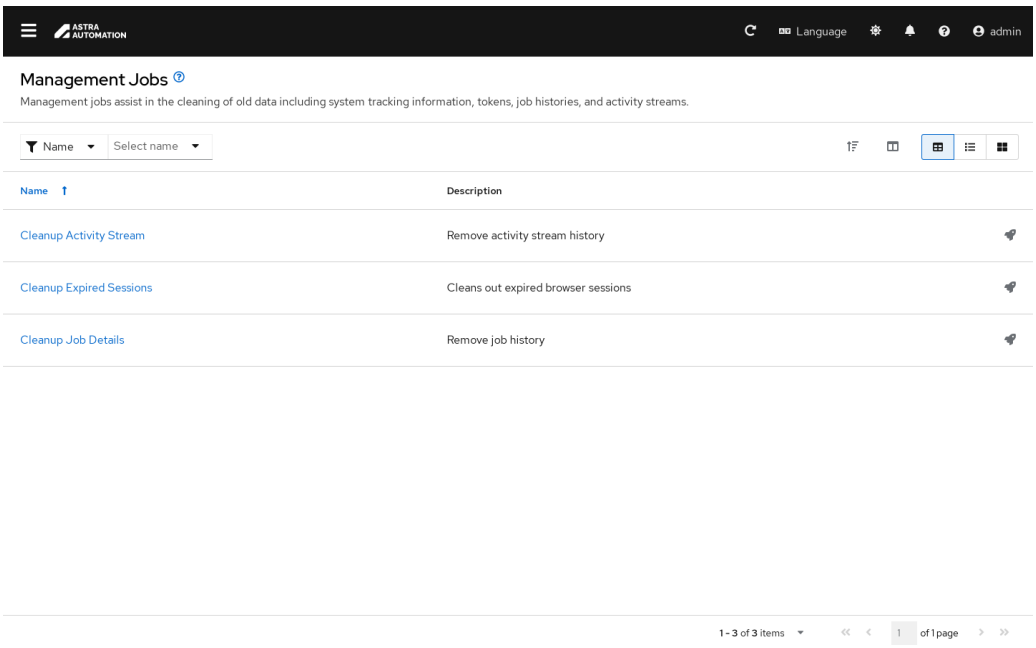


Таблица служебных заданий состоит из следующих столбцов:

- **Название** (Name) – ссылка для перехода в окно просмотра расписаний, связанных с заданием;
- **Описание** (Description);
- кнопка запуска задания.

Запуск

Для запуска служебного задания нажмите кнопку *Запустить управляющее задание* (Launch management job).

Для заданий типа «Cleanup Activity Stream» и «Cleanup Job Details» при этом открывается диалоговое окно **Запустить управляющее задание** (Launch management job).

1. Укажите период хранения записей (количество дней). Все записи старше указанного периода будут удалены.

   Значение по умолчанию: `30`.

2. Нажмите кнопку *Запустить* (Launch).

Настройка расписания

При развертывании контроллера расписание запуска служебных заданий настраивается автоматически.

Чтобы изменить расписание запуска служебного задания, выполните следующие действия:

1. В таблице нажмите на название служебного задания, расписание запуска которого необходимо изменить.
2. Во вкладке **Расписания** (Schedules) нажмите на название существующего расписания.
3. Во вкладке **Подробности** (Details) нажмите кнопку *Редактировать расписание* (Edit schedule).
4. Измените значения в окне **Редактировать <название_расписания>** (Edit <schedule_name>) на необходимые. Описание полей см. в документе *Расписания*.

Внимание

Для заданий «Cleanup Activity Stream» и «Cleanup Job Details» доступна дополнительная настройка **Количество дней хранения данных** (Days of data to keep).

Это период (количество дней), в течение которого должны храниться данные о действиях пользователей и выполнении заданий соответственно. Все записи, созданные ранее этого периода, будут удалены.

Значения по умолчанию:

- Cleanup Activity Schedule – 355;
- Cleanup Job Schedule – 120.

11.1.4 API

API системы автоматизации процессов представляет собой RESTful API, предоставляющий программный доступ ко всем функциям Automation Controller — центрального компонента Astra Automation. API позволяет организациям централизовать и контролировать свою автоматизацию не только через визуальную панель управления, но и путем глубокой интеграции с другими инструментами и процессами. Он предоставляет несколько методов авторизации, включая session authentication, basic authentication, OAuth 2.0 token authentication и single sign-on, что обеспечивает гибкость и безопасность при встраивании платформы в существующие корпоративные системы и рабочие процессы. Доступ к API осуществляется через множество точек доступа (endpoints), начиная с `/api/controller/v2/`, и включает полную поддержку методов GET, POST, PUT, PATCH и DELETE для управления ресурсами, такими как inventory, credentials, projects, job templates и workflows.

Базовый сценарий

В этом документе приведен базовый сценарий использования API Automation Controller. В ходе выполнения сценария производятся следующие действия:

1. Получение токена авторизации.
2. Создание организации.
3. Добавление полномочий Ansible Galaxy для созданной организации.
4. Создание инвентаря.
5. Добавление управляемого узла в инвентарь.
6. Создание проекта.
7. Создание шаблона задания для установки NGINX.
8. Запуск шаблона задания.
9. Проверка статуса выполнения задания.

Предварительные требования

Перед выполнением сценария произведите следующие действия в графическом интерфейсе контроллера:

1. Создайте полномочия для доступа к следующим ресурсам:
   - управляемый узел;
   - Ansible Galaxy;
   - источник проектов.

Примечание

Управление полномочиями объясняется в *описании графического интерфейса*.

2. Получите идентификаторы созданных полномочий:
   a. На панели навигации выберите *Автоматизация процессов* ▸ *Инфраструктура* ▸ *Полномочия* (*Automation Execution* ▸ *Infrastructure* ▸ *Credentials*).
   b. Нажмите на ссылку с названием нужного полномочия, например, Ansible Galaxy.
   c. Скопируйте <credentials_id> в адресной строке браузера, которая имеет следующий вид `https://<address>/#/credentials/<credentials_id>/details`.

Подготовка организационной структуры

Для подготовки организационной структуры выполните следующие действия:

1. *Получите токен авторизации*.
2. Получите идентификатор среды исполнения в графическом интерфейсе контроллера:
   a. На панели навигации выберите *Автоматизация процессов* ▸ *Инфраструктура* ▸ *Среды исполнения* (*Automation Execution* ▸ *Infrastructure* ▸ *Execution Environments*).
   b. Нажмите на ссылку с названием нужной среды исполнения, например, Default execution environment.
   c. Скопируйте <execution_environment_id> в адресной строке браузера, которая имеет следующий вид `https://<address>/execution/infrastructure/execution-environments/<execution_environments_id>/details`.
3. Создайте организацию:

```
curl -X POST \
   -H 'Content-Type: application/json' \
   -H 'Authorization: Bearer <your_token>' \
   -d '{
            "name": "<your_organization_name>",
            "default_environment": "<execution_environment_id>"
       }' \
   https://<address>/api/gateway/v1/organizations/ | jq .
```

Подставьте собственные значения параметров вместо употребленных в примере:

- <address> – FQDN или IP-адрес контроллера, например, `192.168.56.11`;
- <your_token> – токен, полученный на предыдущем шаге, например, `3wijiG7********VtL2gseJESzjJsD`;
- <your_organization_name> – название организации, например, `Jupiter`;
- <execution_environment_id> – идентификатор среды исполнения, например, 2.

Organization


```
{
  "id": 4,
  "url": "/api/gateway/v1/organizations/4/",
  "related": {
    "activity_stream": "/api/gateway/v1/activitystream/?content_type=3&object_id=4",
    "created_by": "/api/gateway/v1/users/2/",
    "modified_by": "/api/gateway/v1/users/2/",
    "teams": "/api/gateway/v1/organizations/4/teams/",
    "users": "/api/gateway/v1/organizations/4/users/",
    "admins": "/api/gateway/v1/organizations/4/admins/"
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
  },
  "summary_fields": {
    "modified_by": {
      "id": 2,
      "username": "admin",
      "first_name": "",
      "last_name": ""
    },
    "created_by": {
      "id": 2,
      "username": "admin",
      "first_name": "",
      "last_name": ""
    },
    "resource": {
      "ansible_id": "3db59f6d-1f89-452e-94b6-4fa2adaa811c",
      "resource_type": "shared.organization"
    },
    "related_field_counts": {
      "teams": 0,
      "users": 0
    }
  },
  "created": "2025-11-25T09:08:20.772225Z",
  "created_by": 2,
  "modified": "2025-11-25T09:08:20.772144Z",
  "modified_by": 2,
  "name": "Jupiter",
  "description": "",
  "managed": false
}
```


4. Добавьте полномочия Ansible Galaxy для созданной организации:

```
curl -X POST \
   -H 'Content-Type: application/json' \
   -H 'Authorization: Bearer <your_token>' \
   -d '{
            "id":<credentials_id>
       }' \
   https://<address>/api/controller/v2/organizations/<organization_id>/galaxy_
↪credentials/ | jq .
```

Подставьте собственные значения параметров вместо употребленных в примере:

- <credentials_id> – идентификатор полномочий для доступа к Ansible Galaxy, например, 2;
- <organization_id> – идентификатор организации, созданной ранее, например, 4.

Подготовка инвентаря

Для подготовки инвентаря выполните следующие действия:

1. Создайте инвентарь:

```
curl -X POST \
   -H 'Content-Type: application/json' \
   -H 'Authorization: Bearer <address>' \
   -d '{
            "name": "<your_inventory_name>",
            "organization": <organization_id>
       }' \
   https://<address>/api/controller/v2/inventories/ | jq .
```

Подставьте собственные значения параметров вместо употребленных в примере. Здесь <your_inventory_name> – название инвентаря, например, `Demo Inventory`.

Inventory


```
{
  "id": 3,
  "type": "inventory",
  "url": "/api/controller/v2/inventories/3/",
  "related": {
    "named_url": "/api/controller/v2/inventories/Demo Inventory++Jupiter/",
    "created_by": "/api/controller/v2/users/1/",
    "modified_by": "/api/controller/v2/users/1/",
    "hosts": "/api/controller/v2/inventories/3/hosts/",
    "variable_data": "/api/controller/v2/inventories/3/variable_data/",
    "script": "/api/controller/v2/inventories/3/script/",
    "activity_stream": "/api/controller/v2/inventories/3/activity_stream/",
    "job_templates": "/api/controller/v2/inventories/3/job_templates/",
    "ad_hoc_commands": "/api/controller/v2/inventories/3/ad_hoc_commands/",
    "access_list": "/api/controller/v2/inventories/3/access_list/",
    "object_roles": "/api/controller/v2/inventories/3/object_roles/",
    "instance_groups": "/api/controller/v2/inventories/3/instance_groups/",
    "copy": "/api/controller/v2/inventories/3/copy/",
    "labels": "/api/controller/v2/inventories/3/labels/",
    "groups": "/api/controller/v2/inventories/3/groups/",
    "root_groups": "/api/controller/v2/inventories/3/root_groups/",
    "update_inventory_sources": "/api/controller/v2/inventories/3/update_inventory_
↪sources/",
    "inventory_sources": "/api/controller/v2/inventories/3/inventory_sources/",
    "tree": "/api/controller/v2/inventories/3/tree/",
    "organization": "/api/controller/v2/organizations/4/"
  },
  "summary_fields": {
    "organization": {
      "id": 4,
      "name": "Jupiter",
      "description": ""
    },
    "created_by": {
      "id": 1,
      "username": "admin",
      "first_name": "",
      "last_name": ""
    },
    "modified_by": {
      "id": 1,
      "username": "admin",
      "first_name": "",
      "last_name": ""
    },
    "object_roles": {
      "admin_role": {
        "description": "Can manage all aspects of the inventory",
        "name": "Admin",
        "id": 123
      },
      "update_role": {
        "description": "May update the inventory",
        "name": "Update",
        "id": 124
      },
      "adhoc_role": {
        "description": "May run ad hoc commands on the inventory",
        "name": "Ad Hoc",
        "id": 125
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
      },
      "use_role": {
        "description": "Can use the inventory in a job template",
        "name": "Use",
        "id": 126
      },
      "read_role": {
        "description": "May view settings for the inventory",
        "name": "Read",
        "id": 127
      }
    },
    "user_capabilities": {
      "edit": true,
      "delete": true,
      "copy": true,
      "adhoc": true
    },
    "labels": {
      "count": 0,
      "results": []
    }
  },
  "created": "2025-11-25T09:47:53.058074Z",
  "modified": "2025-11-25T09:47:53.058086Z",
  "name": "Demo Inventory",
  "description": "",
  "organization": 4,
  "kind": "",
  "host_filter": null,
  "variables": "",
  "has_active_failures": false,
  "total_hosts": 0,
  "hosts_with_active_failures": 0,
  "total_groups": 0,
  "has_inventory_sources": false,
  "total_inventory_sources": 0,
  "inventory_sources_with_failures": 0,
  "pending_deletion": false,
  "prevent_instance_group_fallback": false
}
```


2. Добавьте управляемый узел в инвентарь:

```
curl -X POST \
   -H 'Content-Type: application/json' \
   -H 'Authorization: Bearer <your_token>' \
   -d '{
            "name": "<node_ip>",
            "description": "<node_name>",
            "inventory": <inventory_id>,
            "enabled": true
       }' \
   https://<address>/api/controller/v2/hosts/ | jq .
```

Подставьте собственные значения параметров вместо употребленных в примере:

- <node_ip> – IP-адрес управляемого узла например, `10.2.0.6`;
- <node_name> – название управляемого узла, например, `node01`;
- <inventory_id> – идентификатор инвентаря, например, 3.

Host


```
{
  "id": 3,
  "type": "host",
  "url": "/api/controller/v2/hosts/3/",
  "related": {
    "named_url": "/api/controller/v2/hosts/10.2.0.6++Demo Inventory++Jupiter/",
    "created_by": "/api/controller/v2/users/1/",
    "modified_by": "/api/controller/v2/users/1/",
    "variable_data": "/api/controller/v2/hosts/3/variable_data/",
    "groups": "/api/controller/v2/hosts/3/groups/",
    "all_groups": "/api/controller/v2/hosts/3/all_groups/",
    "job_events": "/api/controller/v2/hosts/3/job_events/",
    "job_host_summaries": "/api/controller/v2/hosts/3/job_host_summaries/",
    "activity_stream": "/api/controller/v2/hosts/3/activity_stream/",
    "inventory_sources": "/api/controller/v2/hosts/3/inventory_sources/",
    "smart_inventories": "/api/controller/v2/hosts/3/smart_inventories/",
    "ad_hoc_commands": "/api/controller/v2/hosts/3/ad_hoc_commands/",
    "ad_hoc_command_events": "/api/controller/v2/hosts/3/ad_hoc_command_events/",
    "ansible_facts": "/api/controller/v2/hosts/3/ansible_facts/",
    "inventory": "/api/controller/v2/inventories/3/"
  },
  "summary_fields": {
    "inventory": {
      "id": 3,
      "name": "Demo Inventory",
      "description": "",
      "has_active_failures": false,
      "total_hosts": 0,
      "hosts_with_active_failures": 0,
      "total_groups": 0,
      "has_inventory_sources": false,
      "total_inventory_sources": 0,
      "inventory_sources_with_failures": 0,
      "organization_id": 4,
      "kind": ""
    },
    "created_by": {
      "id": 1,
      "username": "admin",
      "first_name": "",
      "last_name": ""
    },
    "modified_by": {
      "id": 1,
      "username": "admin",
      "first_name": "",
      "last_name": ""
    },
    "user_capabilities": {
      "edit": true,
      "delete": true
    },
    "groups": {
      "count": 0,
      "results": []
    },
    "recent_jobs": []
  },
  "created": "2025-11-25T09:54:32.429608Z",
  "modified": "2025-11-25T09:54:32.429628Z",
  "name": "10.2.0.6",
  "description": "node01",
  "inventory": 3,
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
  "enabled": true,
  "instance_id": "",
  "variables": "",
  "has_active_failures": false,
  "has_inventory_sources": false,
  "last_job": null,
  "last_job_host_summary": null,
  "ansible_facts_modified": null
}
```


Подготовка задания

Для подготовки задания выполните следующие действия:

1. Создайте проект:

```
curl -X POST \
   -H 'Content-Type: application/json' \
   -H 'Authorization: Bearer <your_token>' \
   -d '{
            "name": "<your_project_name>",
            "scm_type": "git",
            "scm_url": "https://github.com/ansible/ansible-tower-samples",
            "scm_branch": "master",
            "default_environment": <environment_id>,
            "credential": <credential_id>,
            "organization": <organization_id>
       }' \
   https://<address>/api/controller/v2/projects/ | jq .
```

Подставьте собственные значения параметров вместо употребленных в примере:

- <your_project_name> – название проекта, например, `Demo Project`;
- <credentials_id> – идентификатор полномочий для доступа к источнику проекта, если такие полномочия необходимы.

Project


```
{
  "id": 12,
  "type": "project",
  "url": "/api/controller/v2/projects/12/",
  "related": {
    "named_url": "/api/controller/v2/projects/Demo Project++Jupiter/",
    "created_by": "/api/controller/v2/users/1/",
    "modified_by": "/api/controller/v2/users/1/",
    "credential": "/api/controller/v2/credentials/4/",
    "current_job": "/api/controller/v2/project_updates/19/",
    "teams": "/api/controller/v2/projects/12/teams/",
    "playbooks": "/api/controller/v2/projects/12/playbooks/",
    "inventory_files": "/api/controller/v2/projects/12/inventories/",
    "update": "/api/controller/v2/projects/12/update/",
    "project_updates": "/api/controller/v2/projects/12/project_updates/",
    "scm_inventory_sources": "/api/controller/v2/projects/12/scm_inventory_sources/
↪",
    "schedules": "/api/controller/v2/projects/12/schedules/",
    "activity_stream": "/api/controller/v2/projects/12/activity_stream/",
    "notification_templates_started": "/api/controller/v2/projects/12/notification_
↪templates_started/",
    "notification_templates_success": "/api/controller/v2/projects/12/notification_
↪templates_success/",
    "notification_templates_error": "/api/controller/v2/projects/12/notification_
↪templates_error/",
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
    "access_list": "/api/controller/v2/projects/12/access_list/",
    "object_roles": "/api/controller/v2/projects/12/object_roles/",
    "copy": "/api/controller/v2/projects/12/copy/",
    "organization": "/api/controller/v2/organizations/4/",
    "default_environment": "/api/controller/v2/execution_environments/2/",
    "current_update": "/api/controller/v2/project_updates/19/"
  },
  "summary_fields": {
    "organization": {
      "id": 4,
      "name": "Jupiter",
      "description": ""
    },
    "default_environment": {
      "id": 2,
      "name": "Default execution environment",
      "description": "",
      "image": "hub.astra-automation.ru/aa-1.2/aa-full-ee:latest"
    },
    "credential": {
      "id": 4,
      "name": "SCM access",
      "description": "",
      "kind": "scm",
      "cloud": false,
      "kubernetes": false,
      "credential_type_id": 8
    },
    "current_update": {
      "id": 19,
      "name": "Demo Project",
      "description": "",
      "status": "pending",
      "failed": false
    },
    "current_job": {
      "id": 19,
      "name": "Demo Project",
      "description": "",
      "status": "pending",
      "failed": false
    },
    "created_by": {
      "id": 1,
      "username": "admin",
      "first_name": "",
      "last_name": ""
    },
    "modified_by": {
      "id": 1,
      "username": "admin",
      "first_name": "",
      "last_name": ""
    },
    "object_roles": {
      "admin_role": {
        "description": "Can manage all aspects of the project",
        "name": "Admin",
        "id": 128
      },
      "use_role": {
        "description": "Can use the project in a job template",
        "name": "Use",
        "id": 129
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
      },
      "update_role": {
        "description": "May update the project",
        "name": "Update",
        "id": 130
      },
      "read_role": {
        "description": "May view settings for the project",
        "name": "Read",
        "id": 131
      }
    },
    "user_capabilities": {
      "edit": true,
      "delete": true,
      "start": true,
      "schedule": true,
      "copy": true
    }
  },
  "created": "2025-11-25T10:00:14.421954Z",
  "modified": "2025-11-25T10:00:14.421985Z",
  "name": "Demo Project",
  "description": "",
  "local_path": "_12__demo_nginx_project",
  "scm_type": "git",
  "scm_url": "https://github.com/ansible/ansible-tower-samples",
  "scm_branch": "master",
  "scm_refspec": "",
  "scm_clean": false,
  "scm_track_submodules": false,
  "scm_delete_on_update": false,
  "credential": 4,
  "timeout": 0,
  "scm_revision": "",
  "last_job_run": null,
  "last_job_failed": false,
  "next_job_run": null,
  "status": "pending",
  "organization": 4,
  "scm_update_on_launch": false,
  "scm_update_cache_timeout": 0,
  "allow_override": false,
  "custom_virtualenv": null,
  "default_environment": 2,
  "signature_validation_credential": null,
  "last_update_failed": false,
  "last_updated": null
}
```


2. Создайте шаблон задания:

```
curl -X POST \
   -H 'Content-Type: application/json' \
   -H 'Authorization: Bearer <your_token>' \
   -d '{
            "name": "<job_templates_name>",
            "job_type": "run",
            "inventory": <inventory_id>,
            "project": <project_id>,
            "playbook": "<playbook_name>"
       }' \
   https://<address>/api/controller/v2/job_templates/ | jq .
```

Подставьте собственные значения параметров вместо употребленных в примере.

Здесь

- <job_templates_name> – название шаблона задания, например, Demo Job templates;
- <project_id> – идентификатор проекта, созданного ранее, например, 12;
- <playbook_name> – название используемого набора сценариев, например, hello_world.yml.

Job template


```
{
  "id": 14,
  "type": "job_template",
  "url": "/api/controller/v2/job_templates/14/",
  "related": {
    "named_url": "/api/controller/v2/job_templates/Demo Job templates++Jupiter/",
    "created_by": "/api/controller/v2/users/1/",
    "modified_by": "/api/controller/v2/users/1/",
    "labels": "/api/controller/v2/job_templates/14/labels/",
    "inventory": "/api/controller/v2/inventories/3/",
    "project": "/api/controller/v2/projects/12/",
    "organization": "/api/controller/v2/organizations/4/",
    "credentials": "/api/controller/v2/job_templates/14/credentials/",
    "jobs": "/api/controller/v2/job_templates/14/jobs/",
    "schedules": "/api/controller/v2/job_templates/14/schedules/",
    "activity_stream": "/api/controller/v2/job_templates/14/activity_stream/",
    "launch": "/api/controller/v2/job_templates/14/launch/",
    "webhook_key": "/api/controller/v2/job_templates/14/webhook_key/",
    "webhook_receiver": "",
    "notification_templates_started": "/api/controller/v2/job_templates/14/
↪notification_templates_started/",
    "notification_templates_success": "/api/controller/v2/job_templates/14/
↪notification_templates_success/",
    "notification_templates_error": "/api/controller/v2/job_templates/14/
↪notification_templates_error/",
    "access_list": "/api/controller/v2/job_templates/14/access_list/",
    "survey_spec": "/api/controller/v2/job_templates/14/survey_spec/",
    "object_roles": "/api/controller/v2/job_templates/14/object_roles/",
    "instance_groups": "/api/controller/v2/job_templates/14/instance_groups/",
    "slice_workflow_jobs": "/api/controller/v2/job_templates/14/slice_workflow_jobs/
↪",
    "copy": "/api/controller/v2/job_templates/14/copy/"
  },
  "summary_fields": {
    "organization": {
      "id": 4,
      "name": "Jupiter",
      "description": ""
    },
    "inventory": {
      "id": 3,
      "name": "Demo Inventory",
      "description": "",
      "has_active_failures": false,
      "total_hosts": 1,
      "hosts_with_active_failures": 0,
      "total_groups": 0,
      "has_inventory_sources": false,
      "total_inventory_sources": 0,
      "inventory_sources_with_failures": 0,
      "organization_id": 4,
      "kind": ""
    },
    "project": {
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
      "id": 12,
      "name": "Demo Project",
      "description": "",
      "status": "successful",
      "scm_type": "git",
      "allow_override": false
    },
    "created_by": {
      "id": 1,
      "username": "admin",
      "first_name": "",
      "last_name": ""
    },
    "modified_by": {
      "id": 1,
      "username": "admin",
      "first_name": "",
      "last_name": ""
    },
    "object_roles": {
      "admin_role": {
        "description": "Can manage all aspects of the job template",
        "name": "Admin",
        "id": 138
      },
      "execute_role": {
        "description": "May run the job template",
        "name": "Execute",
        "id": 139
      },
      "read_role": {
        "description": "May view settings for the job template",
        "name": "Read",
        "id": 140
      }
    },
    "user_capabilities": {
      "edit": true,
      "delete": true,
      "start": true,
      "schedule": true,
      "copy": true
    },
    "labels": {
      "count": 0,
      "results": []
    },
    "recent_jobs": [],
    "credentials": []
  },
  "created": "2025-11-25T11:26:03.595428Z",
  "modified": "2025-11-25T11:26:03.595453Z",
  "name": "Demo Job templates",
  "description": "",
  "job_type": "run",
  "inventory": 3,
  "project": 12,
  "playbook": "hello_world.yml",
  "scm_branch": "",
  "forks": 0,
  "limit": "",
  "verbosity": 0,
  "extra_vars": "",
  "job_tags": "",
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
  "force_handlers": false,
  "skip_tags": "",
  "start_at_task": "",
  "timeout": 0,
  "use_fact_cache": false,
  "organization": 4,
  "last_job_run": null,
  "last_job_failed": false,
  "next_job_run": null,
  "status": "never updated",
  "execution_environment": null,
  "host_config_key": "",
  "ask_scm_branch_on_launch": false,
  "ask_diff_mode_on_launch": false,
  "ask_variables_on_launch": false,
  "ask_limit_on_launch": false,
  "ask_tags_on_launch": false,
  "ask_skip_tags_on_launch": false,
  "ask_job_type_on_launch": false,
  "ask_verbosity_on_launch": false,
  "ask_inventory_on_launch": false,
  "ask_credential_on_launch": false,
  "ask_execution_environment_on_launch": false,
  "ask_labels_on_launch": false,
  "ask_forks_on_launch": false,
  "ask_job_slice_count_on_launch": false,
  "ask_timeout_on_launch": false,
  "ask_instance_groups_on_launch": false,
  "survey_enabled": false,
  "become_enabled": false,
  "diff_mode": false,
  "allow_simultaneous": false,
  "custom_virtualenv": null,
  "job_slice_count": 1,
  "webhook_service": "",
  "webhook_credential": null,
  "prevent_instance_group_fallback": false
}
```


3. Добавьте полномочия для шаблона задания:

```
curl -X POST \
   -H 'Content-Type: application/json' \
   -H 'Authorization: Bearer <your_token>' \
   -d '{
            "id":<credential_id>
       }' \
   https://<address>/api/controller/v2/job_templates/<job_template_id>/credentials/␣
↪| jq .
```

Подставьте собственные значения параметров вместо употребленных в примере:

- <credentials_id> – идентификатор полномочий для доступа к управляемому узлу, например, 1;
- <job_template_id> – идентификатор созданного ранее шаблона задания, например, 14.

Выполнение и проверка задания

Для запуска и проверки статуса задания выполните следующие действия:

1. Запустите задание, используя созданный шаблон:

```
curl -X POST \
   -H 'Authorization: Bearer <your_token>' \
   https://<address>/api/controller/v2/job_templates/<job_template_id>/launch/ | jq␣
↪.
```

Job template


```
{
  "job": 25,
  "ignored_fields": {},
  "id": 25,
  "type": "job",
  "url": "/api/controller/v2/jobs/25/",
  "related": {
    "created_by": "/api/controller/v2/users/1/",
    "modified_by": "/api/controller/v2/users/1/",
    "labels": "/api/controller/v2/jobs/25/labels/",
    "inventory": "/api/controller/v2/inventories/3/",
    "project": "/api/controller/v2/projects/12/",
    "organization": "/api/controller/v2/organizations/4/",
    "credentials": "/api/controller/v2/jobs/25/credentials/",
    "unified_job_template": "/api/controller/v2/job_templates/14/",
    "stdout": "/api/controller/v2/jobs/25/stdout/",
    "job_events": "/api/controller/v2/jobs/25/job_events/",
    "job_host_summaries": "/api/controller/v2/jobs/25/job_host_summaries/",
    "activity_stream": "/api/controller/v2/jobs/25/activity_stream/",
    "notifications": "/api/controller/v2/jobs/25/notifications/",
    "create_schedule": "/api/controller/v2/jobs/25/create_schedule/",
    "job_template": "/api/controller/v2/job_templates/14/",
    "cancel": "/api/controller/v2/jobs/25/cancel/",
    "relaunch": "/api/controller/v2/jobs/25/relaunch/"
  },
  "summary_fields": {
    "organization": {
      "id": 4,
      "name": "Jupiter",
      "description": ""
    },
    "inventory": {
      "id": 3,
      "name": "Demo Inventor",
      "description": "",
      "has_active_failures": false,
      "total_hosts": 1,
      "hosts_with_active_failures": 0,
      "total_groups": 0,
      "has_inventory_sources": false,
      "total_inventory_sources": 0,
      "inventory_sources_with_failures": 0,
      "organization_id": 4,
      "kind": ""
    },
    "project": {
      "id": 12,
      "name": "Demo Project",
      "description": "",
      "status": "successful",
      "scm_type": "git",
      "allow_override": false
    },
    "job_template": {
      "id": 14,
      "name": "Demo Job templates",
      "description": ""
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
    },
    "unified_job_template": {
      "id": 14,
      "name": "Demo Job templates",
      "description": "",
      "unified_job_type": "job"
    },
    "created_by": {
      "id": 1,
      "username": "admin",
      "first_name": "",
      "last_name": ""
    },
    "modified_by": {
      "id": 1,
      "username": "admin",
      "first_name": "",
      "last_name": ""
    },
    "user_capabilities": {
      "delete": true,
      "start": true
    },
    "labels": {
      "count": 0,
      "results": []
    },
    "credentials": [
      {
        "id": 6,
        "name": "Demo machine credential",
        "description": "",
        "kind": "ssh",
        "cloud": false
      }
    ]
  },
  "created": "2025-11-25T11:42:23.485635Z",
  "modified": "2025-11-25T11:42:23.580598Z",
  "name": "Demo Job templates",
  "description": "",
  "job_type": "run",
  "inventory": 3,
  "project": 12,
  "playbook": "hello_world.yml",
  "scm_branch": "",
  "forks": 0,
  "limit": "",
  "verbosity": 0,
  "extra_vars": "{}",
  "job_tags": "",
  "force_handlers": false,
  "skip_tags": "",
  "start_at_task": "",
  "timeout": 0,
  "use_fact_cache": false,
  "organization": 4,
  "unified_job_template": 14,
  "launch_type": "manual",
  "status": "pending",
  "execution_environment": null,
  "failed": false,
  "started": null,
  "finished": null,
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
  "canceled_on": null,
  "elapsed": 0,
  "job_args": "",
  "job_cwd": "",
  "job_env": {},
  "job_explanation": "",
  "execution_node": "",
  "controller_node": "",
  "result_traceback": "",
  "event_processing_finished": false,
  "launched_by": {
    "id": 1,
    "name": "admin",
    "type": "user",
    "url": "/api/controller/v2/users/1/"
  },
  "work_unit_id": null,
  "job_template": 14,
  "passwords_needed_to_start": [],
  "allow_simultaneous": false,
  "artifacts": {},
  "scm_revision": "",
  "instance_group": null,
  "diff_mode": false,
  "job_slice_number": 0,
  "job_slice_count": 1,
  "webhook_service": "",
  "webhook_credential": null,
  "webhook_guid": ""
}
```


2. Проверьте статус выполнения задания:

```
curl -X GET \
   https://<address>/api/controller/v2/jobs/<id_job>/ \
   -H 'Authorization: Bearer <your_token>' \
   -k -s | jq .
```

Подставьте собственные значения параметров вместо употребленных в примере. Здесь <id_job> – идентификатор задания, запущенного ранее, например, 25.

Job


```
{
  "id": 25,
  "type": "job",
  "url": "/api/controller/v2/jobs/25/",
  "related": {
    "created_by": "/api/controller/v2/users/1/",
    "labels": "/api/controller/v2/jobs/25/labels/",
    "inventory": "/api/controller/v2/inventories/3/",
    "project": "/api/controller/v2/projects/12/",
    "organization": "/api/controller/v2/organizations/4/",
    "credentials": "/api/controller/v2/jobs/25/credentials/",
    "unified_job_template": "/api/controller/v2/job_templates/14/",
    "stdout": "/api/controller/v2/jobs/25/stdout/",
    "execution_environment": "/api/controller/v2/execution_environments/2/",
    "job_events": "/api/controller/v2/jobs/25/job_events/",
    "job_host_summaries": "/api/controller/v2/jobs/25/job_host_summaries/",
    "activity_stream": "/api/controller/v2/jobs/25/activity_stream/",
    "notifications": "/api/controller/v2/jobs/25/notifications/",
    "create_schedule": "/api/controller/v2/jobs/25/create_schedule/",
    "job_template": "/api/controller/v2/job_templates/14/",
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
    "cancel": "/api/controller/v2/jobs/25/cancel/",
    "relaunch": "/api/controller/v2/jobs/25/relaunch/"
  },
  "summary_fields": {
    "organization": {
      "id": 4,
      "name": "Jupiter",
      "description": ""
    },
    "inventory": {
      "id": 3,
      "name": "Demo Inventory",
      "description": "",
      "has_active_failures": true,
      "total_hosts": 1,
      "hosts_with_active_failures": 1,
      "total_groups": 0,
      "has_inventory_sources": false,
      "total_inventory_sources": 0,
      "inventory_sources_with_failures": 0,
      "organization_id": 4,
      "kind": ""
    },
    "execution_environment": {
      "id": 2,
      "name": "Default execution environment",
      "description": "",
      "image": "hub.astra-automation.ru/aa-1.2/aa-full-ee:latest"
    },
    "project": {
      "id": 12,
      "name": "Demo Project",
      "description": "",
      "status": "successful",
      "scm_type": "git",
      "allow_override": false
    },
    "job_template": {
      "id": 14,
      "name": "Demo Job templates",
      "description": ""
    },
    "unified_job_template": {
      "id": 14,
      "name": "Demo Job templates",
      "description": "",
      "unified_job_type": "job"
    },
    "instance_group": {
      "id": 2,
      "name": "default",
      "is_container_group": false
    },
    "created_by": {
      "id": 1,
      "username": "admin",
      "first_name": "",
      "last_name": ""
    },
    "user_capabilities": {
      "delete": true,
      "start": true
    },
    "labels": {
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
      "count": 0,
      "results": []
    },
    "credentials": [
      {
        "id": 6,
        "name": "Demo machine credential",
        "description": "",
        "kind": "ssh",
        "cloud": false
      }
    ]
  },
  "created": "2025-11-25T11:42:23.485635Z",
  "modified": "2025-11-25T11:42:24.065872Z",
  "name": "Demo Job templates",
  "description": "",
  "job_type": "run",
  "inventory": 3,
  "project": 12,
  "playbook": "hello_world.yml",
  "scm_branch": "",
  "forks": 0,
  "limit": "",
  "verbosity": 0,
  "extra_vars": "{}",
  "job_tags": "",
  "force_handlers": false,
  "skip_tags": "",
  "start_at_task": "",
  "timeout": 0,
  "use_fact_cache": false,
  "organization": 4,
  "unified_job_template": 14,
  "launch_type": "manual",
  "status": "successful",
  "execution_environment": 2,
  "failed": false,
  "started": "2025-11-25T11:42:24.174853Z",
  "finished": "2025-11-25T11:42:39.374865Z",
  "canceled_on": null,
  "elapsed": 15.2,
  "job_args": "[\"podman\", \"run\", \"--rm\", \"--tty\", \"--interactive\", \"--
↪workdir\", \"/runner/project\", \"-v\", \"/tmp/awx_25_m3pkq3po/:/runner/:Z\", \"-
↪v\", \"/etc/ssl/certs/:/etc/ssl/certs/:O\", \"-v\", \"/usr/local/share/ca-
↪certificates/:/usr/local/share/ca-certificates/:O\", \"--env-file\", \"/tmp/awx_
↪25_m3pkq3po/artifacts/25/env.list\", \"--quiet\", \"--name\", \"ansible_runner_25\
↪", \"--user=root\", \"--network\", \"slirp4netns:enable_ipv6=true\", \"hub.astra-
↪automation.ru/aa-1.2/aa-full-ee:latest\", \"ansible-playbook\", \"-u\", \"root\",␣
↪\"-i\", \"/runner/inventory\", \"-e\", \"@/runner/env/extravars\", \"hello_world.
↪yml\"]",
  "job_cwd": "/runner/project",
  "job_explanation": "",
  "result_traceback": "",
  "event_processing_finished": true,
  "launched_by": {
    "id": 1,
    "name": "admin",
    "type": "user",
    "url": "/api/controller/v2/users/1/"
  },
  "work_unit_id": "1020521290cEbb2WhP",
  "job_template": 14,
  "passwords_needed_to_start": [],
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
  "allow_simultaneous": false,
  "artifacts": {},
  "scm_revision": "347e44fea036c94d5f60e544de006453ee5c71ad",
  "instance_group": 2,
  "diff_mode": false,
  "job_slice_number": 0,
  "job_slice_count": 1,
  "webhook_service": "",
  "webhook_credential": null,
  "webhook_guid": "",
  "host_status_counts": {
    "dark": 1
  },
  "playbook_counts": {
    "play_count": 1,
    "task_count": 1
  },
  "custom_virtualenv": null
}
```


Если в поле `failed` указано значение `false`, значит задание завершилось успешно.

Спецификация API

Примечание

Подробное формальное описание HTTP API представлено в *спецификации*.

11.1.5 Интерфейс командной строки (CLI)

В дополнении к API контроллером можно также удаленно управлять через интерфейс командной строки с помощью пакета CLI (command line interface). Однако, следует учитывать, что с помощью CLI можно управлять контроллером только напрямую, без использования шлюза платформы.

CLI позволяет управлять ресурсами и настройками контроллера:

- создавать, изменять и удалять инвентарные списки, сведения об управляемых узлах, проектах, шаблонах и других ресурсах;
- управлять организациями, командами, пользователями и их ролями;
- создавать, запускать, отменять, планировать, прерывать задания и проверять статус их выполнения;
- и многое другое.

Установка

Для установки CLI контроллера на рабочую машину под управлением ОС Astra Linux Special Edition 1.8 выполните следующие действия:

1. В каталоге `/etc/apt/sources.list.d/` создайте файл `astra-automation.list` со ссылкой на репозиторий Astra Automation:

   ```
   deb https://dl.astralinux.ru/aa/aa-debs-for-alse-1.8 <version> main
   ```

   Вместо <version> необходимо подставить версию устанавливаемой платформы, например, `2.0`.

   Доступные версии продукта опубликованы в таблице *История обновлений*.

2. Обновите список доступных пакетов:

```
sudo apt update
```

3. Установите пакет `automation-controller-cli`:

```
sudo apt install automation-controller-cli --yes
```

Авторизация

Авторизацию пользователя на выполнение действия в контроллере рекомендуется производить с помощью токена OAuth2.0. Для токена могут быть настроены разрешения на чтение и запись, его можно отозвать в любой момент.

Примечание

Без авторизации вы являетесь анонимным пользователем с ограничениями на обращения к точкам доступа API.

Чтобы авторизоваться с помощью OAuth2.0, выполните следующие действия:

1. Сгенерируйте персональный токен доступа на установочном узле контроллера:

```
CONTROLLER_HOST=<address> CONTROLLER_USERNAME=<username> CONTROLLER_PASSWORD=
↪<password> awx login
```

Здесь:

- <address> – URL контроллера, например, `https://192.168.56.11`;
- <username> – название учетной записи пользователя, например, `admin`;
- <password> – пароль, например, `awx`.

Пример вывода команды:

```
{
   "token": "70L1bQv6cj***EHkFaqSGrNZ2S5l4n"
}
```

Важно

Сохраните полученный токен в надежном месте – в случае утраты его нужно будет сгенерировать заново. Восстановить существующий токен невозможно.

Если необходимо создать токен только для чтения, добавьте в команду аргумент `--conf.scope read`:

```
CONTROLLER_HOST=https://<address> CONTROLLER_USERNAME=<username> CONTROLLER_
↪PASSWORD=<password> awx login --conf.scope read
```

Клиентская часть должна иметь возможность установить защищенное соединение с сервером, используя TLS-сертификаты, и провести проверку их подлинности. В случае отсутствия сертификата возможно возникновение ошибки:

```
Could not establish a secure connection.
Please add your server to your certificate authority.
You can also run this command by specifying -k or --conf.insecure
HTTPSConnectionPool(host='192.168.56.11', port=443): Max retries exceeded with url:␣
↪/api/ (Caused by SSLError(SSLCertVerificationError("hostname '192.168.56.11' doesn
↪'t match '=192.168.56.11'")))
```

Если верификация сертификата не требуется или невозможна, добавьте к параметрам вызова команды ключ `--conf.insecure` (`-k`).

2. Для использования токена в последующих командах сохраните его значение в переменную окружения CONTROLLER_OAUTH_TOKEN:

```
export CONTROLLER_OAUTH_TOKEN=<token>
```

Структура команды

Основу CLI контроллера составляет утилита awx, синтаксис вызова которой имеет следующий вид:

```
[<enviroments>] awx [<global-options>] <resource> <action> [<arguments>]
```

- <enviroments> – переменные окружения:
  - CONTROLLER_HOST – URL контроллера.
  - CONTROLLER_VERIFY_SSL – параметры проверки сертификата TLS. Возможные значения:
    - true – строгая проверка сертификата;
    - false – проверка сертификата не выполняется.
  - CONTROLLER_USERNAME – название учетной записи пользователя контроллера.
  - CONTROLLER_PASSWORD – пароль пользователя контроллера.
  - CONTROLLER_OAUTH_TOKEN – токен пользователя контроллера.

Примечание

Переменные окружения и глобальные опции, указывающие одни и те же значения, не могут использоваться одновременно.

- <global-options> – глобальные опции, которые могут применяться ко всем командам:
  - **-h, --help**

    Информация о доступных опциях и командах.
  - **-v, --verbose**

    Дополнительная информация о ходе выполнения команды.
  - **-f, --conf.format**

    Формат вывода:
    - json – JSON;
    - jaml – YAML;
    - jq – формат, используемый утилитой jq[191];
    - human – вывод, удобный для чтения человеком.
  - **--conf.host**

    URL контроллера.
  - **-k, --conf.insecure**

    Разрешение небезопасного соединения с сервером при использовании сертификатов TLS.
  - **--conf.username**

    Название учетной записи пользователя контроллера.
  - **--conf.password**

    Пароль пользователя контроллера.

[191] https://jqlang.github.io/jq/

- **--conf.token**

  Токен пользователя контроллера.

- <resource> – тип объекта или ресурса. Для получения полного списка объектов и ресурсов выполните команду:

```
awx --help
```

- <action> – действие, которое необходимо выполнить с объектом или ресурсом. Ресурсы и объекты, как правило, имеют базовый набор действий:
  - `get` – получение сведений об отдельном экземпляре;
  - `list` – получение списка экземпляров;
  - `create` – создание экземпляра;
  - `modify` – изменение свойств существующего экземпляра;
  - `delete` – удаление экземпляра.

  Чтобы узнать полный список доступных действий для конкретного объекта или ресурса, выполните команду:

```
awx <resource> --help
```

- <arguments> – дополнительные параметры, которые можно передать вместе с командой. Например, идентификатор инвентаря или проекта, название задания и другие.

Примеры использования CLI

Вывод списка организаций:

```
awx organizations list -f human
```

Пример вывода команды:

```
id name
== =====================
1  Default
```

Вывод информации о необходимой организации:

```
awx organizations get <organizations_id>
```

Здесь <organizations_id> – идентификатор организации, например, 1.

Пример вывода команды:

```
{
  "id": 1,
  "type": "organization",
  "url": "/api/v2/organizations/1/",
  "summary_fields": {
    "created_by": {
      "id": 1,
      "username": "admin",
      "first_name": "",
      "last_name": ""
    },
    "modified_by": {
      "id": 1,
      "username": "admin",
      "first_name": "",
      "last_name": ""
    },
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
    "object_roles": {
      "admin_role": {
        "description": "Can manage all aspects of the organization",
        "name": "Admin",
        "id": 9,
        "user_only": true
      },
      "execute_role": {
        "description": "May run any executable resources in the organization",
        "name": "Execute",
        "id": 10
      },
      "project_admin_role": {
        "description": "Can manage all projects of the organization",
        "name": "Project Admin",
        "id": 11
      },
      "inventory_admin_role": {
        "description": "Can manage all inventories of the organization",
        "name": "Inventory Admin",
        "id": 12
      },
      "credential_admin_role": {
        "description": "Can manage all credentials of the organization",
        "name": "Credential Admin",
        "id": 13
      },
      "workflow_admin_role": {
        "description": "Can manage all workflows of the organization",
        "name": "Workflow Admin",
        "id": 14
      },
      "notification_admin_role": {
        "description": "Can manage all notifications of the organization",
        "name": "Notification Admin",
        "id": 15
      },
      "job_template_admin_role": {
        "description": "Can manage all job templates of the organization",
        "name": "Job Template Admin",
        "id": 16
      },
      "execution_environment_admin_role": {
        "description": "Can manage all execution environments of the organization",
        "name": "Execution Environment Admin",
        "id": 17
      },
      "auditor_role": {
        "description": "Can view all aspects of the organization",
        "name": "Auditor",
        "id": 18
      },
      "member_role": {
        "description": "User is a member of the organization",
        "name": "Member",
        "id": 19,
        "user_only": true
      },
      "read_role": {
        "description": "May view settings for the organization",
        "name": "Read",
        "id": 20
      },
      "approval_role": {
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
        "description": "Can approve or deny a workflow approval node",
        "name": "Approve",
        "id": 21
      }
    },
    "user_capabilities": {
      "edit": true,
      "delete": true
    },
    "related_field_counts": {
      "users": 2,
      "admins": 0,
      "inventories": 2,
      "teams": 3,
      "projects": 4,
      "job_templates": 2,
      "hosts": 2
    }
  },
  "created": "2024-02-16T00:06:02.349704Z",
  "modified": "2024-02-16T00:30:54.043146Z",
  "name": "Default",
  "description": "",
  "max_hosts": 0,
  "custom_virtualenv": null,
  "default_environment": null
}
```


Создание организации:

```
awx organizations create --name "<your_organization_name>"
```

Здесь <your_organization_name> – название организации, например, Saturn.

Пример вывода команды:


```
{
  "id": 4,
  "type": "organization",
  "url": "/api/v2/organizations/4/",
  "summary_fields": {
    "created_by": {
      "id": 1,
      "username": "admin",
      "first_name": "",
      "last_name": ""
    },
    "modified_by": {
      "id": 1,
      "username": "admin",
      "first_name": "",
      "last_name": ""
    },
    "object_roles": {
      "admin_role": {
        "description": "Can manage all aspects of the organization",
        "name": "Admin",
        "id": 171,
        "user_only": true
      },
      "execute_role": {
        "description": "May run any executable resources in the organization",
        "name": "Execute",
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
      "id": 172
    },
    "project_admin_role": {
      "description": "Can manage all projects of the organization",
      "name": "Project Admin",
      "id": 173
    },
    "inventory_admin_role": {
      "description": "Can manage all inventories of the organization",
      "name": "Inventory Admin",
      "id": 174
    },
    "credential_admin_role": {
      "description": "Can manage all credentials of the organization",
      "name": "Credential Admin",
      "id": 175
    },
    "workflow_admin_role": {
      "description": "Can manage all workflows of the organization",
      "name": "Workflow Admin",
      "id": 176
    },
    "notification_admin_role": {
      "description": "Can manage all notifications of the organization",
      "name": "Notification Admin",
      "id": 177
    },
    "job_template_admin_role": {
      "description": "Can manage all job templates of the organization",
      "name": "Job Template Admin",
      "id": 178
    },
    "execution_environment_admin_role": {
      "description": "Can manage all execution environments of the organization",
      "name": "Execution Environment Admin",
      "id": 179
    },
    "auditor_role": {
      "description": "Can view all aspects of the organization",
      "name": "Auditor",
      "id": 180
    },
    "member_role": {
      "description": "User is a member of the organization",
      "name": "Member",
      "id": 181,
      "user_only": true
    },
    "read_role": {
      "description": "May view settings for the organization",
      "name": "Read",
      "id": 182
    },
    "approval_role": {
      "description": "Can approve or deny a workflow approval node",
      "name": "Approve",
      "id": 183
    }
  },
  "user_capabilities": {
    "edit": true,
    "delete": true
  },
  "related_field_counts": {
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
      "inventories": 0,
      "teams": 0,
      "users": 0,
      "job_templates": 0,
      "admins": 0,
      "projects": 0
    }
  },
  "created": "2024-06-10T11:12:29.167911Z",
  "modified": "2024-06-10T11:12:29.167924Z",
  "name": "Saturn",
  "description": "",
  "max_hosts": 0,
  "custom_virtualenv": null,
  "default_environment": null
}
```


Изменение названия организации:

```
awx organizations modify <organization_id> --name "<new_name_organization>"
```

Здесь <new_name_organization> – новое название организации, например, Neptune.

Пример вывода команды:


```
{
  "id": 4,
  "type": "organization",
  "url": "/api/v2/organizations/4/",
  "summary_fields": {
    "created_by": {
      "id": 1,
      "username": "admin",
      "first_name": "",
      "last_name": ""
    },
    "modified_by": {
      "id": 1,
      "username": "admin",
      "first_name": "",
      "last_name": ""
    },
    "object_roles": {
      "admin_role": {
        "description": "Can manage all aspects of the organization",
        "name": "Admin",
        "id": 171,
        "user_only": true
      },
      "execute_role": {
        "description": "May run any executable resources in the organization",
        "name": "Execute",
        "id": 172
      },
      "project_admin_role": {
        "description": "Can manage all projects of the organization",
        "name": "Project Admin",
        "id": 173
      },
      "inventory_admin_role": {
        "description": "Can manage all inventories of the organization",
        "name": "Inventory Admin",
        "id": 174
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)


```
    },
    "credential_admin_role": {
      "description": "Can manage all credentials of the organization",
      "name": "Credential Admin",
      "id": 175
    },
    "workflow_admin_role": {
      "description": "Can manage all workflows of the organization",
      "name": "Workflow Admin",
      "id": 176
    },
    "notification_admin_role": {
      "description": "Can manage all notifications of the organization",
      "name": "Notification Admin",
      "id": 177
    },
    "job_template_admin_role": {
      "description": "Can manage all job templates of the organization",
      "name": "Job Template Admin",
      "id": 178
    },
    "execution_environment_admin_role": {
      "description": "Can manage all execution environments of the organization",
      "name": "Execution Environment Admin",
      "id": 179
    },
    "auditor_role": {
      "description": "Can view all aspects of the organization",
      "name": "Auditor",
      "id": 180
    },
    "member_role": {
      "description": "User is a member of the organization",
      "name": "Member",
      "id": 181,
      "user_only": true
    },
    "read_role": {
      "description": "May view settings for the organization",
      "name": "Read",
      "id": 182
    },
    "approval_role": {
      "description": "Can approve or deny a workflow approval node",
      "name": "Approve",
      "id": 183
    }
  },
  "user_capabilities": {
    "edit": true,
    "delete": true
  },
  "related_field_counts": {
    "users": 0,
    "admins": 0,
    "inventories": 0,
    "teams": 0,
    "projects": 0,
    "job_templates": 0,
    "hosts": 0
  }
},
"created": "2024-06-10T11:12:29.167911Z",
"modified": "2024-06-10T11:19:11.396969Z",
```


(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  "name": "Neptune",
  "description": "",
  "max_hosts": 0,
  "custom_virtualenv": null,
  "default_environment": null
}
```

Удаление организации:

```
awx organizations delete <organization_id>
```

Базовый сценарий

Базовый сценарий позволяет ознакомиться с основами использования утилиты CLI awx для управления Automation Controller.

Описание сценария

Этот сценарий содержит следующие действия:

1. Настройка окружения для получения доступа к Automation Controller с помощью утилиты awx.
2. Создание организации, с которой связываются все создаваемые далее ресурсы.
3. Создание инвентаря и добавление в него двух управляемых узлов.
4. Создание полномочий следующих типов:
   - *Машина* (Machine) для доступа к управляемым узлам;
   - *Система управления версиями* (Source Control) для доступа к Git-репозиторию с кодом проекта.
5. Создание проекта.

   Код проекта загружается из репозитория ПАО Группа Астра, доступного по адресу https://source.astragroup.ru/aa-gca/AA/aac-samples.
6. Создание шаблона задания, использующего наборов сценариев `playbooks/demo_helloworld.yml`.
7. Запуск задания на основе шаблона и вывод результатов его выполнения.

Подготовка к работе

Для подготовки окружения к работе выполните следующие действия:

1. Настройте управляемые узлы согласно *инструкции*.
2. На сайте https://source.astragroup.ru/ настройте доступ к сервису Git по SSH:
   1. Авторизуйтесь, используя свою учетную запись в Личном кабинете[192].
   2. В правом верхнем углу нажмите на кнопку с аватаром и в открывшемся меню выберите **Edit profile**.
   3. На панели навигации выберите **SSH Keys**.
   4. Заполните форму добавления ключа SSH:
      - **Key** – содержимое публичного ключа SSH, используемого для доступа к репозиторию.
      - **Title** – необязательное название ключа.
      - **Expiration date** – срок действия ключа. Если это поле не заполнено, ключ действует неограниченно долго.

[192] https://lk.astra.ru

5. Нажмите кнопку *Add key*.

3. На вашей рабочей станции установите пакет `automation-controller-cli` согласно *инструкции*.

Авторизация

Чтобы управлять Automation Controller с помощью утилиты awx, выполните следующие действия:

1. Импортируйте в переменные окружения необходимые сведения о контроллере:

```
export CONTROLLER_HOST=https://<address>
export CONTROLLER_USERNAME=<username>
export CONTROLLER_PASSWORD=<password>
```

   Здесь:

   - <address> – IP-адрес или FQDN контроллера;
   - <username> – название учетной записи администратора;
   - <password> – пароль администратора контроллера.

2. Сгенерируйте персональный токен доступа:

```
awx login
```

   Эта команда возвращает токен в формате JSON, например:

```
{
  "token": "70L1bQv6cj***EHkFaqSGrNZ2S5l4n"
}
```

   Сохраните значение поля `token` в надежном месте – в случае утраты его нужно будет сгенерировать заново. Восстановить существующий токен невозможно.

3. Экспортируйте токен в переменную окружения `CONTROLLER_OAUTH_TOKEN`:

```
export CONTROLLER_OAUTH_TOKEN=<token>
```

Создание организации

1. Создайте организацию Jupiter:

```
awx organizations create --name 'Jupiter'
```

2. Для получения идентификатора организации выполните команду:

```
awx organizations list --all -f human
```

   Она возвращает таблицу следующего вида:

```
id name
== =======
1  Default
2  Jupiter
```

Описание инвентаря

Подготовьте описание инвентаря с помощью следующих действий:

1. Создайте инвентарь Jupiter Inventory, принадлежащий организации Jupiter (ID=2):

```
awx inventory create \
   --name 'Jupiter Inventory' \
   --organization 2
```

2. Для получения идентификатора созданного инвентаря выполните команду:

```
awx inventory list --all -f human
```

Она возвращает таблицу следующего вида:

```
id name
== =================
1  Demo inventory
2  Jupiter Inventory
```

3. Добавьте в Jupiter Inventory (ID=2) два управляемых узла:

```
awx host create \
   --name '<node_name>' \
   --inventory 2 \
   --variables '<node_variables>'
```

Здесь:

- <node_name> – название управляемого узла, например, node01;
- <node_variables> –переменные управляемого узла в формате JSON или YAML, например:

```
{
  "ansible_host": "192.168.56.101"
}
```

Создание полномочий

Для доступа к управляемым узлам и Git-репозиторию с кодом проекта необходимы полномочия. Создайте их с помощью команды *awx credentials create*. Данные укажите в формате JSON, используя следующие поля:

- username – название учетной записи.
- password – пароль.
- ssh_key_data – содержимое файла с приватным ключом SSH или путь к нему.

  При указании содержимого файла замените переходы на новую строку последовательностью \n, например:

```
{
  "username": "astra",
  "ssh_key_data": "-----BEGIN OPENSSH PRIVATE KEY-----\nb3******FBg==\n-----END␣
↪OPENSSH PRIVATE KEY-----"
}
```

  При указании пути к файлу добавьте в начало строки символ @:

```
{
  "username": "astra",
  "ssh_key_data": "@~/.ssh/private-key"
}
```

Примечание

По умолчанию поиск выполняется относительно текущего каталога.

- ssh_key_unlock – если приватный ключ SSH защищен паролем, укажите его в этом поле.

Список 3: Пример заполнения JSON с данными полномочия

```
{
  "username": "administrator",
  "ssh_key_data": "@~/.ssh/private-key",
  "ssh_key_unlock": "p@ssW0rD"
}
```

1. Создайте полномочие Demo machine credential типа *Машина* (Machine), принадлежащее организации Jupiter (ID=2):

   ```
   awx credentials create \
      --name 'Demo machine credential' \
      --credential_type 'Machine' \
      --organization 2 \
      --inputs '<node_data>'
   ```

   Здесь <node_inputs> – учетные данные в формате JSON или YAML для доступа к управляемому узлу.

2. Создайте полномочие Demo sources credential типа *Система управления версиями* (Source Control), принадлежащее организации Jupiter (ID=2):

   ```
   awx credentials create \
      --name 'Demo sources credential' \
      --credential_type 'Source Control' \
      --organization 2 \
      --inputs '<git_data>'
   ```

   Здесь <git_data> – учетные данные в формате JSON или YAML для доступа к Git-репозиторию с кодом демонстрационного проекта.

3. Для получения идентификаторов созданных полномочий выполните команду:

   ```
   awx credentials list --all -f human
   ```

   Она возвращает таблицу следующего вида:

   ```
   id name
   == ======================
   2  Ansible Galaxy
   1  Demo Credential
   3  Demo machine credential
   4  Demo sources credential
   ```

Подготовка проекта

1. Создайте проект AWX CLI demo project:

   ```
   awx projects create \
      --wait \
      --organization 2 \
      --name='AWX CLI demo project' \
      --scm_type git \
      --scm_url ssh://git@source.astragroup.ru:2222/aa-gca/AA/aac-samples.git \
      --credential 4
   ```

   Эта команда создает проект, используя следующие ресурсы:

Тип ресурса	Идентификатор	Название
Организация	2	Jupiter
Полномочие	4	Demo sources credential

2. Для получения идентификатора созданного проекта выполните команду:

```
awx projects list --all -f human
```

Она возвращает таблицу следующего вида:

```
id name
== ====================
6  Demo Project
8  AWX CLI demo project
```

Подготовка задания

Для подготовки задания выполните следующие действия:

1. Создайте шаблон задания AWX CLI demo job template:

```
awx job_templates create \
   --name 'AWX CLI demo job template' \
   --project 8 \
   --playbook playbooks/demo_helloworld.yml \
   --inventory 2
```

Эта команда создает шаблон задания, используя следующие ресурсы:

Тип ресурса	Идентификатор	Название
Проект	8	AWX CLI demo project
Инвентарь	2	Jupiter Inventory

2. Для получения идентификатора шаблона задания выполните команду:

```
awx job_templates list --all -f human
```

Она возвращает таблицу следующего вида:

```
id name
== ====
10 AWX CLI demo job template
7  Demo Job Template
```

Запуск задания и проверка его статуса

Для запуска задания на основе шаблона AWX CLI demo job template выполните команду:

```
awx job_templates launch \
  --credentials 3 \
  --monitor \
  10
```

Эта команда запускает задание, используя следующие ресурсы:

Тип ресурса	Идентификатор	Название
Шаблон задания	10	AWX CLI demo job template
Полномочия	3	Demo machine credential

Результаты выполнения задания выводятся в терминал (часть вывода опущена с целью сокращения):

```
------Starting Standard Out Stream------
Identity added: /runner/artifacts/4/ssh_key_data (Vagrant SSH key)

PLAY [Hello World Playbook] ****************************************************

TASK [Print Hello World] *******************************************************
ok: [managed1] => {
    "msg": "Hello World from managed1"
}
ok: [managed2] => {
    "msg": "Hello World from managed2"
}

PLAY RECAP *********************************************************************
managed1                   : ok=1    changed=0    unreachable=0    failed=0    skipped=0␣
↪    rescued=0    ignored=0
managed2                   : ok=1    changed=0    unreachable=0    failed=0    skipped=0␣
↪    rescued=0    ignored=0
------End of Standard Out Stream--------
```

Если в поле `failed` указано значение `0`, значит задание завершилось успешно.

CLI specification

Здесь представлена полная формальная спецификация интерфейса командной строки (*CLI*), основанного на использовании утилиты awx. Подобное описание для любого сочетания ресурс-действие (resource-action) можно получить с помощью аргумента `--help`, выполнив соответствующую команду по отношению к работающему контроллеру.

Для получения синтаксиса команды по управлению конкретным ресурсом выберите в панели навигации слева название этого ресурса. Локальная панель навигации справа позволит переключиться на описание синтаксиса необходимого действия по отношению к этому ресурсу.

Ниже представлен обобщенный синтаксис утилиты awx.

```
usage: awx [--help] [--version] [--conf.host https://example.awx.org] [--conf.token␣
↪TEXT] [--conf.username TEXT] [--conf.password TEXT] [-k] [-f {json,yaml,jq,human}]
           [--filter TEXT] [--conf.color BOOLEAN] [-v]
           resource ...

positional arguments:
  resource
    login               authenticate and retrieve an OAuth2 token
    config              print current configuration values
    import              import resources into Tower
    export              export resources from Tower
    ping
    instances (instance)
    instance_groups (instance_group)
    receptor_addresses
    settings (setting)
    me
    organizations (organization)
    users (user)
    execution_environments (execution_environment)
    projects (project)
    project_updates (project_update)
    teams (team)
    credentials (credential)
    credential_types (credential_type)
    credential_input_sources
    applications (application)
    tokens
    metrics
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
    inventory (inventories)
    constructed_inventory
    inventory_sources (inventory_source)
    inventory_updates (inventory_update)
    groups (group)
    hosts (host)
    host_metrics
    host_metric_summary_monthly
    job_templates (job_template)
    jobs (job)
    ad_hoc_commands (ad_hoc)
    system_job_templates
    system_jobs
    schedules (schedule)
    roles (role)
    notification_templates (notification_template)
    notifications
    labels (label)
    unified_job_templates
    unified_jobs
    activity_stream
    workflow_job_templates (workflow)
    workflow_jobs (workflow_job)
    workflow_approvals
    workflow_job_template_nodes (node)
    workflow_job_nodes
    mesh_visualizer
    bulk
    analytics
    service_index
    role_definitions
    role_user_assignments
    role_team_assignments

options:
  --help                prints usage information for the awx tool
  --version             display awx CLI version

authentication:
  --conf.host https://example.awx.org
  --conf.token TEXT     an OAuth2.0 token (get one by using `awx login`)
  --conf.username TEXT
  --conf.password TEXT
  -k, --conf.insecure   Allow insecure server connections when using SSL

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

activity_stream

```
usage: awx activity_stream [-h] action ...

positional arguments:
  action
    list
    get

options:
  -h, --help  show this help message and exit
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
awx activity_stream: the following arguments are required: action
```

activity_stream list

```
usage: awx activity_stream list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,jq,
↪human}] [--filter TEXT] [--conf.color BOOLEAN] [-v] [--type {activity_stream}]
                                [--timestamp TIMESTAMP] [--operation {create,update,
↪delete,associate,disassociate}] [--changes JSON/YAML] [--object1 TEXT]
                                [--object2 TEXT] [--action_node TEXT]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --type {activity_stream}
                        only list activity_stream with the specified type
  --timestamp TIMESTAMP
                        only list activity_stream with the specified timestamp
  --operation {create,update,delete,associate,disassociate}
                        only list activity_stream with the specified operation
  --changes JSON/YAML   only list activity_stream with the specified changes
  --object1 TEXT        only list activity_stream with the specified object1
  --object2 TEXT        only list activity_stream with the specified object2
  --action_node TEXT    only list activity_stream with the specified action_node

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

activity_stream get

```
usage: awx activity_stream get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--conf.
↪color BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx activity_stream get: the following arguments are required: id
```

ad_hoc_commands

```
usage: awx ad_hoc_commands [-h] action ...

positional arguments:
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  action
    list
    create
    get
    delete
    stdout

options:
  -h, --help  show this help message and exit

awx ad_hoc_commands: the following arguments are required: action
```

ad_hoc_commands list

```
usage: awx ad_hoc_commands list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,jq,
↪human}] [--filter TEXT] [--conf.color BOOLEAN] [-v] [--type {ad_hoc_command}]
                                [--created CREATED] [--modified MODIFIED] [--name TEXT]
                                [--launch_type {manual,relaunch,callback,scheduled,
↪dependency,workflow,webhook,sync,scm}]
                                [--status {new,pending,waiting,running,successful,failed,
↪error,canceled}] [--execution_environment ID] [--failed BOOLEAN]
                                [--started STARTED] [--finished FINISHED] [--canceled_on␣
↪CANCELED_ON] [--elapsed ELAPSED] [--job_explanation TEXT]
                                [--execution_node TEXT] [--controller_node TEXT] [--work_
↪unit_id TEXT] [--job_type {run,check}] [--inventory ID] [--limit TEXT]
                                [--block_inventory BOOLEAN] [--credential ID]
                                [--module_name {command,shell,yum,apt,apt_key,apt_
↪repository,apt_rpm,service,group,user,mount,ping,selinux,setup,win_ping,win_service,
↪win_updates,win_group,win_user}]
                                [--module_args TEXT] [--forks INTEGER] [--verbosity {0,1,
↪2,3,4,5}] [--extra_vars TEXT] [--become_enabled BOOLEAN]
                                [--diff_mode BOOLEAN]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --type {ad_hoc_command}
                        only list ad_hoc_commands with the specified type
  --created CREATED     only list ad_hoc_commands with the specified created
  --modified MODIFIED   only list ad_hoc_commands with the specified modified
  --name TEXT           only list ad_hoc_commands with the specified name
  --launch_type {manual,relaunch,callback,scheduled,dependency,workflow,webhook,sync,scm}
                        only list ad_hoc_commands with the specified launch_type
  --status {new,pending,waiting,running,successful,failed,error,canceled}
                        only list ad_hoc_commands with the specified status
  --execution_environment ID
                        only list ad_hoc_commands with the specified execution_
↪environment
  --failed BOOLEAN      only list ad_hoc_commands with the specified failed
  --started STARTED     only list ad_hoc_commands with the specified started
  --finished FINISHED   only list ad_hoc_commands with the specified finished
  --canceled_on CANCELED_ON
                        only list ad_hoc_commands with the specified canceled_on
  --elapsed ELAPSED     only list ad_hoc_commands with the specified elapsed
  --job_explanation TEXT
                        only list ad_hoc_commands with the specified job_explanation
  --execution_node TEXT
                        only list ad_hoc_commands with the specified execution_node
  --controller_node TEXT
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
                        only list ad_hoc_commands with the specified controller_node
  --work_unit_id TEXT   only list ad_hoc_commands with the specified work_unit_id
  --job_type {run,check}
                        only list ad_hoc_commands with the specified job_type
  --inventory ID        only list ad_hoc_commands with the specified inventory
  --limit TEXT          only list ad_hoc_commands with the specified limit
  --block_inventory BOOLEAN
                        only list ad_hoc_commands with the specified block_inventory
  --credential ID       only list ad_hoc_commands with the specified credential
  --module_name {command,shell,yum,apt,apt_key,apt_repository,apt_rpm,service,group,user,
↪mount,ping,selinux,setup,win_ping,win_service,win_updates,win_group,win_user}
                        only list ad_hoc_commands with the specified module_name
  --module_args TEXT    only list ad_hoc_commands with the specified module_args
  --forks INTEGER       only list ad_hoc_commands with the specified forks
  --verbosity {0,1,2,3,4,5}
                        only list ad_hoc_commands with the specified verbosity
  --extra_vars TEXT     only list ad_hoc_commands with the specified extra_vars
  --become_enabled BOOLEAN
                        only list ad_hoc_commands with the specified become_enabled
  --diff_mode BOOLEAN   only list ad_hoc_commands with the specified diff_mode

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

ad_hoc_commands create

```
usage: awx ad_hoc_commands create [-h] [--monitor] [--action-timeout ACTION_TIMEOUT] [--
↪wait] [--interval INTERVAL] [--execution_environment ID]
                                  [--job_type {run,check}] --inventory ID [--limit TEXT]␣
↪[--block_inventory BOOLEAN] --credential ID
                                  [--module_name {command,shell,yum,apt,apt_key,apt_
↪repository,apt_rpm,service,group,user,mount,ping,selinux,setup,win_ping,win_service,
↪win_updates,win_group,win_user}]
                                  [--module_args TEXT] [--forks INTEGER] [--verbosity {0,
↪1,2,3,4,5}] [--extra_vars TEXT] [--become_enabled BOOLEAN]
                                  [--diff_mode BOOLEAN]

required arguments:
  --inventory ID        the ID of the associated inventory
  --credential ID       the ID of the associated credential

options:
  -h, --help            show this help message and exit
  --monitor             If set, prints stdout of the launched job until it finishes.
  --action-timeout ACTION_TIMEOUT
                        If set with --monitor or --wait, time out waiting on job␣
↪completion.
  --wait                If set, waits until the launched job finishes.
  --interval INTERVAL   If set with --monitor or --wait, amount of time to wait in␣
↪seconds between api calls. Minimum value is 2.5 seconds to avoid overwhelming the
                        api
  --execution_environment ID
                        The container image to be used for execution.
  --job_type {run,check}
  --limit TEXT
  --block_inventory BOOLEAN
  --module_name {command,shell,yum,apt,apt_key,apt_repository,apt_rpm,service,group,user,
↪mount,ping,selinux,setup,win_ping,win_service,win_updates,win_group,win_user}
  --module_args TEXT
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  --forks INTEGER
  --verbosity {0,1,2,3,4,5}
  --extra_vars TEXT
  --become_enabled BOOLEAN
  --diff_mode BOOLEAN

awx ad_hoc_commands create: the following arguments are required: --inventory, --
↪credential
```

ad_hoc_commands get

```
usage: awx ad_hoc_commands get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--conf.
↪color BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx ad_hoc_commands get: the following arguments are required: id
```

ad_hoc_commands delete

```
usage: awx ad_hoc_commands delete [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx ad_hoc_commands delete: the following arguments are required: id
```

ad_hoc_commands stdout

```
usage: awx ad_hoc_commands stdout [-h] id

positional arguments:
  id

options:
  -h, --help  show this help message and exit

awx ad_hoc_commands stdout: the following arguments are required: id
```

analytics

```
usage: awx analytics [-h] action ...

positional arguments:
  action
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
    list
    get

options:
  -h, --help  show this help message and exit

awx analytics: the following arguments are required: action
```

analytics list

```
usage: awx analytics list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,jq,human}] [-
↪-filter TEXT] [--conf.color BOOLEAN] [-v]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

analytics get

```
usage: awx analytics get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--conf.color␣
↪BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx analytics get: the following arguments are required: id
```

applications

```
usage: awx applications [-h] action ...

positional arguments:
  action
    list
    create
    get
    modify
    delete

options:
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  -h, --help  show this help message and exit

awx applications: the following arguments are required: action
```

applications list

```
usage: awx applications list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,jq,human}
↪] [--filter TEXT] [--conf.color BOOLEAN] [-v] [--type {o_auth2_application}]
                             [--created CREATED] [--name TEXT] [--description TEXT] [--
↪client_id TEXT] [--client_secret TEXT] [--client_type {confidential,public}]
                             [--redirect_uris TEXT] [--authorization_grant_type
↪{authorization-code,password}] [--skip_authorization BOOLEAN] [--organization ID]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --type {o_auth2_application}
                        only list applications with the specified type
  --created CREATED     only list applications with the specified created
  --name TEXT           only list applications with the specified name
  --description TEXT    only list applications with the specified description
  --client_id TEXT      only list applications with the specified client_id
  --client_secret TEXT  only list applications with the specified client_secret
  --client_type {confidential,public}
                        only list applications with the specified client_type
  --redirect_uris TEXT  only list applications with the specified redirect_uris
  --authorization_grant_type {authorization-code,password}
                        only list applications with the specified authorization_grant_
↪type
  --skip_authorization BOOLEAN
                        only list applications with the specified skip_authorization
  --organization ID     only list applications with the specified organization

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

applications create

```
usage: awx applications create [-h] --name TEXT [--description TEXT] --client_type
↪{confidential,public} [--redirect_uris TEXT] --authorization_grant_type
                               {authorization-code,password} [--skip_authorization␣
↪BOOLEAN] --organization ID

required arguments:
  --name TEXT           Name of this application.
  --client_type {confidential,public}
                        Set to Public or Confidential depending on how secure the client␣
↪device is.
  --authorization_grant_type {authorization-code,password}
                        The Grant type the user must use for acquire tokens for this␣
↪application.
  --organization ID     Organization containing this application.

options:
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  -h, --help            show this help message and exit
  --description TEXT    Optional description of this application.
  --redirect_uris TEXT  Allowed URIs list, space separated
  --skip_authorization BOOLEAN
                        Set True to skip authorization step for completely trusted␣
↪applications.

awx applications create: the following arguments are required: --name, --client_type, --
↪authorization_grant_type, --organization
```

applications get

```
usage: awx applications get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--conf.color␣
↪BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx applications get: the following arguments are required: id
```

applications modify

```
usage: awx applications modify [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx applications modify: the following arguments are required: id
```

applications delete

```
usage: awx applications delete [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx applications delete: the following arguments are required: id
```

bulk

```
usage: awx bulk [-h] action ...

positional arguments:
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  action
    list
    get
    job_launch
    host_create
    host_delete

options:
  -h, --help   show this help message and exit

awx bulk: the following arguments are required: action
```

bulk list

```
usage: awx bulk list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,jq,human}] [--
↪filter TEXT] [--conf.color BOOLEAN] [-v]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

bulk get

```
usage: awx bulk get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--conf.color␣
↪BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx bulk get: the following arguments are required: id
```

bulk job_launch

```
usage: awx bulk job_launch [-h] [--monitor] [--action-timeout ACTION_TIMEOUT] [--wait] [-
↪-interval INTERVAL] [--name TEXT] --jobs JOBS [--description TEXT]
                           [--extra_vars JSON/YAML] [--organization ID] [--inventory ID]␣
↪[--limit TEXT] [--scm_branch TEXT] [--skip_tags TEXT] [--job_tags TEXT]

required arguments:
  --jobs JOBS           List of jobs to be launched, JSON. e.g. [{"unified_job_template
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
↪": 7}, {"unified_job_template": 10}]

options:
  -h, --help            show this help message and exit
  --monitor             If set, prints stdout of the launched job until it finishes.
  --action-timeout ACTION_TIMEOUT
                        If set with --monitor or --wait, time out waiting on job␣
↪completion.
  --wait                If set, waits until the launched job finishes.
  --interval INTERVAL   If set with --monitor or --wait, amount of time to wait in␣
↪seconds between api calls. Minimum value is 2.5 seconds to avoid overwhelming the
                        api
  --name TEXT           Name of this workflow job.
  --description TEXT    Optional description of this workflow job.
  --extra_vars JSON/YAML
                        a JSON or YAML string. You can optionally specify a file path e.
↪g., @path/to/file.yml
  --organization ID     Inherit permissions from this organization. If not provided, a␣
↪organization the user is a member of will be selected automatically.
  --inventory ID        the ID of the associated inventory
  --limit TEXT
  --scm_branch TEXT
  --skip_tags TEXT
  --job_tags TEXT

awx bulk job_launch: the following arguments are required: --jobs
```

bulk host_create

```
usage: awx bulk host_create [-h] --inventory ID --hosts HOSTS

required arguments:
  --inventory ID  Primary Key ID of inventory to add hosts to.
  --hosts HOSTS   List of hosts to be created, JSON. e.g. [{"name": "example.com"}, {
↪"name": "127.0.0.1"}]

options:
  -h, --help      show this help message and exit

awx bulk host_create: the following arguments are required: --inventory, --hosts
```

bulk host_delete

```
usage: awx bulk host_delete [-h] --hosts HOSTS

required arguments:
  --hosts HOSTS  List of hosts ids to be deleted, e.g. [105, 130, 131, 200]

options:
  -h, --help     show this help message and exit

awx bulk host_delete: the following arguments are required: --hosts
```

config

```
usage: awx config [-h]

options:
  -h, --help  show this help message and exit
```

constructed_inventory

```
usage: awx constructed_inventory [-h] action ...

positional arguments:
  action
    list
    create
    get
    modify
    delete

options:
  -h, --help  show this help message and exit

awx constructed_inventory: the following arguments are required: action
```

constructed_inventory list

```
usage: awx constructed_inventory list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,
↪jq,human}] [--filter TEXT] [--conf.color BOOLEAN] [-v] [--type {inventory}]
                                      [--created CREATED] [--modified MODIFIED] [--name␣
↪TEXT] [--description TEXT] [--organization ID] [--kind {,smart,constructed}]
                                      [--variables JSON/YAML] [--has_active_failures␣
↪BOOLEAN] [--total_hosts INTEGER] [--hosts_with_active_failures INTEGER]
                                      [--total_groups INTEGER] [--has_inventory_sources␣
↪BOOLEAN] [--total_inventory_sources INTEGER]
                                      [--inventory_sources_with_failures INTEGER] [--
↪pending_deletion BOOLEAN] [--prevent_instance_group_fallback BOOLEAN]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --type {inventory}    only list constructed_inventory with the specified type
  --created CREATED     only list constructed_inventory with the specified created
  --modified MODIFIED   only list constructed_inventory with the specified modified
  --name TEXT           only list constructed_inventory with the specified name
  --description TEXT    only list constructed_inventory with the specified description
  --organization ID     only list constructed_inventory with the specified organization
  --kind {,smart,constructed}
                        only list constructed_inventory with the specified kind
  --variables JSON/YAML
                        only list constructed_inventory with the specified variables
  --has_active_failures BOOLEAN
                        only list constructed_inventory with the specified has_active_
↪failures
  --total_hosts INTEGER
                        only list constructed_inventory with the specified total_hosts
  --hosts_with_active_failures INTEGER
                        only list constructed_inventory with the specified hosts_with_
↪active_failures
  --total_groups INTEGER
                        only list constructed_inventory with the specified total_groups
  --has_inventory_sources BOOLEAN
                        only list constructed_inventory with the specified has_inventory_
↪sources
  --total_inventory_sources INTEGER
                        only list constructed_inventory with the specified total_
↪inventory_sources
  --inventory_sources_with_failures INTEGER
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
                        only list constructed_inventory with the specified inventory_
↪sources_with_failures
  --pending_deletion BOOLEAN
                        only list constructed_inventory with the specified pending_
↪deletion
  --prevent_instance_group_fallback BOOLEAN
                        only list constructed_inventory with the specified prevent_
↪instance_group_fallback

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

constructed_inventory create

```
usage: awx constructed_inventory create [-h] --name TEXT [--description TEXT] --
↪organization ID [--variables JSON/YAML] [--prevent_instance_group_fallback BOOLEAN]
                                        [--source_vars TEXT] [--update_cache_timeout␣
↪INTEGER] [--limit TEXT] [--verbosity INTEGER]

required arguments:
  --name TEXT           Name of this inventory.
  --organization ID     Organization containing this inventory.

options:
  -h, --help            show this help message and exit
  --description TEXT    Optional description of this inventory.
  --variables JSON/YAML
                        Inventory variables in JSON or YAML format. You can optionally␣
↪specify a file path e.g., @path/to/file.yml
  --prevent_instance_group_fallback BOOLEAN
                        If enabled, the inventory will prevent adding any organization␣
↪instance groups to the list of preferred instances groups to run associated job
                        templates on.If this setting is enabled and you provided an␣
↪empty list, the global instance groups will be applied.
  --source_vars TEXT    The source_vars for the related auto-created inventory source,␣
↪special to constructed inventory.
  --update_cache_timeout INTEGER
                        The cache timeout for the related auto-created inventory source,␣
↪special to constructed inventory
  --limit TEXT          The limit to restrict the returned hosts for the related auto-
↪created inventory source, special to constructed inventory.
  --verbosity INTEGER   The verbosity level for the related auto-created inventory␣
↪source, special to constructed inventory

awx constructed_inventory create: the following arguments are required: --name, --
↪organization
```

constructed_inventory get

```
usage: awx constructed_inventory get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--
↪conf.color BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx constructed_inventory get: the following arguments are required: id
```

constructed_inventory modify

```
usage: awx constructed_inventory modify [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx constructed_inventory modify: the following arguments are required: id
```

constructed_inventory delete

```
usage: awx constructed_inventory delete [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx constructed_inventory delete: the following arguments are required: id
```

credential_input_sources

```
usage: awx credential_input_sources [-h] action ...

positional arguments:
  action
    list
    create
    get
    modify
    delete

options:
  -h, --help  show this help message and exit

awx credential_input_sources: the following arguments are required: action
```

credential_input_sources list

```
usage: awx credential_input_sources list [-h] [--all] [--order_by ORDER_BY] [-f {json,
↪yaml,jq,human}] [--filter TEXT] [--conf.color BOOLEAN] [-v]
                                         [--type {credential_input_source}] [--created␣
↪CREATED] [--modified MODIFIED] [--description TEXT] [--input_field_name TEXT]
                                         [--metadata JSON/YAML] [--target_credential ID]␣
↪[--source_credential ID]
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --type {credential_input_source}
                        only list credential_input_sources with the specified type
  --created CREATED     only list credential_input_sources with the specified created
  --modified MODIFIED   only list credential_input_sources with the specified modified
  --description TEXT    only list credential_input_sources with the specified description
  --input_field_name TEXT
                        only list credential_input_sources with the specified input_
↪field_name
  --metadata JSON/YAML  only list credential_input_sources with the specified metadata
  --target_credential ID
                        only list credential_input_sources with the specified target_
↪credential
  --source_credential ID
                        only list credential_input_sources with the specified source_
↪credential

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

credential_input_sources create

```
usage: awx credential_input_sources create [-h] [--description TEXT] --input_field_name␣
↪TEXT [--metadata JSON/YAML] --target_credential ID --source_credential ID

required arguments:
  --input_field_name TEXT
  --target_credential ID
                        the ID of the associated target_credential
  --source_credential ID
                        the ID of the associated source_credential

options:
  -h, --help            show this help message and exit
  --description TEXT    Optional description of this credential input source.
  --metadata JSON/YAML  a JSON or YAML string. You can optionally specify a file path e.
↪g., @path/to/file.yml

awx credential_input_sources create: the following arguments are required: --input_field_
↪name, --target_credential, --source_credential
```

credential_input_sources get

```
usage: awx credential_input_sources get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [-
↪-conf.color BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx credential_input_sources get: the following arguments are required: id
```

credential_input_sources modify

```
usage: awx credential_input_sources modify [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx credential_input_sources modify: the following arguments are required: id
```

credential_input_sources delete

```
usage: awx credential_input_sources delete [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx credential_input_sources delete: the following arguments are required: id
```

credential_types

```
usage: awx credential_types [-h] action ...

positional arguments:
  action
    list
    create
    get
    modify
    delete

options:
  -h, --help  show this help message and exit

awx credential_types: the following arguments are required: action
```

credential_types list

```
usage: awx credential_types list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,jq,
↪human}] [--filter TEXT] [--conf.color BOOLEAN] [-v] [--type {credential_type}]
                                 [--created CREATED] [--modified MODIFIED] [--name TEXT]␣
↪[--description TEXT]
                                 [--kind {ssh,vault,net,scm,cloud,registry,token,
↪insights,external,kubernetes,galaxy,cryptography}] [--namespace TEXT]
                                 [--managed BOOLEAN] [--inputs JSON/YAML] [--injectors␣
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
↪JSON/YAML]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --type {credential_type}
                        only list credential_types with the specified type
  --created CREATED     only list credential_types with the specified created
  --modified MODIFIED   only list credential_types with the specified modified
  --name TEXT           only list credential_types with the specified name
  --description TEXT    only list credential_types with the specified description
  --kind {ssh,vault,net,scm,cloud,registry,token,insights,external,kubernetes,galaxy,
↪cryptography}
                        only list credential_types with the specified kind
  --namespace TEXT      only list credential_types with the specified namespace
  --managed BOOLEAN     only list credential_types with the specified managed
  --inputs JSON/YAML    only list credential_types with the specified inputs
  --injectors JSON/YAML
                        only list credential_types with the specified injectors

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

credential_types create

```
usage: awx credential_types create [-h] --name TEXT [--description TEXT] --kind {net,
↪cloud} [--inputs JSON/YAML] [--injectors JSON/YAML]

required arguments:
  --name TEXT           Name of this credential type.
  --kind {net,cloud}

options:
  -h, --help            show this help message and exit
  --description TEXT    Optional description of this credential type.
  --inputs JSON/YAML    Enter inputs using either JSON or YAML syntax. Refer to the␣
↪documentation for example syntax. You can optionally specify a file path e.g.,
                        @path/to/file.yml
  --injectors JSON/YAML
                        Enter injectors using either JSON or YAML syntax. Refer to the␣
↪documentation for example syntax. You can optionally specify a file path e.g.,
                        @path/to/file.yml

awx credential_types create: the following arguments are required: --name, --kind
```

credential_types get

```
usage: awx credential_types get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--conf.
↪color BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx credential_types get: the following arguments are required: id
```

credential_types modify

```
usage: awx credential_types modify [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx credential_types modify: the following arguments are required: id
```

credential_types delete

```
usage: awx credential_types delete [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx credential_types delete: the following arguments are required: id
```

credentials

```
usage: awx credentials [-h] action ...

positional arguments:
  action
    list
    create
    get
    modify
    delete

options:
  -h, --help  show this help message and exit

awx credentials: the following arguments are required: action
```

credentials list

```
usage: awx credentials list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,jq,human}]␣
↪[--filter TEXT] [--conf.color BOOLEAN] [-v] [--type {credential}]
                            [--created CREATED] [--modified MODIFIED] [--name TEXT] [--
↪description TEXT] [--organization ID] [--credential_type ID]
                            [--managed BOOLEAN] [--inputs JSON/YAML]
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --type {credential}   only list credentials with the specified type
  --created CREATED     only list credentials with the specified created
  --modified MODIFIED   only list credentials with the specified modified
  --name TEXT           only list credentials with the specified name
  --description TEXT    only list credentials with the specified description
  --organization ID     only list credentials with the specified organization
  --credential_type ID  only list credentials with the specified credential_type
  --managed BOOLEAN     only list credentials with the specified managed
  --inputs JSON/YAML    only list credentials with the specified inputs

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

credentials create

```
usage: awx credentials create [-h] --name TEXT [--description TEXT] [--organization ID] -
↪-credential_type ID [--inputs JSON/YAML] [--user ID] [--team ID]

required arguments:
  --name TEXT           Name of this credential.
  --credential_type ID  Specify the type of credential you want to create. Refer to the␣
↪documentation for details on each type.

options:
  -h, --help            show this help message and exit
  --description TEXT    Optional description of this credential.
  --organization ID     Inherit permissions from organization roles. If provided on␣
↪creation, do not give either user or team.
  --inputs JSON/YAML    Enter inputs using either JSON or YAML syntax. Refer to the␣
↪documentation for example syntax. You can optionally specify a file path e.g.,
                        @path/to/file.yml
  --user ID             Write-only field used to add user to owner role. If provided, do␣
↪not give either team or organization. Only valid for creation.
  --team ID             Write-only field used to add team to owner role. If provided, do␣
↪not give either user or organization. Only valid for creation.

awx credentials create: the following arguments are required: --name, --credential_type
```

credentials get

```
usage: awx credentials get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--conf.color␣
↪BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx credentials get: the following arguments are required: id
```

credentials modify

```
usage: awx credentials modify [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx credentials modify: the following arguments are required: id
```

credentials delete

```
usage: awx credentials delete [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx credentials delete: the following arguments are required: id
```

execution_environments

```
usage: awx execution_environments [-h] action ...

positional arguments:
  action
    list
    create
    get
    modify
    delete

options:
  -h, --help  show this help message and exit

awx execution_environments: the following arguments are required: action
```

execution_environments list

```
usage: awx execution_environments list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,
↪jq,human}] [--filter TEXT] [--conf.color BOOLEAN] [-v]
                                       [--type {execution_environment}] [--created␣
↪CREATED] [--modified MODIFIED] [--name TEXT] [--description TEXT]
                                       [--organization ID] [--image TEXT] [--managed␣
↪BOOLEAN] [--credential ID] [--pull {,always,missing,never}]

options:
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --type {execution_environment}
                        only list execution_environments with the specified type
  --created CREATED     only list execution_environments with the specified created
  --modified MODIFIED   only list execution_environments with the specified modified
  --name TEXT           only list execution_environments with the specified name
  --description TEXT    only list execution_environments with the specified description
  --organization ID     only list execution_environments with the specified organization
  --image TEXT          only list execution_environments with the specified image
  --managed BOOLEAN     only list execution_environments with the specified managed
  --credential ID       only list execution_environments with the specified credential
  --pull {,always,missing,never}
                        only list execution_environments with the specified pull

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

execution_environments create

```
usage: awx execution_environments create [-h] --name TEXT [--description TEXT] [--
↪organization ID] --image TEXT [--credential ID] [--pull {,always,missing,never}]

required arguments:
  --name TEXT           Name of this execution environment.
  --image TEXT          The full image location, including the container registry, image␣
↪name, and version tag.

options:
  -h, --help            show this help message and exit
  --description TEXT    Optional description of this execution environment.
  --organization ID     The organization used to determine access to this execution␣
↪environment.
  --credential ID       the ID of the associated credential
  --pull {,always,missing,never}
                        Pull image before running?

awx execution_environments create: the following arguments are required: --name, --image
```

execution_environments get

```
usage: awx execution_environments get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--
↪conf.color BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx execution_environments get: the following arguments are required: id
```

execution_environments modify

```
usage: awx execution_environments modify [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx execution_environments modify: the following arguments are required: id
```

execution_environments delete

```
usage: awx execution_environments delete [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx execution_environments delete: the following arguments are required: id
```

export

```
usage: awx export > exportfile

export resources to stdout

options:
  -h, --help            show this help message and exit

resources:
  --users [USERS ...]
  --organizations [ORGANIZATIONS ...]
  --teams [TEAMS ...]
  --credential_types [CREDENTIAL_TYPES ...]
  --credentials [CREDENTIALS ...]
  --notification_templates [NOTIFICATION_TEMPLATES ...]
  --projects [PROJECTS ...]
  --inventory [INVENTORY ...]
  --inventory_sources [INVENTORY_SOURCES ...]
  --job_templates [JOB_TEMPLATES ...]
  --workflow_job_templates [WORKFLOW_JOB_TEMPLATES ...]
  --execution_environments [EXECUTION_ENVIRONMENTS ...]
  --applications [APPLICATIONS ...]
  --schedules [SCHEDULES ...]

input/output formatting:
  -f {json,yaml}, --conf.format {json,yaml}
                        specify a format for the input and output
  -v, --verbose         print debug-level logs, including requests made
```

groups

```
usage: awx groups [-h] action ...

positional arguments:
  action
    list
    create
    get
    modify
    delete

options:
  -h, --help  show this help message and exit

awx groups: the following arguments are required: action
```

groups list

```
usage: awx groups list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,jq,human}] [--
↪filter TEXT] [--conf.color BOOLEAN] [-v] [--type {group}] [--created CREATED]
                       [--modified MODIFIED] [--name TEXT] [--description TEXT] [--
↪inventory ID] [--variables JSON/YAML]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --type {group}        only list groups with the specified type
  --created CREATED     only list groups with the specified created
  --modified MODIFIED   only list groups with the specified modified
  --name TEXT           only list groups with the specified name
  --description TEXT    only list groups with the specified description
  --inventory ID        only list groups with the specified inventory
  --variables JSON/YAML
                        only list groups with the specified variables

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

groups create

```
usage: awx groups create [-h] --name TEXT [--description TEXT] --inventory ID [--
↪variables JSON/YAML]

required arguments:
  --name TEXT           Name of this group.
  --inventory ID        the ID of the associated inventory

options:
  -h, --help            show this help message and exit
  --description TEXT    Optional description of this group.
  --variables JSON/YAML
                        Group variables in JSON or YAML format. You can optionally␣
↪specify a file path e.g., @path/to/file.yml
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
awx groups create: the following arguments are required: --name, --inventory
```

groups get

```
usage: awx groups get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--conf.color␣
↪BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx groups get: the following arguments are required: id
```

groups modify

```
usage: awx groups modify [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx groups modify: the following arguments are required: id
```

groups delete

```
usage: awx groups delete [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx groups delete: the following arguments are required: id
```

host_metric_summary_monthly

```
usage: awx host_metric_summary_monthly [-h] action ...

positional arguments:
  action
    list
    get

options:
  -h, --help  show this help message and exit

awx host_metric_summary_monthly: the following arguments are required: action
```

host_metric_summary_monthly list

```
usage: awx host_metric_summary_monthly list [-h] [--all] [--order_by ORDER_BY] [-f {json,
↪yaml,jq,human}] [--filter TEXT] [--conf.color BOOLEAN] [-v] [--date DATE]
                                            [--license_consumed INTEGER] [--license_
↪capacity INTEGER] [--hosts_added INTEGER] [--hosts_deleted INTEGER]
                                            [--indirectly_managed_hosts INTEGER]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --date DATE           only list host_metric_summary_monthly with the specified date
  --license_consumed INTEGER
                        only list host_metric_summary_monthly with the specified license_
↪consumed
  --license_capacity INTEGER
                        only list host_metric_summary_monthly with the specified license_
↪capacity
  --hosts_added INTEGER
                        only list host_metric_summary_monthly with the specified hosts_
↪added
  --hosts_deleted INTEGER
                        only list host_metric_summary_monthly with the specified hosts_
↪deleted
  --indirectly_managed_hosts INTEGER
                        only list host_metric_summary_monthly with the specified␣
↪indirectly_managed_hosts

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

host_metric_summary_monthly get

```
usage: awx host_metric_summary_monthly get [-h] [-f {json,yaml,jq,human}] [--filter␣
↪TEXT] [--conf.color BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx host_metric_summary_monthly get: the following arguments are required: id
```

host_metrics

```
usage: awx host_metrics [-h] action ...

positional arguments:
  action
    list
    get
    delete

options:
  -h, --help  show this help message and exit

awx host_metrics: the following arguments are required: action
```

host_metrics list

```
usage: awx host_metrics list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,jq,human}
↪] [--filter TEXT] [--conf.color BOOLEAN] [-v] [--hostname TEXT]
                             [--first_automation FIRST_AUTOMATION] [--last_automation␣
↪LAST_AUTOMATION] [--last_deleted LAST_DELETED] [--automated_counter INTEGER]
                             [--deleted_counter INTEGER] [--deleted BOOLEAN] [--used_in_
↪inventories INTEGER]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --hostname TEXT       only list host_metrics with the specified hostname
  --first_automation FIRST_AUTOMATION
                        only list host_metrics with the specified first_automation
  --last_automation LAST_AUTOMATION
                        only list host_metrics with the specified last_automation
  --last_deleted LAST_DELETED
                        only list host_metrics with the specified last_deleted
  --automated_counter INTEGER
                        only list host_metrics with the specified automated_counter
  --deleted_counter INTEGER
                        only list host_metrics with the specified deleted_counter
  --deleted BOOLEAN     only list host_metrics with the specified deleted
  --used_in_inventories INTEGER
                        only list host_metrics with the specified used_in_inventories

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

host_metrics get

```
usage: awx host_metrics get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--conf.color␣
↪BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx host_metrics get: the following arguments are required: id
```

host_metrics delete

```
usage: awx host_metrics delete [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx host_metrics delete: the following arguments are required: id
```

hosts

```
usage: awx hosts [-h] action ...

positional arguments:
  action
    list
    create
    get
    modify
    delete

options:
  -h, --help  show this help message and exit

awx hosts: the following arguments are required: action
```

hosts list

```
usage: awx hosts list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,jq,human}] [--
↪filter TEXT] [--conf.color BOOLEAN] [-v] [--type {host}] [--created CREATED]
                      [--modified MODIFIED] [--name TEXT] [--description TEXT] [--
↪inventory ID] [--enabled BOOLEAN] [--instance_id TEXT] [--variables JSON/YAML]
                      [--last_job ID] [--last_job_host_summary ID] [--ansible_facts_
↪modified ANSIBLE_FACTS_MODIFIED]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --type {host}         only list hosts with the specified type
  --created CREATED     only list hosts with the specified created
  --modified MODIFIED   only list hosts with the specified modified
  --name TEXT           only list hosts with the specified name
  --description TEXT    only list hosts with the specified description
  --inventory ID        only list hosts with the specified inventory
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  --enabled BOOLEAN     only list hosts with the specified enabled
  --instance_id TEXT    only list hosts with the specified instance_id
  --variables JSON/YAML
                        only list hosts with the specified variables
  --last_job ID         only list hosts with the specified last_job
  --last_job_host_summary ID
                        only list hosts with the specified last_job_host_summary
  --ansible_facts_modified ANSIBLE_FACTS_MODIFIED
                        only list hosts with the specified ansible_facts_modified

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

hosts create

```
usage: awx hosts create [-h] --name TEXT [--description TEXT] --inventory ID [--enabled␣
↪BOOLEAN] [--instance_id TEXT] [--variables JSON/YAML]

required arguments:
  --name TEXT           Name of this host.
  --inventory ID        the ID of the associated inventory

options:
  -h, --help            show this help message and exit
  --description TEXT    Optional description of this host.
  --enabled BOOLEAN     Is this host online and available for running jobs?
  --instance_id TEXT    The value used by the remote inventory source to uniquely␣
↪identify the host
  --variables JSON/YAML
                        Host variables in JSON or YAML format. You can optionally␣
↪specify a file path e.g., @path/to/file.yml

awx hosts create: the following arguments are required: --name, --inventory
```

hosts get

```
usage: awx hosts get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--conf.color␣
↪BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx hosts get: the following arguments are required: id
```

hosts modify

```
usage: awx hosts modify [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx hosts modify: the following arguments are required: id
```

hosts delete

```
usage: awx hosts delete [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx hosts delete: the following arguments are required: id
```

import

```
usage: awx import < exportfile

import resources from stdin

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml}, --conf.format {json,yaml}
                        specify a format for the input and output
  -v, --verbose         print debug-level logs, including requests made
```

instance_groups

```
usage: awx instance_groups [-h] action ...

positional arguments:
  action
    list
    create
    get
    modify
    delete

options:
  -h, --help  show this help message and exit

awx instance_groups: the following arguments are required: action
```

instance_groups list

```
usage: awx instance_groups list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,jq,
↪human}] [--filter TEXT] [--conf.color BOOLEAN] [-v] [--type {instance_group}]
                                [--name TEXT] [--created CREATED] [--modified MODIFIED]␣
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
↪[--max_concurrent_jobs INTEGER] [--max_forks INTEGER]
                                [--is_container_group BOOLEAN] [--credential ID] [--
↪policy_instance_percentage INTEGER] [--policy_instance_minimum INTEGER]
                                [--policy_instance_list JSON/YAML] [--pod_spec_override␣
↪TEXT]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --type {instance_group}
                        only list instance_groups with the specified type
  --name TEXT           only list instance_groups with the specified name
  --created CREATED     only list instance_groups with the specified created
  --modified MODIFIED   only list instance_groups with the specified modified
  --max_concurrent_jobs INTEGER
                        only list instance_groups with the specified max_concurrent_jobs
  --max_forks INTEGER   only list instance_groups with the specified max_forks
  --is_container_group BOOLEAN
                        only list instance_groups with the specified is_container_group
  --credential ID       only list instance_groups with the specified credential
  --policy_instance_percentage INTEGER
                        only list instance_groups with the specified policy_instance_
↪percentage
  --policy_instance_minimum INTEGER
                        only list instance_groups with the specified policy_instance_
↪minimum
  --policy_instance_list JSON/YAML
                        only list instance_groups with the specified policy_instance_list
  --pod_spec_override TEXT
                        only list instance_groups with the specified pod_spec_override

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

instance_groups create

```
usage: awx instance_groups create [-h] --name TEXT [--max_concurrent_jobs INTEGER] [--
↪max_forks INTEGER] [--is_container_group BOOLEAN] [--credential ID]
                                  [--policy_instance_percentage INTEGER] [--policy_
↪instance_minimum INTEGER] [--policy_instance_list JSON/YAML]
                                  [--pod_spec_override TEXT]

required arguments:
  --name TEXT           Name of this instance group.

options:
  -h, --help            show this help message and exit
  --max_concurrent_jobs INTEGER
                        Maximum number of concurrent jobs to run on a group. When set to␣
↪zero, no maximum is enforced.
  --max_forks INTEGER   Maximum number of forks to execute concurrently on a group. When␣
↪set to zero, no maximum is enforced.
  --is_container_group BOOLEAN
                        Indicates whether instances in this group are containerized.
↪Containerized groups have a designated Openshift or Kubernetes cluster.
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  --credential ID       the ID of the associated credential
  --policy_instance_percentage INTEGER
                        Minimum percentage of all instances that will be automatically␣
↪assigned to this group when new instances come online.
  --policy_instance_minimum INTEGER
                        Static minimum number of Instances that will be automatically␣
↪assign to this group when new instances come online.
  --policy_instance_list JSON/YAML
                        List of exact-match Instances that will be assigned to this␣
↪group You can optionally specify a file path e.g., @path/to/file.yml
  --pod_spec_override TEXT

awx instance_groups create: the following arguments are required: --name
```

instance_groups get

```
usage: awx instance_groups get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--conf.
↪color BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx instance_groups get: the following arguments are required: id
```

instance_groups modify

```
usage: awx instance_groups modify [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx instance_groups modify: the following arguments are required: id
```

instance_groups delete

```
usage: awx instance_groups delete [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx instance_groups delete: the following arguments are required: id
```

instances

```
usage: awx instances [-h] action ...

positional arguments:
  action
    list
    create
    get
    modify

options:
  -h, --help  show this help message and exit

awx instances: the following arguments are required: action
```

instances list

```
usage: awx instances list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,jq,human}] [-
↪-filter TEXT] [--conf.color BOOLEAN] [-v] [--hostname TEXT]
                          [--type {instance}] [--uuid TEXT] [--created CREATED] [--
↪modified MODIFIED] [--last_seen LAST_SEEN]
                          [--health_check_started HEALTH_CHECK_STARTED] [--last_health_
↪check LAST_HEALTH_CHECK] [--errors TEXT]
                          [--capacity_adjustment CAPACITY_ADJUSTMENT] [--version TEXT] [-
↪-capacity INTEGER] [--cpu CPU] [--memory INTEGER] [--cpu_capacity INTEGER]
                          [--mem_capacity INTEGER] [--enabled BOOLEAN] [--managed_by_
↪policy BOOLEAN] [--node_type {control,execution,hybrid,hop}]
                          [--node_state {provisioning,provision-fail,installed,ready,
↪unavailable,deprovisioning,deprovision-fail}] [--managed BOOLEAN]
                          [--ip_address TEXT]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --hostname TEXT       only list instances with the specified hostname
  --type {instance}     only list instances with the specified type
  --uuid TEXT           only list instances with the specified uuid
  --created CREATED     only list instances with the specified created
  --modified MODIFIED   only list instances with the specified modified
  --last_seen LAST_SEEN
                        only list instances with the specified last_seen
  --health_check_started HEALTH_CHECK_STARTED
                        only list instances with the specified health_check_started
  --last_health_check LAST_HEALTH_CHECK
                        only list instances with the specified last_health_check
  --errors TEXT         only list instances with the specified errors
  --capacity_adjustment CAPACITY_ADJUSTMENT
                        only list instances with the specified capacity_adjustment
  --version TEXT        only list instances with the specified version
  --capacity INTEGER    only list instances with the specified capacity
  --cpu CPU             only list instances with the specified cpu
  --memory INTEGER      only list instances with the specified memory
  --cpu_capacity INTEGER
                        only list instances with the specified cpu_capacity
  --mem_capacity INTEGER
                        only list instances with the specified mem_capacity
  --enabled BOOLEAN     only list instances with the specified enabled
  --managed_by_policy BOOLEAN
                        only list instances with the specified managed_by_policy
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  --node_type {control,execution,hybrid,hop}
                        only list instances with the specified node_type
  --node_state {provisioning,provision-fail,installed,ready,unavailable,deprovisioning,
↪deprovision-fail}
                        only list instances with the specified node_state
  --managed BOOLEAN     only list instances with the specified managed
  --ip_address TEXT     only list instances with the specified ip_address

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

instances create

```
usage: awx instances create [-h] --hostname TEXT [--capacity_adjustment CAPACITY_
↪ADJUSTMENT] [--enabled BOOLEAN] [--managed_by_policy BOOLEAN]
                            [--node_type {control,execution,hybrid,hop}]
                            [--node_state {provisioning,provision-fail,installed,ready,
↪unavailable,deprovisioning,deprovision-fail}] [--peers PEERS]
                            [--listener_port INTEGER] [--peers_from_control_nodes␣
↪BOOLEAN]

required arguments:
  --hostname TEXT

options:
  -h, --help            show this help message and exit
  --capacity_adjustment CAPACITY_ADJUSTMENT
  --enabled BOOLEAN
  --managed_by_policy BOOLEAN
  --node_type {control,execution,hybrid,hop}
                        Role that this node plays in the mesh.
  --node_state {provisioning,provision-fail,installed,ready,unavailable,deprovisioning,
↪deprovision-fail}
                        Indicates the current life cycle stage of this instance.
  --peers PEERS         Primary keys of receptor addresses to peer to.
  --listener_port INTEGER
  --peers_from_control_nodes BOOLEAN

awx instances create: the following arguments are required: --hostname
```

instances get

```
usage: awx instances get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--conf.color␣
↪BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
awx instances get: the following arguments are required: id
```

instances modify

```
usage: awx instances modify [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx instances modify: the following arguments are required: id
```

inventory

```
usage: awx inventory [-h] action ...

positional arguments:
  action
    list
    create
    get
    modify
    delete

options:
  -h, --help  show this help message and exit

awx inventory: the following arguments are required: action
```

inventory list

```
usage: awx inventory list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,jq,human}] [-
↪-filter TEXT] [--conf.color BOOLEAN] [-v] [--type {inventory}]
                          [--created CREATED] [--modified MODIFIED] [--name TEXT] [--
↪description TEXT] [--organization ID] [--kind {,smart,constructed}]
                          [--host_filter TEXT] [--variables JSON/YAML] [--has_active_
↪failures BOOLEAN] [--total_hosts INTEGER] [--hosts_with_active_failures INTEGER]
                          [--total_groups INTEGER] [--has_inventory_sources BOOLEAN] [--
↪total_inventory_sources INTEGER] [--inventory_sources_with_failures INTEGER]
                          [--pending_deletion BOOLEAN] [--prevent_instance_group_
↪fallback BOOLEAN]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --type {inventory}    only list inventory with the specified type
  --created CREATED     only list inventory with the specified created
  --modified MODIFIED   only list inventory with the specified modified
  --name TEXT           only list inventory with the specified name
  --description TEXT    only list inventory with the specified description
  --organization ID     only list inventory with the specified organization
  --kind {,smart,constructed}
                        only list inventory with the specified kind
  --host_filter TEXT    only list inventory with the specified host_filter
  --variables JSON/YAML
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
                        only list inventory with the specified variables
  --has_active_failures BOOLEAN
                        only list inventory with the specified has_active_failures
  --total_hosts INTEGER
                        only list inventory with the specified total_hosts
  --hosts_with_active_failures INTEGER
                        only list inventory with the specified hosts_with_active_failures
  --total_groups INTEGER
                        only list inventory with the specified total_groups
  --has_inventory_sources BOOLEAN
                        only list inventory with the specified has_inventory_sources
  --total_inventory_sources INTEGER
                        only list inventory with the specified total_inventory_sources
  --inventory_sources_with_failures INTEGER
                        only list inventory with the specified inventory_sources_with_
↪failures
  --pending_deletion BOOLEAN
                        only list inventory with the specified pending_deletion
  --prevent_instance_group_fallback BOOLEAN
                        only list inventory with the specified prevent_instance_group_
↪fallback

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

inventory create

```
usage: awx inventory create [-h] --name TEXT [--description TEXT] --organization ID [--
↪kind {,smart,constructed}] [--host_filter TEXT] [--variables JSON/YAML]
                            [--prevent_instance_group_fallback BOOLEAN]

required arguments:
  --name TEXT           Name of this inventory.
  --organization ID     Organization containing this inventory.

options:
  -h, --help            show this help message and exit
  --description TEXT    Optional description of this inventory.
  --kind {,smart,constructed}
                        Kind of inventory being represented.
  --host_filter TEXT    Filter that will be applied to the hosts of this inventory.
  --variables JSON/YAML
                        Inventory variables in JSON or YAML format. You can optionally␣
↪specify a file path e.g., @path/to/file.yml
  --prevent_instance_group_fallback BOOLEAN
                        If enabled, the inventory will prevent adding any organization␣
↪instance groups to the list of preferred instances groups to run associated job
                        templates on.If this setting is enabled and you provided an␣
↪empty list, the global instance groups will be applied.

awx inventory create: the following arguments are required: --name, --organization
```

inventory get

```
usage: awx inventory get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--conf.color␣
↪BOOLEAN] [-v] id

positional arguments:
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx inventory get: the following arguments are required: id
```

inventory modify

```
usage: awx inventory modify [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx inventory modify: the following arguments are required: id
```

inventory delete

```
usage: awx inventory delete [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx inventory delete: the following arguments are required: id
```

inventory_sources

```
usage: awx inventory_sources [-h] action ...

positional arguments:
  action
    list
    create
    get
    modify
    delete
    update
    associate
    disassociate

options:
  -h, --help    show this help message and exit

awx inventory_sources: the following arguments are required: action
```

inventory_sources list

```
usage: awx inventory_sources list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,jq,
↪human}] [--filter TEXT] [--conf.color BOOLEAN] [-v] [--type {inventory_source}]
                                  [--created CREATED] [--modified MODIFIED] [--name␣
↪TEXT] [--description TEXT]
                                  [--source {file,constructed,scm,ec2,gce,azure_rm,
↪vmware,satellite6,openstack,rhv,controller,insights,terraform,openshift_virtualization}
↪]
                                  [--source_path TEXT] [--source_vars TEXT] [--scm_
↪branch TEXT] [--enabled_var TEXT] [--enabled_value TEXT] [--host_filter TEXT]
                                  [--overwrite BOOLEAN] [--overwrite_vars BOOLEAN] [--
↪custom_virtualenv TEXT] [--timeout INTEGER] [--verbosity {0,1,2}] [--limit TEXT]
                                  [--last_job_run LAST_JOB_RUN] [--last_job_failed␣
↪BOOLEAN] [--next_job_run NEXT_JOB_RUN]
                                  [--status {new,pending,waiting,running,successful,
↪failed,error,canceled,never updated,none}] [--execution_environment ID]
                                  [--inventory ID] [--update_on_launch BOOLEAN] [--
↪update_cache_timeout INTEGER] [--source_project ID]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --type {inventory_source}
                        only list inventory_sources with the specified type
  --created CREATED     only list inventory_sources with the specified created
  --modified MODIFIED   only list inventory_sources with the specified modified
  --name TEXT           only list inventory_sources with the specified name
  --description TEXT    only list inventory_sources with the specified description
  --source {file,constructed,scm,ec2,gce,azure_rm,vmware,satellite6,openstack,rhv,
↪controller,insights,terraform,openshift_virtualization}
                        only list inventory_sources with the specified source
  --source_path TEXT    only list inventory_sources with the specified source_path
  --source_vars TEXT    only list inventory_sources with the specified source_vars
  --scm_branch TEXT     only list inventory_sources with the specified scm_branch
  --enabled_var TEXT    only list inventory_sources with the specified enabled_var
  --enabled_value TEXT  only list inventory_sources with the specified enabled_value
  --host_filter TEXT    only list inventory_sources with the specified host_filter
  --overwrite BOOLEAN   only list inventory_sources with the specified overwrite
  --overwrite_vars BOOLEAN
                        only list inventory_sources with the specified overwrite_vars
  --custom_virtualenv TEXT
                        only list inventory_sources with the specified custom_virtualenv
  --timeout INTEGER     only list inventory_sources with the specified timeout
  --verbosity {0,1,2}   only list inventory_sources with the specified verbosity
  --limit TEXT          only list inventory_sources with the specified limit
  --last_job_run LAST_JOB_RUN
                        only list inventory_sources with the specified last_job_run
  --last_job_failed BOOLEAN
                        only list inventory_sources with the specified last_job_failed
  --next_job_run NEXT_JOB_RUN
                        only list inventory_sources with the specified next_job_run
  --status {new,pending,waiting,running,successful,failed,error,canceled,never updated,
↪none}
                        only list inventory_sources with the specified status
  --execution_environment ID
                        only list inventory_sources with the specified execution_
↪environment
  --inventory ID        only list inventory_sources with the specified inventory
  --update_on_launch BOOLEAN
                        only list inventory_sources with the specified update_on_launch
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  --update_cache_timeout INTEGER
                        only list inventory_sources with the specified update_cache_
↪timeout
  --source_project ID   only list inventory_sources with the specified source_project

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

inventory_sources create

```
usage: awx inventory_sources create [-h] --name TEXT [--description TEXT]
                                    [--source {file,constructed,scm,ec2,gce,azure_rm,
↪vmware,satellite6,openstack,rhv,controller,insights,terraform,openshift_virtualization}
↪]
                                    [--source_path TEXT] [--source_vars TEXT] [--scm_
↪branch TEXT] [--credential INTEGER] [--enabled_var TEXT] [--enabled_value TEXT]
                                    [--host_filter TEXT] [--overwrite BOOLEAN] [--
↪overwrite_vars BOOLEAN] [--timeout INTEGER] [--verbosity {0,1,2}] [--limit TEXT]
                                    [--execution_environment ID] --inventory ID [--
↪update_on_launch BOOLEAN] [--update_cache_timeout INTEGER] [--source_project ID]

required arguments:
  --name TEXT           Name of this inventory source.
  --inventory ID        the ID of the associated inventory

options:
  -h, --help            show this help message and exit
  --description TEXT    Optional description of this inventory source.
  --source {file,constructed,scm,ec2,gce,azure_rm,vmware,satellite6,openstack,rhv,
↪controller,insights,terraform,openshift_virtualization}
  --source_path TEXT
  --source_vars TEXT    Inventory source variables in YAML or JSON format.
  --scm_branch TEXT     Inventory source SCM branch. Project default used if blank. Only␣
↪allowed if project allow_override field is set to true.
  --credential INTEGER  Cloud credential to use for inventory updates.
  --enabled_var TEXT    Retrieve the enabled state from the given dict of host variables.
↪ The enabled variable may be specified as "foo.bar", in which case the lookup
                        will traverse into nested dicts, equivalent to: from_dict.get(
↪"foo", {}).get("bar", default)
  --enabled_value TEXT  Only used when enabled_var is set. Value when the host is␣
↪considered enabled. For example if enabled_var="status.power_state"and
                        enabled_value="powered_on" with host variables:{ "status": {
↪"power_state": "powered_on", "created": "2020-08-04T18:13:04+00:00", "healthy":
                        true }, "name": "foobar", "ip_address": "192.168.2.1"}The host␣
↪would be marked enabled. If power_state where any value other than powered_on
                        then the host would be disabled when imported. If the key is not␣
↪found then the host will be enabled
  --host_filter TEXT    This field is deprecated and will be removed in a future release.
↪ Regex where only matching hosts will be imported.
  --overwrite BOOLEAN   Overwrite local groups and hosts from remote inventory source.
  --overwrite_vars BOOLEAN
                        Overwrite local variables from remote inventory source.
  --timeout INTEGER     The amount of time (in seconds) to run before the task is␣
↪canceled.
  --verbosity {0,1,2}
  --limit TEXT          Enter host, group or pattern match
  --execution_environment ID
                        The container image to be used for execution.
  --update_on_launch BOOLEAN
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  --update_cache_timeout INTEGER
  --source_project ID   Project containing inventory file used as source.

awx inventory_sources create: the following arguments are required: --name, --inventory
```

inventory_sources get

```
usage: awx inventory_sources get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--conf.
↪color BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx inventory_sources get: the following arguments are required: id
```

inventory_sources modify

```
usage: awx inventory_sources modify [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx inventory_sources modify: the following arguments are required: id
```

inventory_sources delete

```
usage: awx inventory_sources delete [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx inventory_sources delete: the following arguments are required: id
```

inventory_sources update

```
usage: awx inventory_sources update [-h] [--monitor] [--action-timeout ACTION_TIMEOUT] [-
↪-wait] [--interval INTERVAL] id

positional arguments:
  id

options:
  -h, --help            show this help message and exit
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  --monitor             If set, prints stdout of the launched job until it finishes.
  --action-timeout ACTION_TIMEOUT
                        If set with --monitor or --wait, time out waiting on job␣
↪completion.
  --wait                If set, waits until the launched job finishes.
  --interval INTERVAL   If set with --monitor or --wait, amount of time to wait in␣
↪seconds between api calls. Minimum value is 2.5 seconds to avoid overwhelming the
                        api

awx inventory_sources update: the following arguments are required: id
```

inventory_sources associate

```
usage: awx inventory_sources associate [-h] (--start_notification  | --success_
↪notification  | --failure_notification ) id

positional arguments:
  id

options:
  -h, --help            show this help message and exit
  --start_notification
                        The ID (or name) of the notification_template to associate
  --success_notification
                        The ID (or name) of the notification_template to associate
  --failure_notification
                        The ID (or name) of the notification_template to associate

awx inventory_sources associate: the following arguments are required: id
```

inventory_sources disassociate

```
usage: awx inventory_sources disassociate [-h] (--start_notification  | --success_
↪notification  | --failure_notification ) id

positional arguments:
  id

options:
  -h, --help            show this help message and exit
  --start_notification
                        The ID (or name) of the notification_template to disassociate
  --success_notification
                        The ID (or name) of the notification_template to disassociate
  --failure_notification
                        The ID (or name) of the notification_template to disassociate

awx inventory_sources disassociate: the following arguments are required: id
```

inventory_updates

```
usage: awx inventory_updates [-h] action ...

positional arguments:
  action
    list
    get
    delete
    stdout

options:
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  -h, --help  show this help message and exit

awx inventory_updates: the following arguments are required: action
```

inventory_updates list

```
usage: awx inventory_updates list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,jq,
↪human}] [--filter TEXT] [--conf.color BOOLEAN] [-v] [--type {inventory_update}]
                                  [--created CREATED] [--modified MODIFIED] [--name␣
↪TEXT] [--description TEXT] [--unified_job_template ID]
                                  [--launch_type {manual,relaunch,callback,scheduled,
↪dependency,workflow,webhook,sync,scm}]
                                  [--status {new,pending,waiting,running,successful,
↪failed,error,canceled}] [--execution_environment ID] [--failed BOOLEAN]
                                  [--started STARTED] [--finished FINISHED] [--canceled_
↪on CANCELED_ON] [--elapsed ELAPSED] [--job_explanation TEXT]
                                  [--execution_node TEXT] [--controller_node TEXT] [--
↪work_unit_id TEXT]
                                  [--source {file,constructed,scm,ec2,gce,azure_rm,
↪vmware,satellite6,openstack,rhv,controller,insights,terraform,openshift_virtualization}
↪]
                                  [--source_path TEXT] [--source_vars TEXT] [--scm_
↪branch TEXT] [--enabled_var TEXT] [--enabled_value TEXT] [--host_filter TEXT]
                                  [--overwrite BOOLEAN] [--overwrite_vars BOOLEAN] [--
↪custom_virtualenv TEXT] [--timeout INTEGER] [--verbosity {0,1,2}] [--limit TEXT]
                                  [--inventory ID] [--inventory_source ID] [--license_
↪error BOOLEAN] [--org_host_limit_error BOOLEAN] [--source_project_update ID]
                                  [--instance_group ID] [--scm_revision TEXT]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --type {inventory_update}
                        only list inventory_updates with the specified type
  --created CREATED     only list inventory_updates with the specified created
  --modified MODIFIED   only list inventory_updates with the specified modified
  --name TEXT           only list inventory_updates with the specified name
  --description TEXT    only list inventory_updates with the specified description
  --unified_job_template ID
                        only list inventory_updates with the specified unified_job_
↪template
  --launch_type {manual,relaunch,callback,scheduled,dependency,workflow,webhook,sync,scm}
                        only list inventory_updates with the specified launch_type
  --status {new,pending,waiting,running,successful,failed,error,canceled}
                        only list inventory_updates with the specified status
  --execution_environment ID
                        only list inventory_updates with the specified execution_
↪environment
  --failed BOOLEAN      only list inventory_updates with the specified failed
  --started STARTED     only list inventory_updates with the specified started
  --finished FINISHED   only list inventory_updates with the specified finished
  --canceled_on CANCELED_ON
                        only list inventory_updates with the specified canceled_on
  --elapsed ELAPSED     only list inventory_updates with the specified elapsed
  --job_explanation TEXT
                        only list inventory_updates with the specified job_explanation
  --execution_node TEXT
                        only list inventory_updates with the specified execution_node
  --controller_node TEXT
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
                        only list inventory_updates with the specified controller_node
  --work_unit_id TEXT   only list inventory_updates with the specified work_unit_id
  --source {file,constructed,scm,ec2,gce,azure_rm,vmware,satellite6,openstack,rhv,
↪controller,insights,terraform,openshift_virtualization}
                        only list inventory_updates with the specified source
  --source_path TEXT    only list inventory_updates with the specified source_path
  --source_vars TEXT    only list inventory_updates with the specified source_vars
  --scm_branch TEXT     only list inventory_updates with the specified scm_branch
  --enabled_var TEXT    only list inventory_updates with the specified enabled_var
  --enabled_value TEXT  only list inventory_updates with the specified enabled_value
  --host_filter TEXT    only list inventory_updates with the specified host_filter
  --overwrite BOOLEAN   only list inventory_updates with the specified overwrite
  --overwrite_vars BOOLEAN
                        only list inventory_updates with the specified overwrite_vars
  --custom_virtualenv TEXT
                        only list inventory_updates with the specified custom_virtualenv
  --timeout INTEGER     only list inventory_updates with the specified timeout
  --verbosity {0,1,2}   only list inventory_updates with the specified verbosity
  --limit TEXT          only list inventory_updates with the specified limit
  --inventory ID        only list inventory_updates with the specified inventory
  --inventory_source ID
                        only list inventory_updates with the specified inventory_source
  --license_error BOOLEAN
                        only list inventory_updates with the specified license_error
  --org_host_limit_error BOOLEAN
                        only list inventory_updates with the specified org_host_limit_
↪error
  --source_project_update ID
                        only list inventory_updates with the specified source_project_
↪update
  --instance_group ID   only list inventory_updates with the specified instance_group
  --scm_revision TEXT   only list inventory_updates with the specified scm_revision

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

inventory_updates get

```
usage: awx inventory_updates get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--conf.
↪color BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx inventory_updates get: the following arguments are required: id
```

inventory_updates delete

```
usage: awx inventory_updates delete [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx inventory_updates delete: the following arguments are required: id
```

inventory_updates stdout

```
usage: awx inventory_updates stdout [-h] id

positional arguments:
  id

options:
  -h, --help  show this help message and exit

awx inventory_updates stdout: the following arguments are required: id
```

job_templates

```
usage: awx job_templates [-h] action ...

positional arguments:
  action
    list
    create
    get
    modify
    delete
    launch
    associate
    disassociate

options:
  -h, --help    show this help message and exit

awx job_templates: the following arguments are required: action
```

job_templates list

```
usage: awx job_templates list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,jq,human}
↪] [--filter TEXT] [--conf.color BOOLEAN] [-v] [--type {job_template}]
                              [--created CREATED] [--modified MODIFIED] [--name TEXT] [--
↪description TEXT] [--job_type {run,check}] [--inventory ID] [--project ID]
                              [--playbook TEXT] [--scm_branch TEXT] [--forks INTEGER] [--
↪limit TEXT] [--verbosity {0,1,2,3,4,5}] [--extra_vars JSON/YAML]
                              [--job_tags TEXT] [--force_handlers BOOLEAN] [--skip_tags␣
↪TEXT] [--start_at_task TEXT] [--timeout INTEGER] [--use_fact_cache BOOLEAN]
                              [--organization ID] [--last_job_run LAST_JOB_RUN] [--last_
↪job_failed BOOLEAN] [--next_job_run NEXT_JOB_RUN]
                              [--status {new,pending,waiting,running,successful,failed,
↪error,canceled,never updated}] [--execution_environment ID]
                              [--host_config_key TEXT] [--ask_scm_branch_on_launch␣
↪BOOLEAN] [--ask_diff_mode_on_launch BOOLEAN] [--ask_variables_on_launch BOOLEAN]
                              [--ask_limit_on_launch BOOLEAN] [--ask_tags_on_launch␣
↪BOOLEAN] [--ask_skip_tags_on_launch BOOLEAN] [--ask_job_type_on_launch BOOLEAN]
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
                              [--ask_verbosity_on_launch BOOLEAN] [--ask_inventory_on_
↪launch BOOLEAN] [--ask_credential_on_launch BOOLEAN]
                              [--ask_execution_environment_on_launch BOOLEAN] [--ask_
↪labels_on_launch BOOLEAN] [--ask_forks_on_launch BOOLEAN]
                              [--ask_job_slice_count_on_launch BOOLEAN] [--ask_timeout_
↪on_launch BOOLEAN] [--ask_instance_groups_on_launch BOOLEAN]
                              [--survey_enabled BOOLEAN] [--become_enabled BOOLEAN] [--
↪diff_mode BOOLEAN] [--allow_simultaneous BOOLEAN] [--custom_virtualenv TEXT]
                              [--job_slice_count INTEGER] [--webhook_service {,gitflic,
↪github,gitlab,bitbucket_dc}] [--webhook_credential ID]
                              [--prevent_instance_group_fallback BOOLEAN]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --type {job_template}
                        only list job_templates with the specified type
  --created CREATED     only list job_templates with the specified created
  --modified MODIFIED   only list job_templates with the specified modified
  --name TEXT           only list job_templates with the specified name
  --description TEXT    only list job_templates with the specified description
  --job_type {run,check}
                        only list job_templates with the specified job_type
  --inventory ID        only list job_templates with the specified inventory
  --project ID          only list job_templates with the specified project
  --playbook TEXT       only list job_templates with the specified playbook
  --scm_branch TEXT     only list job_templates with the specified scm_branch
  --forks INTEGER       only list job_templates with the specified forks
  --limit TEXT          only list job_templates with the specified limit
  --verbosity {0,1,2,3,4,5}
                        only list job_templates with the specified verbosity
  --extra_vars JSON/YAML, -e JSON/YAML
                        only list job_templates with the specified extra_vars
  --job_tags TEXT       only list job_templates with the specified job_tags
  --force_handlers BOOLEAN
                        only list job_templates with the specified force_handlers
  --skip_tags TEXT      only list job_templates with the specified skip_tags
  --start_at_task TEXT  only list job_templates with the specified start_at_task
  --timeout INTEGER     only list job_templates with the specified timeout
  --use_fact_cache BOOLEAN
                        only list job_templates with the specified use_fact_cache
  --organization ID     only list job_templates with the specified organization
  --last_job_run LAST_JOB_RUN
                        only list job_templates with the specified last_job_run
  --last_job_failed BOOLEAN
                        only list job_templates with the specified last_job_failed
  --next_job_run NEXT_JOB_RUN
                        only list job_templates with the specified next_job_run
  --status {new,pending,waiting,running,successful,failed,error,canceled,never updated}
                        only list job_templates with the specified status
  --execution_environment ID
                        only list job_templates with the specified execution_environment
  --host_config_key TEXT
                        only list job_templates with the specified host_config_key
  --ask_scm_branch_on_launch BOOLEAN
                        only list job_templates with the specified ask_scm_branch_on_
↪launch
  --ask_diff_mode_on_launch BOOLEAN
                        only list job_templates with the specified ask_diff_mode_on_
↪launch
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  --ask_variables_on_launch BOOLEAN
                        only list job_templates with the specified ask_variables_on_
↪launch
  --ask_limit_on_launch BOOLEAN
                        only list job_templates with the specified ask_limit_on_launch
  --ask_tags_on_launch BOOLEAN
                        only list job_templates with the specified ask_tags_on_launch
  --ask_skip_tags_on_launch BOOLEAN
                        only list job_templates with the specified ask_skip_tags_on_
↪launch
  --ask_job_type_on_launch BOOLEAN
                        only list job_templates with the specified ask_job_type_on_launch
  --ask_verbosity_on_launch BOOLEAN
                        only list job_templates with the specified ask_verbosity_on_
↪launch
  --ask_inventory_on_launch BOOLEAN
                        only list job_templates with the specified ask_inventory_on_
↪launch
  --ask_credential_on_launch BOOLEAN
                        only list job_templates with the specified ask_credential_on_
↪launch
  --ask_execution_environment_on_launch BOOLEAN
                        only list job_templates with the specified ask_execution_
↪environment_on_launch
  --ask_labels_on_launch BOOLEAN
                        only list job_templates with the specified ask_labels_on_launch
  --ask_forks_on_launch BOOLEAN
                        only list job_templates with the specified ask_forks_on_launch
  --ask_job_slice_count_on_launch BOOLEAN
                        only list job_templates with the specified ask_job_slice_count_
↪on_launch
  --ask_timeout_on_launch BOOLEAN
                        only list job_templates with the specified ask_timeout_on_launch
  --ask_instance_groups_on_launch BOOLEAN
                        only list job_templates with the specified ask_instance_groups_
↪on_launch
  --survey_enabled BOOLEAN
                        only list job_templates with the specified survey_enabled
  --become_enabled BOOLEAN
                        only list job_templates with the specified become_enabled
  --diff_mode BOOLEAN   only list job_templates with the specified diff_mode
  --allow_simultaneous BOOLEAN
                        only list job_templates with the specified allow_simultaneous
  --custom_virtualenv TEXT
                        only list job_templates with the specified custom_virtualenv
  --job_slice_count INTEGER
                        only list job_templates with the specified job_slice_count
  --webhook_service {,gitflic,github,gitlab,bitbucket_dc}
                        only list job_templates with the specified webhook_service
  --webhook_credential ID
                        only list job_templates with the specified webhook_credential
  --prevent_instance_group_fallback BOOLEAN
                        only list job_templates with the specified prevent_instance_
↪group_fallback

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

job_templates create

```
usage: awx job_templates create [-h] --name TEXT [--description TEXT] [--job_type {run,
↪check}] [--inventory ID] --project ID --playbook TEXT [--scm_branch TEXT]
                                [--forks INTEGER] [--limit TEXT] [--verbosity {0,1,2,3,4,
↪5}] [--extra_vars JSON/YAML] [--job_tags TEXT] [--force_handlers BOOLEAN]
                                [--skip_tags TEXT] [--start_at_task TEXT] [--timeout␣
↪INTEGER] [--use_fact_cache BOOLEAN] [--execution_environment ID]
                                [--host_config_key TEXT] [--ask_scm_branch_on_launch␣
↪BOOLEAN] [--ask_diff_mode_on_launch BOOLEAN] [--ask_variables_on_launch BOOLEAN]
                                [--ask_limit_on_launch BOOLEAN] [--ask_tags_on_launch␣
↪BOOLEAN] [--ask_skip_tags_on_launch BOOLEAN] [--ask_job_type_on_launch BOOLEAN]
                                [--ask_verbosity_on_launch BOOLEAN] [--ask_inventory_on_
↪launch BOOLEAN] [--ask_credential_on_launch BOOLEAN]
                                [--ask_execution_environment_on_launch BOOLEAN] [--ask_
↪labels_on_launch BOOLEAN] [--ask_forks_on_launch BOOLEAN]
                                [--ask_job_slice_count_on_launch BOOLEAN] [--ask_timeout_
↪on_launch BOOLEAN] [--ask_instance_groups_on_launch BOOLEAN]
                                [--survey_enabled BOOLEAN] [--become_enabled BOOLEAN] [--
↪diff_mode BOOLEAN] [--allow_simultaneous BOOLEAN] [--job_slice_count INTEGER]
                                [--webhook_service {,gitflic,github,gitlab,bitbucket_dc}
↪] [--webhook_credential ID] [--prevent_instance_group_fallback BOOLEAN]

required arguments:
  --name TEXT           Name of this job template.
  --project ID          the ID of the associated project
  --playbook TEXT

options:
  -h, --help            show this help message and exit
  --description TEXT    Optional description of this job template.
  --job_type {run,check}
  --inventory ID        the ID of the associated inventory
  --scm_branch TEXT     Branch to use in job run. Project default used if blank. Only␣
↪allowed if project allow_override field is set to true.
  --forks INTEGER
  --limit TEXT
  --verbosity {0,1,2,3,4,5}
  --extra_vars JSON/YAML, -e JSON/YAML
                        a JSON or YAML string. You can optionally specify a file path e.
↪g., @path/to/file.yml
  --job_tags TEXT
  --force_handlers BOOLEAN
  --skip_tags TEXT
  --start_at_task TEXT
  --timeout INTEGER     The amount of time (in seconds) to run before the task is␣
↪canceled.
  --use_fact_cache BOOLEAN
                        If enabled, the service will act as an Ansible Fact Cache Plugin;
↪ persisting facts at the end of a playbook run to the database and caching
                        facts for use by Ansible.
  --execution_environment ID
                        The container image to be used for execution.
  --host_config_key TEXT
  --ask_scm_branch_on_launch BOOLEAN
  --ask_diff_mode_on_launch BOOLEAN
  --ask_variables_on_launch BOOLEAN
  --ask_limit_on_launch BOOLEAN
  --ask_tags_on_launch BOOLEAN
  --ask_skip_tags_on_launch BOOLEAN
  --ask_job_type_on_launch BOOLEAN
  --ask_verbosity_on_launch BOOLEAN
  --ask_inventory_on_launch BOOLEAN
  --ask_credential_on_launch BOOLEAN
  --ask_execution_environment_on_launch BOOLEAN
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  --ask_labels_on_launch BOOLEAN
  --ask_forks_on_launch BOOLEAN
  --ask_job_slice_count_on_launch BOOLEAN
  --ask_timeout_on_launch BOOLEAN
  --ask_instance_groups_on_launch BOOLEAN
  --survey_enabled BOOLEAN
  --become_enabled BOOLEAN
  --diff_mode BOOLEAN   If enabled, textual changes made to any templated files on the␣
↪host are shown in the standard output
  --allow_simultaneous BOOLEAN
  --job_slice_count INTEGER
                        The number of jobs to slice into at runtime. Will cause the Job␣
↪Template to launch a workflow if value is greater than 1.
  --webhook_service {,gitflic,github,gitlab,bitbucket_dc}
                        Service that webhook requests will be accepted from
  --webhook_credential ID
                        Personal Access Token for posting back the status to the service␣
↪API
  --prevent_instance_group_fallback BOOLEAN
                        If enabled, the job template will prevent adding any inventory␣
↪or organization instance groups to the list of preferred instances groups to
                        run on.If this setting is enabled and you provided an empty list,
↪ the global instance groups will be applied.

awx job_templates create: the following arguments are required: --name, --project, --
↪playbook
```

job_templates get

```
usage: awx job_templates get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--conf.
↪color BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx job_templates get: the following arguments are required: id
```

job_templates modify

```
usage: awx job_templates modify [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx job_templates modify: the following arguments are required: id
```

job_templates delete

```
usage: awx job_templates delete [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx job_templates delete: the following arguments are required: id
```

job_templates launch

```
usage: awx job_templates launch [-h] [--monitor] [--action-timeout ACTION_TIMEOUT] [--
↪wait] [--interval INTERVAL] [--extra_vars JSON/YAML] [--inventory ID]
                                [--scm_branch TEXT] [--limit TEXT] [--job_tags TEXT] [--
↪skip_tags TEXT] [--job_type {run,check}] [--verbosity {0,1,2,3,4,5}]
                                [--diff_mode BOOLEAN] [--credentials [ID, ID, ...]] [--
↪credential_passwords JSON/YAML] [--execution_environment ID] [--labels LABELS]
                                [--forks INTEGER] [--job_slice_count INTEGER] [--timeout␣
↪INTEGER] [--instance_groups INSTANCE_GROUPS]
                                id

positional arguments:
  id

options:
  -h, --help            show this help message and exit
  --monitor             If set, prints stdout of the launched job until it finishes.
  --action-timeout ACTION_TIMEOUT
                        If set with --monitor or --wait, time out waiting on job␣
↪completion.
  --wait                If set, waits until the launched job finishes.
  --interval INTERVAL   If set with --monitor or --wait, amount of time to wait in␣
↪seconds between api calls. Minimum value is 2.5 seconds to avoid overwhelming the
                        api
  --extra_vars JSON/YAML, -e JSON/YAML
                        a JSON or YAML string. You can optionally specify a file path e.
↪g., @path/to/file.yml
  --inventory ID        the ID of the associated inventory
  --scm_branch TEXT
  --limit TEXT
  --job_tags TEXT
  --skip_tags TEXT
  --job_type {run,check}
  --verbosity {0,1,2,3,4,5}
  --diff_mode BOOLEAN
  --credentials [ID, ID, ...]
                        a list of comma-delimited credentials to associate (IDs or␣
↪unique names)
  --credential_passwords JSON/YAML
                        a JSON or YAML string. You can optionally specify a file path e.
↪g., @path/to/file.yml
  --execution_environment ID
                        the ID of the associated execution_environment
  --labels LABELS
  --forks INTEGER
  --job_slice_count INTEGER
  --timeout INTEGER
  --instance_groups INSTANCE_GROUPS

awx job_templates launch: the following arguments are required: id
```

job_templates associate

```
usage: awx job_templates associate [-h] (--start_notification  | --success_notification ␣
↪| --failure_notification  | --credential ) id

positional arguments:
  id

options:
  -h, --help            show this help message and exit
  --start_notification
                        The ID (or name) of the notification_template to associate
  --success_notification
                        The ID (or name) of the notification_template to associate
  --failure_notification
                        The ID (or name) of the notification_template to associate
  --credential          The ID (or name) of the credential to associate

awx job_templates associate: the following arguments are required: id
```

job_templates disassociate

```
usage: awx job_templates disassociate [-h] (--start_notification  | --success_
↪notification  | --failure_notification  | --credential ) id

positional arguments:
  id

options:
  -h, --help            show this help message and exit
  --start_notification
                        The ID (or name) of the notification_template to disassociate
  --success_notification
                        The ID (or name) of the notification_template to disassociate
  --failure_notification
                        The ID (or name) of the notification_template to disassociate
  --credential          The ID (or name) of the credential to disassociate

awx job_templates disassociate: the following arguments are required: id
```

jobs

```
usage: awx jobs [-h] action ...

positional arguments:
  action
    list
    get
    delete
    stdout
    monitor

options:
  -h, --help  show this help message and exit

awx jobs: the following arguments are required: action
```

jobs list

```
usage: awx jobs list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,jq,human}] [--
↪filter TEXT] [--conf.color BOOLEAN] [-v] [--type {job}] [--created CREATED]
                     [--modified MODIFIED] [--name TEXT] [--description TEXT] [--unified_
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
↪job_template ID]
                     [--launch_type {manual,relaunch,callback,scheduled,dependency,
↪workflow,webhook,sync,scm}]
                     [--status {new,pending,waiting,running,successful,failed,error,
↪canceled}] [--execution_environment ID] [--failed BOOLEAN] [--started STARTED]
                     [--finished FINISHED] [--canceled_on CANCELED_ON] [--elapsed␣
↪ELAPSED] [--job_explanation TEXT] [--execution_node TEXT] [--controller_node TEXT]
                     [--work_unit_id TEXT] [--job_type {run,check,scan}] [--inventory␣
↪ID] [--project ID] [--playbook TEXT] [--scm_branch TEXT] [--forks INTEGER]
                     [--limit TEXT] [--verbosity {0,1,2,3,4,5}] [--extra_vars JSON/YAML]␣
↪[--job_tags TEXT] [--force_handlers BOOLEAN] [--skip_tags TEXT]
                     [--start_at_task TEXT] [--timeout INTEGER] [--use_fact_cache␣
↪BOOLEAN] [--organization ID] [--job_template ID] [--allow_simultaneous BOOLEAN]
                     [--artifacts JSON/YAML] [--scm_revision TEXT] [--instance_group ID]␣
↪[--diff_mode BOOLEAN] [--job_slice_number INTEGER]
                     [--job_slice_count INTEGER] [--webhook_service {,gitflic,github,
↪gitlab,bitbucket_dc}] [--webhook_credential ID] [--webhook_guid TEXT]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --type {job}          only list jobs with the specified type
  --created CREATED     only list jobs with the specified created
  --modified MODIFIED   only list jobs with the specified modified
  --name TEXT           only list jobs with the specified name
  --description TEXT    only list jobs with the specified description
  --unified_job_template ID
                        only list jobs with the specified unified_job_template
  --launch_type {manual,relaunch,callback,scheduled,dependency,workflow,webhook,sync,scm}
                        only list jobs with the specified launch_type
  --status {new,pending,waiting,running,successful,failed,error,canceled}
                        only list jobs with the specified status
  --execution_environment ID
                        only list jobs with the specified execution_environment
  --failed BOOLEAN      only list jobs with the specified failed
  --started STARTED     only list jobs with the specified started
  --finished FINISHED   only list jobs with the specified finished
  --canceled_on CANCELED_ON
                        only list jobs with the specified canceled_on
  --elapsed ELAPSED     only list jobs with the specified elapsed
  --job_explanation TEXT
                        only list jobs with the specified job_explanation
  --execution_node TEXT
                        only list jobs with the specified execution_node
  --controller_node TEXT
                        only list jobs with the specified controller_node
  --work_unit_id TEXT   only list jobs with the specified work_unit_id
  --job_type {run,check,scan}
                        only list jobs with the specified job_type
  --inventory ID        only list jobs with the specified inventory
  --project ID          only list jobs with the specified project
  --playbook TEXT       only list jobs with the specified playbook
  --scm_branch TEXT     only list jobs with the specified scm_branch
  --forks INTEGER       only list jobs with the specified forks
  --limit TEXT          only list jobs with the specified limit
  --verbosity {0,1,2,3,4,5}
                        only list jobs with the specified verbosity
  --extra_vars JSON/YAML
                        only list jobs with the specified extra_vars
  --job_tags TEXT       only list jobs with the specified job_tags
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  --force_handlers BOOLEAN
                        only list jobs with the specified force_handlers
  --skip_tags TEXT      only list jobs with the specified skip_tags
  --start_at_task TEXT  only list jobs with the specified start_at_task
  --timeout INTEGER     only list jobs with the specified timeout
  --use_fact_cache BOOLEAN
                        only list jobs with the specified use_fact_cache
  --organization ID     only list jobs with the specified organization
  --job_template ID     only list jobs with the specified job_template
  --allow_simultaneous BOOLEAN
                        only list jobs with the specified allow_simultaneous
  --artifacts JSON/YAML
                        only list jobs with the specified artifacts
  --scm_revision TEXT   only list jobs with the specified scm_revision
  --instance_group ID   only list jobs with the specified instance_group
  --diff_mode BOOLEAN   only list jobs with the specified diff_mode
  --job_slice_number INTEGER
                        only list jobs with the specified job_slice_number
  --job_slice_count INTEGER
                        only list jobs with the specified job_slice_count
  --webhook_service {,gitflic,github,gitlab,bitbucket_dc}
                        only list jobs with the specified webhook_service
  --webhook_credential ID
                        only list jobs with the specified webhook_credential
  --webhook_guid TEXT   only list jobs with the specified webhook_guid

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

jobs get

```
usage: awx jobs get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--conf.color␣
↪BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx jobs get: the following arguments are required: id
```

jobs delete

```
usage: awx jobs delete [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
awx jobs delete: the following arguments are required: id
```

jobs stdout

```
usage: awx jobs stdout [-h] id

positional arguments:
  id

options:
  -h, --help  show this help message and exit

awx jobs stdout: the following arguments are required: id
```

jobs monitor

```
usage: awx jobs monitor [-h] id

positional arguments:
  id

options:
  -h, --help  show this help message and exit

awx jobs monitor: the following arguments are required: id
```

labels

```
usage: awx labels [-h] action ...

positional arguments:
  action
    list
    create
    get
    modify

options:
  -h, --help  show this help message and exit

awx labels: the following arguments are required: action
```

labels list

```
usage: awx labels list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,jq,human}] [--
↪filter TEXT] [--conf.color BOOLEAN] [-v] [--type {label}] [--created CREATED]
                       [--modified MODIFIED] [--name TEXT] [--organization ID]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --type {label}        only list labels with the specified type
  --created CREATED     only list labels with the specified created
  --modified MODIFIED   only list labels with the specified modified
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  --name TEXT           only list labels with the specified name
  --organization ID     only list labels with the specified organization

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

labels create

```
usage: awx labels create [-h] --name TEXT --organization ID

required arguments:
  --name TEXT        Name of this label.
  --organization ID  Organization this label belongs to.

options:
  -h, --help         show this help message and exit

awx labels create: the following arguments are required: --name, --organization
```

labels get

```
usage: awx labels get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--conf.color␣
↪BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx labels get: the following arguments are required: id
```

labels modify

```
usage: awx labels modify [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx labels modify: the following arguments are required: id
```

login

```
usage: awx login [-h] [--description TEXT] [--conf.client_id TEXT] [--conf.client_secret␣
↪TEXT] [--conf.scope {read,write}] [--conf.host https://example.awx.org]
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
                [--conf.token TEXT] [--conf.username TEXT] [--conf.password TEXT] [-k]

options:
  -h, --help            show this help message and exit

OAuth2.0 Options:
  --description TEXT    description of the generated OAuth2.0 token
  --conf.client_id TEXT
  --conf.client_secret TEXT
  --conf.scope {read,write}

authentication:
  --conf.host https://example.awx.org
  --conf.token TEXT     an OAuth2.0 token (get one by using `awx login`)
  --conf.username TEXT
  --conf.password TEXT
  -k, --conf.insecure   Allow insecure server connections when using SSL
```

me

```
usage: awx me [-h]

options:
  -h, --help  show this help message and exit
```

mesh_visualizer

```
usage: awx mesh_visualizer [-h]

options:
  -h, --help  show this help message and exit
```

metrics

```
usage: awx metrics [-h]

options:
  -h, --help  show this help message and exit
```

notification_templates

```
usage: awx notification_templates [-h] action ...

positional arguments:
  action
    list
    create
    get
    modify
    delete

options:
  -h, --help  show this help message and exit

awx notification_templates: the following arguments are required: action
```

notification_templates list

```
usage: awx notification_templates list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,
↪jq,human}] [--filter TEXT] [--conf.color BOOLEAN] [-v]
                                       [--type {notification_template}] [--created␣
↪CREATED] [--modified MODIFIED] [--name TEXT] [--description TEXT]
                                       [--organization ID] [--notification_type {awssns,
↪email,grafana,irc,mattermost,pagerduty,rocketchat,slack,twilio,webhook}]
                                       [--notification_configuration JSON/YAML] [--
↪messages JSON/YAML]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --type {notification_template}
                        only list notification_templates with the specified type
  --created CREATED     only list notification_templates with the specified created
  --modified MODIFIED   only list notification_templates with the specified modified
  --name TEXT           only list notification_templates with the specified name
  --description TEXT    only list notification_templates with the specified description
  --organization ID     only list notification_templates with the specified organization
  --notification_type {awssns,email,grafana,irc,mattermost,pagerduty,rocketchat,slack,
↪twilio,webhook}
                        only list notification_templates with the specified notification_
↪type
  --notification_configuration JSON/YAML
                        only list notification_templates with the specified notification_
↪configuration
  --messages JSON/YAML  only list notification_templates with the specified messages

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

notification_templates create

```
usage: awx notification_templates create [-h] --name TEXT [--description TEXT] --
↪organization ID --notification_type
                                         {awssns,email,grafana,irc,mattermost,pagerduty,
↪rocketchat,slack,twilio,webhook} [--notification_configuration JSON/YAML]
                                         [--messages JSON/YAML]

required arguments:
  --name TEXT           Name of this notification template.
  --organization ID     the ID of the associated organization
  --notification_type {awssns,email,grafana,irc,mattermost,pagerduty,rocketchat,slack,
↪twilio,webhook}

options:
  -h, --help            show this help message and exit
  --description TEXT    Optional description of this notification template.
  --notification_configuration JSON/YAML
                        a JSON or YAML string. You can optionally specify a file path e.
↪g., @path/to/file.yml
  --messages JSON/YAML  Optional custom messages for notification template. You can␣
↪optionally specify a file path e.g., @path/to/file.yml
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
awx notification_templates create: the following arguments are required: --name, --
↪organization, --notification_type
```

notification_templates get

```
usage: awx notification_templates get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--
↪conf.color BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx notification_templates get: the following arguments are required: id
```

notification_templates modify

```
usage: awx notification_templates modify [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx notification_templates modify: the following arguments are required: id
```

notification_templates delete

```
usage: awx notification_templates delete [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx notification_templates delete: the following arguments are required: id
```

notifications

```
usage: awx notifications [-h] action ...

positional arguments:
  action
    list
    get

options:
  -h, --help  show this help message and exit

awx notifications: the following arguments are required: action
```

notifications list

```
usage: awx notifications list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,jq,human}
↪] [--filter TEXT] [--conf.color BOOLEAN] [-v] [--type {notification}]
                              [--created CREATED] [--modified MODIFIED] [--notification_
↪template ID] [--error TEXT] [--status {pending,successful,failed}]
                              [--notifications_sent INTEGER] [--notification_type
↪{awssns,email,grafana,irc,mattermost,pagerduty,rocketchat,slack,twilio,webhook}]
                              [--recipients TEXT] [--subject TEXT] [--body JSON/YAML]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --type {notification}
                        only list notifications with the specified type
  --created CREATED     only list notifications with the specified created
  --modified MODIFIED   only list notifications with the specified modified
  --notification_template ID
                        only list notifications with the specified notification_template
  --error TEXT          only list notifications with the specified error
  --status {pending,successful,failed}
                        only list notifications with the specified status
  --notifications_sent INTEGER
                        only list notifications with the specified notifications_sent
  --notification_type {awssns,email,grafana,irc,mattermost,pagerduty,rocketchat,slack,
↪twilio,webhook}
                        only list notifications with the specified notification_type
  --recipients TEXT     only list notifications with the specified recipients
  --subject TEXT        only list notifications with the specified subject
  --body JSON/YAML      only list notifications with the specified body

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

notifications get

```
usage: awx notifications get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--conf.
↪color BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx notifications get: the following arguments are required: id
```

organizations

```
usage: awx organizations [-h] action ...

positional arguments:
  action
    list
    create
    get
    modify
    delete
    associate
    disassociate

options:
  -h, --help    show this help message and exit

awx organizations: the following arguments are required: action
```

organizations list

```
usage: awx organizations list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,jq,human}
↪] [--filter TEXT] [--conf.color BOOLEAN] [-v] [--type {organization}]
                              [--created CREATED] [--modified MODIFIED] [--name TEXT] [--
↪description TEXT] [--max_hosts INTEGER] [--custom_virtualenv TEXT]
                              [--default_environment ID]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --type {organization}
                        only list organizations with the specified type
  --created CREATED     only list organizations with the specified created
  --modified MODIFIED   only list organizations with the specified modified
  --name TEXT           only list organizations with the specified name
  --description TEXT    only list organizations with the specified description
  --max_hosts INTEGER   only list organizations with the specified max_hosts
  --custom_virtualenv TEXT
                        only list organizations with the specified custom_virtualenv
  --default_environment ID
                        only list organizations with the specified default_environment

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

organizations create

```
usage: awx organizations create [-h] --name TEXT [--description TEXT] [--max_hosts␣
↪INTEGER] [--default_environment ID]

required arguments:
  --name TEXT           Name of this organization.

options:
  -h, --help            show this help message and exit
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  --description TEXT    Optional description of this organization.
  --max_hosts INTEGER   Maximum number of hosts allowed to be managed by this␣
↪organization.
  --default_environment ID
                        The default execution environment for jobs run by this␣
↪organization.

awx organizations create: the following arguments are required: --name
```

organizations get

```
usage: awx organizations get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--conf.
↪color BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx organizations get: the following arguments are required: id
```

organizations modify

```
usage: awx organizations modify [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx organizations modify: the following arguments are required: id
```

organizations delete

```
usage: awx organizations delete [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx organizations delete: the following arguments are required: id
```

organizations associate

```
usage: awx organizations associate [-h] (--start_notification  | --success_notification ␣
↪| --failure_notification  | --approval_notification  | --galaxy_credential )
                                   id
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
positional arguments:
  id

options:
  -h, --help            show this help message and exit
  --start_notification
                        The ID (or name) of the notification_template to associate
  --success_notification
                        The ID (or name) of the notification_template to associate
  --failure_notification
                        The ID (or name) of the notification_template to associate
  --approval_notification
                        The ID (or name) of the notification_template to associate
  --galaxy_credential   The ID (or name) of the credential to associate

awx organizations associate: the following arguments are required: id
```

organizations disassociate

```
usage: awx organizations disassociate [-h]
                                      (--start_notification  | --success_notification  |␣
↪--failure_notification  | --approval_notification  | --galaxy_credential )
                                      id

positional arguments:
  id

options:
  -h, --help            show this help message and exit
  --start_notification
                        The ID (or name) of the notification_template to disassociate
  --success_notification
                        The ID (or name) of the notification_template to disassociate
  --failure_notification
                        The ID (or name) of the notification_template to disassociate
  --approval_notification
                        The ID (or name) of the notification_template to disassociate
  --galaxy_credential   The ID (or name) of the credential to disassociate

awx organizations disassociate: the following arguments are required: id
```

ping

```
usage: awx ping [-h]

options:
  -h, --help  show this help message and exit
```

project_updates

```
usage: awx project_updates [-h] action ...

positional arguments:
  action
    list
    get
    delete
    stdout

options:
  -h, --help  show this help message and exit
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
awx project_updates: the following arguments are required: action
```

project_updates list

```
usage: awx project_updates list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,jq,
↪human}] [--filter TEXT] [--conf.color BOOLEAN] [-v] [--type {project_update}]
                                [--created CREATED] [--modified MODIFIED] [--name TEXT]␣
↪[--description TEXT] [--unified_job_template ID]
                                [--launch_type {manual,relaunch,callback,scheduled,
↪dependency,workflow,webhook,sync,scm}]
                                [--status {new,pending,waiting,running,successful,failed,
↪error,canceled}] [--execution_environment ID] [--failed BOOLEAN]
                                [--started STARTED] [--finished FINISHED] [--canceled_on␣
↪CANCELED_ON] [--elapsed ELAPSED] [--job_explanation TEXT]
                                [--execution_node TEXT] [--work_unit_id TEXT] [--local_
↪path TEXT] [--scm_type {,git,svn,insights,archive}] [--scm_url TEXT]
                                [--scm_branch TEXT] [--scm_refspec TEXT] [--scm_clean␣
↪BOOLEAN] [--scm_track_submodules BOOLEAN] [--scm_delete_on_update BOOLEAN]
                                [--credential ID] [--timeout INTEGER] [--scm_revision␣
↪TEXT] [--project ID] [--job_type {run,check}] [--job_tags TEXT]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --type {project_update}
                        only list project_updates with the specified type
  --created CREATED     only list project_updates with the specified created
  --modified MODIFIED   only list project_updates with the specified modified
  --name TEXT           only list project_updates with the specified name
  --description TEXT    only list project_updates with the specified description
  --unified_job_template ID
                        only list project_updates with the specified unified_job_template
  --launch_type {manual,relaunch,callback,scheduled,dependency,workflow,webhook,sync,scm}
                        only list project_updates with the specified launch_type
  --status {new,pending,waiting,running,successful,failed,error,canceled}
                        only list project_updates with the specified status
  --execution_environment ID
                        only list project_updates with the specified execution_
↪environment
  --failed BOOLEAN      only list project_updates with the specified failed
  --started STARTED     only list project_updates with the specified started
  --finished FINISHED   only list project_updates with the specified finished
  --canceled_on CANCELED_ON
                        only list project_updates with the specified canceled_on
  --elapsed ELAPSED     only list project_updates with the specified elapsed
  --job_explanation TEXT
                        only list project_updates with the specified job_explanation
  --execution_node TEXT
                        only list project_updates with the specified execution_node
  --work_unit_id TEXT   only list project_updates with the specified work_unit_id
  --local_path TEXT     only list project_updates with the specified local_path
  --scm_type {,git,svn,insights,archive}
                        only list project_updates with the specified scm_type
  --scm_url TEXT        only list project_updates with the specified scm_url
  --scm_branch TEXT     only list project_updates with the specified scm_branch
  --scm_refspec TEXT    only list project_updates with the specified scm_refspec
  --scm_clean BOOLEAN   only list project_updates with the specified scm_clean
  --scm_track_submodules BOOLEAN
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
                        only list project_updates with the specified scm_track_submodules
  --scm_delete_on_update BOOLEAN
                        only list project_updates with the specified scm_delete_on_update
  --credential ID       only list project_updates with the specified credential
  --timeout INTEGER     only list project_updates with the specified timeout
  --scm_revision TEXT   only list project_updates with the specified scm_revision
  --project ID          only list project_updates with the specified project
  --job_type {run,check}
                        only list project_updates with the specified job_type
  --job_tags TEXT       only list project_updates with the specified job_tags

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

project_updates get

```
usage: awx project_updates get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--conf.
↪color BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx project_updates get: the following arguments are required: id
```

project_updates delete

```
usage: awx project_updates delete [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx project_updates delete: the following arguments are required: id
```

project_updates stdout

```
usage: awx project_updates stdout [-h] id

positional arguments:
  id

options:
  -h, --help  show this help message and exit

awx project_updates stdout: the following arguments are required: id
```

projects

```
usage: awx projects [-h] action ...

positional arguments:
  action
    list
    create
    get
    modify
    delete
    update
    associate
    disassociate

options:
  -h, --help    show this help message and exit

awx projects: the following arguments are required: action
```

projects list

```
usage: awx projects list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,jq,human}] [--
↪filter TEXT] [--conf.color BOOLEAN] [-v] [--type {project}]
                         [--created CREATED] [--modified MODIFIED] [--name TEXT] [--
↪description TEXT] [--local_path TEXT] [--scm_type {,git,svn,insights,archive}]
                         [--scm_url TEXT] [--scm_branch TEXT] [--scm_refspec TEXT] [--
↪scm_clean BOOLEAN] [--scm_track_submodules BOOLEAN]
                         [--scm_delete_on_update BOOLEAN] [--credential ID] [--timeout␣
↪INTEGER] [--scm_revision TEXT] [--last_job_run LAST_JOB_RUN]
                         [--last_job_failed BOOLEAN] [--next_job_run NEXT_JOB_RUN]
                         [--status {new,pending,waiting,running,successful,failed,error,
↪canceled,never updated,ok,missing}] [--organization ID]
                         [--scm_update_on_launch BOOLEAN] [--scm_update_cache_timeout␣
↪INTEGER] [--allow_override BOOLEAN] [--custom_virtualenv TEXT]
                         [--default_environment ID] [--signature_validation_credential␣
↪ID]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --type {project}      only list projects with the specified type
  --created CREATED     only list projects with the specified created
  --modified MODIFIED   only list projects with the specified modified
  --name TEXT           only list projects with the specified name
  --description TEXT    only list projects with the specified description
  --local_path TEXT     only list projects with the specified local_path
  --scm_type {,git,svn,insights,archive}
                        only list projects with the specified scm_type
  --scm_url TEXT        only list projects with the specified scm_url
  --scm_branch TEXT     only list projects with the specified scm_branch
  --scm_refspec TEXT    only list projects with the specified scm_refspec
  --scm_clean BOOLEAN   only list projects with the specified scm_clean
  --scm_track_submodules BOOLEAN
                        only list projects with the specified scm_track_submodules
  --scm_delete_on_update BOOLEAN
                        only list projects with the specified scm_delete_on_update
  --credential ID       only list projects with the specified credential
  --timeout INTEGER     only list projects with the specified timeout
  --scm_revision TEXT   only list projects with the specified scm_revision
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  --last_job_run LAST_JOB_RUN
                        only list projects with the specified last_job_run
  --last_job_failed BOOLEAN
                        only list projects with the specified last_job_failed
  --next_job_run NEXT_JOB_RUN
                        only list projects with the specified next_job_run
  --status {new,pending,waiting,running,successful,failed,error,canceled,never updated,
↪ok,missing}
                        only list projects with the specified status
  --organization ID     only list projects with the specified organization
  --scm_update_on_launch BOOLEAN
                        only list projects with the specified scm_update_on_launch
  --scm_update_cache_timeout INTEGER
                        only list projects with the specified scm_update_cache_timeout
  --allow_override BOOLEAN
                        only list projects with the specified allow_override
  --custom_virtualenv TEXT
                        only list projects with the specified custom_virtualenv
  --default_environment ID
                        only list projects with the specified default_environment
  --signature_validation_credential ID
                        only list projects with the specified signature_validation_
↪credential

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

projects create

```
usage: awx projects create [-h] [--monitor] [--wait] --name TEXT [--description TEXT] [--
↪local_path TEXT] [--scm_type {,git,svn,insights,archive}] [--scm_url TEXT]
                           [--scm_branch TEXT] [--scm_refspec TEXT] [--scm_clean␣
↪BOOLEAN] [--scm_track_submodules BOOLEAN] [--scm_delete_on_update BOOLEAN]
                           [--credential ID] [--timeout INTEGER] [--organization ID] [--
↪scm_update_on_launch BOOLEAN] [--scm_update_cache_timeout INTEGER]
                           [--allow_override BOOLEAN] [--default_environment ID] [--
↪signature_validation_credential ID]

required arguments:
  --name TEXT           Name of this project.

options:
  -h, --help            show this help message and exit
  --monitor             If set, prints stdout of the project update until it finishes.
  --wait                If set, waits until the new project has updated.
  --description TEXT    Optional description of this project.
  --local_path TEXT     Local path (relative to PROJECTS_ROOT) containing playbooks and␣
↪related files for this project.
  --scm_type {,git,svn,insights,archive}
                        Specifies the source control system used to store the project.
  --scm_url TEXT        The location where the project is stored.
  --scm_branch TEXT     Specific branch, tag or commit to checkout.
  --scm_refspec TEXT    For git projects, an additional refspec to fetch.
  --scm_clean BOOLEAN   Discard any local changes before syncing the project.
  --scm_track_submodules BOOLEAN
                        Track submodules latest commits on defined branch.
  --scm_delete_on_update BOOLEAN
                        Delete the project before syncing.
  --credential ID       the ID of the associated credential
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  --timeout INTEGER     The amount of time (in seconds) to run before the task is␣
↪canceled.
  --organization ID     The organization used to determine access to this template.
  --scm_update_on_launch BOOLEAN
                        Update the project when a job is launched that uses the project.
  --scm_update_cache_timeout INTEGER
                        The number of seconds after the last project update ran that a␣
↪new project update will be launched as a job dependency.
  --allow_override BOOLEAN
                        Allow changing the SCM branch or revision in a job template that␣
↪uses this project.
  --default_environment ID
                        The default execution environment for jobs run using this␣
↪project.
  --signature_validation_credential ID
                        An optional credential used for validating files in the project␣
↪against unexpected changes.

awx projects create: the following arguments are required: --name
```

projects get

```
usage: awx projects get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--conf.color␣
↪BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx projects get: the following arguments are required: id
```

projects modify

```
usage: awx projects modify [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx projects modify: the following arguments are required: id
```

projects delete

```
usage: awx projects delete [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
awx projects delete: the following arguments are required: id
```

projects update

```
usage: awx projects update [-h] [--monitor] [--action-timeout ACTION_TIMEOUT] [--wait] [-
↪-interval INTERVAL] id

positional arguments:
  id

options:
  -h, --help            show this help message and exit
  --monitor             If set, prints stdout of the launched job until it finishes.
  --action-timeout ACTION_TIMEOUT
                        If set with --monitor or --wait, time out waiting on job␣
↪completion.
  --wait                If set, waits until the launched job finishes.
  --interval INTERVAL   If set with --monitor or --wait, amount of time to wait in␣
↪seconds between api calls. Minimum value is 2.5 seconds to avoid overwhelming the
                        api

awx projects update: the following arguments are required: id
```

projects associate

```
usage: awx projects associate [-h] (--start_notification  | --success_notification  | --
↪failure_notification ) id

positional arguments:
  id

options:
  -h, --help            show this help message and exit
  --start_notification
                        The ID (or name) of the notification_template to associate
  --success_notification
                        The ID (or name) of the notification_template to associate
  --failure_notification
                        The ID (or name) of the notification_template to associate

awx projects associate: the following arguments are required: id
```

projects disassociate

```
usage: awx projects disassociate [-h] (--start_notification  | --success_notification  |␣
↪--failure_notification ) id

positional arguments:
  id

options:
  -h, --help            show this help message and exit
  --start_notification
                        The ID (or name) of the notification_template to disassociate
  --success_notification
                        The ID (or name) of the notification_template to disassociate
  --failure_notification
                        The ID (or name) of the notification_template to disassociate

awx projects disassociate: the following arguments are required: id
```

receptor_addresses

```
usage: awx receptor_addresses [-h] action ...

positional arguments:
  action
    list
    get

options:
  -h, --help  show this help message and exit

awx receptor_addresses: the following arguments are required: action
```

receptor_addresses list

```
usage: awx receptor_addresses list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,jq,
↪human}] [--filter TEXT] [--conf.color BOOLEAN] [-v] [--address TEXT]
                                   [--port INTEGER] [--protocol {tcp,ws,wss}] [--
↪websocket_path TEXT] [--is_internal BOOLEAN] [--canonical BOOLEAN] [--instance ID]
                                   [--peers_from_control_nodes BOOLEAN]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --address TEXT        only list receptor_addresses with the specified address
  --port INTEGER        only list receptor_addresses with the specified port
  --protocol {tcp,ws,wss}
                        only list receptor_addresses with the specified protocol
  --websocket_path TEXT
                        only list receptor_addresses with the specified websocket_path
  --is_internal BOOLEAN
                        only list receptor_addresses with the specified is_internal
  --canonical BOOLEAN   only list receptor_addresses with the specified canonical
  --instance ID         only list receptor_addresses with the specified instance
  --peers_from_control_nodes BOOLEAN
                        only list receptor_addresses with the specified peers_from_
↪control_nodes

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

receptor_addresses get

```
usage: awx receptor_addresses get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--conf.
↪color BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx receptor_addresses get: the following arguments are required: id
```

role_definitions

```
usage: awx role_definitions [-h] action ...

positional arguments:
  action
    list
    create
    get
    modify
    delete

options:
  -h, --help  show this help message and exit

awx role_definitions: the following arguments are required: action
```

role_definitions list

```
usage: awx role_definitions list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,jq,
↪human}] [--filter TEXT] [--conf.color BOOLEAN] [-v]
                                 [--content_type {None,awx.credential,awx.
↪executionenvironment,awx.instancegroup,awx.inventory,awx.jobtemplate,awx.
↪notificationtemplate,awx.project,awx.workflowjobtemplate,shared.organization,shared.
↪team}]
                                 [--modified MODIFIED] [--created CREATED] [--name TEXT]␣
↪[--description TEXT] [--managed BOOLEAN] [--modified_by ID] [--created_by ID]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --content_type {None,awx.credential,awx.executionenvironment,awx.instancegroup,awx.
↪inventory,awx.jobtemplate,awx.notificationtemplate,awx.project,awx.workflowjobtemplate,
↪shared.organization,shared.team}
                        only list role_definitions with the specified content_type
  --modified MODIFIED   only list role_definitions with the specified modified
  --created CREATED     only list role_definitions with the specified created
  --name TEXT           only list role_definitions with the specified name
  --description TEXT    only list role_definitions with the specified description
  --managed BOOLEAN     only list role_definitions with the specified managed
  --modified_by ID      only list role_definitions with the specified modified_by
  --created_by ID       only list role_definitions with the specified created_by

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

role_definitions create

```
usage: awx role_definitions create [-h] --permissions PERMISSIONS
                                   [--content_type {None,awx.credential,awx.
↪executionenvironment,awx.instancegroup,awx.inventory,awx.jobtemplate,awx.
↪notificationtemplate,awx.project,awx.workflowjobtemplate,shared.organization,shared.
↪team}]
                                   --name TEXT [--description TEXT]

required arguments:
  --permissions PERMISSIONS
  --name TEXT           Name of this role definition.

options:
  -h, --help            show this help message and exit
  --content_type {None,awx.credential,awx.executionenvironment,awx.instancegroup,awx.
↪inventory,awx.jobtemplate,awx.notificationtemplate,awx.project,awx.workflowjobtemplate,
↪shared.organization,shared.team}
                        The type of resource this applies to
  --description TEXT    Optional description of this role definition.

awx role_definitions create: the following arguments are required: --permissions, --name
```

role_definitions get

```
usage: awx role_definitions get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--conf.
↪color BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx role_definitions get: the following arguments are required: id
```

role_definitions modify

```
usage: awx role_definitions modify [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx role_definitions modify: the following arguments are required: id
```

role_definitions delete

```
usage: awx role_definitions delete [-h] id

positional arguments:
  id          the ID (or unique name) of the resource
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
options:
  -h, --help  show this help message and exit

awx role_definitions delete: the following arguments are required: id
```

role_team_assignments

```
usage: awx role_team_assignments [-h] action ...

positional arguments:
  action
    list
    create
    get
    delete

options:
  -h, --help  show this help message and exit

awx role_team_assignments: the following arguments are required: action
```

role_team_assignments list

```
usage: awx role_team_assignments list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,
↪jq,human}] [--filter TEXT] [--conf.color BOOLEAN] [-v] [--created CREATED]
                                      [--created_by ID]
                                      [--content_type {awx.credential,awx.
↪executionenvironment,awx.instancegroup,awx.inventory,awx.jobtemplate,awx.
↪notificationtemplate,awx.project,awx.workflowjobtemplate,shared.organization,shared.
↪team}]
                                      [--object_id TEXT] [--role_definition ID] [--team␣
↪ID]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --created CREATED     only list role_team_assignments with the specified created
  --created_by ID       only list role_team_assignments with the specified created_by
  --content_type {awx.credential,awx.executionenvironment,awx.instancegroup,awx.
↪inventory,awx.jobtemplate,awx.notificationtemplate,awx.project,awx.workflowjobtemplate,
↪shared.organization,shared.team}
                        only list role_team_assignments with the specified content_type
  --object_id TEXT      only list role_team_assignments with the specified object_id
  --role_definition ID  only list role_team_assignments with the specified role_
↪definition
  --team ID             only list role_team_assignments with the specified team

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

role_team_assignments create

```
usage: awx role_team_assignments create [-h] [--object_id TEXT] [--object_ansible_id␣
↪TEXT] --role_definition ID [--team ID] [--team_ansible_id TEXT]

required arguments:
  --role_definition ID  The role definition which defines permissions conveyed by this␣
↪assignment

options:
  -h, --help            show this help message and exit
  --object_id TEXT      Primary key of the object this assignment applies to, null value␣
↪indicates system-wide assignment
  --object_ansible_id TEXT
                        Resource id of the object this role applies to. Alternative to␣
↪the object_id field.
  --team ID             the ID of the associated team
  --team_ansible_id TEXT
                        Resource id of the team who will receive permissions from this␣
↪assignment. Alternative to team field.

awx role_team_assignments create: the following arguments are required: --role_definition
```

role_team_assignments get

```
usage: awx role_team_assignments get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--
↪conf.color BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx role_team_assignments get: the following arguments are required: id
```

role_team_assignments delete

```
usage: awx role_team_assignments delete [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx role_team_assignments delete: the following arguments are required: id
```

role_user_assignments

```
usage: awx role_user_assignments [-h] action ...

positional arguments:
  action
    list
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
    create
    get
    delete

options:
  -h, --help  show this help message and exit

awx role_user_assignments: the following arguments are required: action
```

role_user_assignments list

```
usage: awx role_user_assignments list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,
↪jq,human}] [--filter TEXT] [--conf.color BOOLEAN] [-v] [--created CREATED]
                                      [--created_by ID]
                                      [--content_type {awx.credential,awx.
↪executionenvironment,awx.instancegroup,awx.inventory,awx.jobtemplate,awx.
↪notificationtemplate,awx.project,awx.workflowjobtemplate,shared.organization,shared.
↪team}]
                                      [--object_id TEXT] [--role_definition ID] [--user␣
↪ID]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --created CREATED     only list role_user_assignments with the specified created
  --created_by ID       only list role_user_assignments with the specified created_by
  --content_type {awx.credential,awx.executionenvironment,awx.instancegroup,awx.
↪inventory,awx.jobtemplate,awx.notificationtemplate,awx.project,awx.workflowjobtemplate,
↪shared.organization,shared.team}
                        only list role_user_assignments with the specified content_type
  --object_id TEXT      only list role_user_assignments with the specified object_id
  --role_definition ID  only list role_user_assignments with the specified role_
↪definition
  --user ID             only list role_user_assignments with the specified user

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

role_user_assignments create

```
usage: awx role_user_assignments create [-h] [--object_id TEXT] [--object_ansible_id␣
↪TEXT] --role_definition ID [--user ID] [--user_ansible_id TEXT]

required arguments:
  --role_definition ID  The role definition which defines permissions conveyed by this␣
↪assignment

options:
  -h, --help            show this help message and exit
  --object_id TEXT      Primary key of the object this assignment applies to, null value␣
↪indicates system-wide assignment
  --object_ansible_id TEXT
                        Resource id of the object this role applies to. Alternative to␣
↪the object_id field.
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  --user ID             the ID of the associated user
  --user_ansible_id TEXT
                        Resource id of the user who will receive permissions from this␣
↪assignment. Alternative to user field.

awx role_user_assignments create: the following arguments are required: --role_definition
```

role_user_assignments get

```
usage: awx role_user_assignments get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--
↪conf.color BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx role_user_assignments get: the following arguments are required: id
```

role_user_assignments delete

```
usage: awx role_user_assignments delete [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx role_user_assignments delete: the following arguments are required: id
```

roles

```
usage: awx roles [-h] action ...

This resource has been deprecated and will be removed in a future release.

positional arguments:
  action
    list
    get

options:
  -h, --help  show this help message and exit

awx roles: the following arguments are required: action
```

roles list

```
usage: awx roles list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,jq,human}] [--
↪filter TEXT] [--conf.color BOOLEAN] [-v] [--type {role}]
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --type {role}         only list roles with the specified type

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

roles get

```
usage: awx roles get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--conf.color␣
↪BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx roles get: the following arguments are required: id
```

schedules

```
usage: awx schedules [-h] action ...

positional arguments:
  action
    list
    create
    get
    modify
    delete

options:
  -h, --help  show this help message and exit

awx schedules: the following arguments are required: action
```

schedules list

```
usage: awx schedules list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,jq,human}] [-
↪-filter TEXT] [--conf.color BOOLEAN] [-v] [--rrule TEXT] [--type {schedule}]
                          [--created CREATED] [--modified MODIFIED] [--name TEXT] [--
↪description TEXT] [--extra_data JSON/YAML] [--inventory ID]
                          [--execution_environment ID] [--unified_job_template ID] [--
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
↪enabled BOOLEAN] [--dtstart DTSTART] [--dtend DTEND] [--next_run NEXT_RUN]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --rrule TEXT          only list schedules with the specified rrule
  --type {schedule}     only list schedules with the specified type
  --created CREATED     only list schedules with the specified created
  --modified MODIFIED   only list schedules with the specified modified
  --name TEXT           only list schedules with the specified name
  --description TEXT    only list schedules with the specified description
  --extra_data JSON/YAML
                        only list schedules with the specified extra_data
  --inventory ID        only list schedules with the specified inventory
  --execution_environment ID
                        only list schedules with the specified execution_environment
  --unified_job_template ID
                        only list schedules with the specified unified_job_template
  --enabled BOOLEAN     only list schedules with the specified enabled
  --dtstart DTSTART     only list schedules with the specified dtstart
  --dtend DTEND         only list schedules with the specified dtend
  --next_run NEXT_RUN   only list schedules with the specified next_run

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

schedules create

```
usage: awx schedules create [-h] --rrule TEXT --name TEXT [--description TEXT] [--extra_
↪data JSON/YAML] [--inventory ID] [--scm_branch TEXT]
                            [--job_type {None,,run,check}] [--job_tags TEXT] [--skip_
↪tags TEXT] [--limit TEXT] [--diff_mode BOOLEAN] [--verbosity {None,0,1,2,3,4,5}]
                            [--execution_environment ID] [--forks INTEGER] [--job_slice_
↪count INTEGER] [--timeout INTEGER] --unified_job_template ID
                            [--enabled BOOLEAN]

required arguments:
  --rrule TEXT          A value representing the schedules iCal recurrence rule.
  --name TEXT           Name of this schedule.
  --unified_job_template ID
                        the ID of the associated unified_job_template

options:
  -h, --help            show this help message and exit
  --description TEXT    Optional description of this schedule.
  --extra_data JSON/YAML
                        a JSON or YAML string. You can optionally specify a file path e.
↪g., @path/to/file.yml
  --inventory ID        Inventory applied as a prompt, assuming job template prompts for␣
↪inventory
  --scm_branch TEXT
  --job_type {None,,run,check}
  --job_tags TEXT
  --skip_tags TEXT
  --limit TEXT
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  --diff_mode BOOLEAN
  --verbosity {None,0,1,2,3,4,5}
  --execution_environment ID
                        The container image to be used for execution.
  --forks INTEGER
  --job_slice_count INTEGER
  --timeout INTEGER
  --enabled BOOLEAN     Enables processing of this schedule.

awx schedules create: the following arguments are required: --rrule, --name, --unified_
↪job_template
```

schedules get

```
usage: awx schedules get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--conf.color␣
↪BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx schedules get: the following arguments are required: id
```

schedules modify

```
usage: awx schedules modify [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx schedules modify: the following arguments are required: id
```

schedules delete

```
usage: awx schedules delete [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx schedules delete: the following arguments are required: id
```

service_index

```
usage: awx service_index [-h] action ...

positional arguments:
  action
    list
    get

options:
  -h, --help  show this help message and exit

awx service_index: the following arguments are required: action
```

service_index list

```
usage: awx service_index list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,jq,human}
↪] [--filter TEXT] [--conf.color BOOLEAN] [-v]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

service_index get

```
usage: awx service_index get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--conf.
↪color BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx service_index get: the following arguments are required: id
```

settings

```
usage: awx settings [-h] action ...

positional arguments:
  action
    list
    modify
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
options:
  -h, --help  show this help message and exit

awx settings: the following arguments are required: action
```

settings list

```
usage: awx settings list [-h] [--slug SLUG]

options:
  -h, --help   show this help message and exit
  --slug SLUG  optional setting category/slug
```

settings modify

```
usage: awx settings modify [-h] key value

positional arguments:
  key
  value

options:
  -h, --help  show this help message and exit

awx settings modify: the following arguments are required: key, value
```

system_job_templates

```
usage: awx system_job_templates [-h] action ...

positional arguments:
  action
    list
    get

options:
  -h, --help  show this help message and exit

awx system_job_templates: the following arguments are required: action
```

system_job_templates list

```
usage: awx system_job_templates list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,
↪jq,human}] [--filter TEXT] [--conf.color BOOLEAN] [-v]
                                     [--type {system_job_template}] [--created CREATED]␣
↪[--modified MODIFIED] [--name TEXT] [--description TEXT]
                                     [--last_job_run LAST_JOB_RUN] [--last_job_failed␣
↪BOOLEAN] [--next_job_run NEXT_JOB_RUN]
                                     [--status {new,pending,waiting,running,successful,
↪failed,error,canceled,never updated,ok,missing,none,updating}]
                                     [--execution_environment ID] [--job_type {,cleanup_
↪jobs,cleanup_activitystream,cleanup_sessions,cleanup_tokens}]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --type {system_job_template}
                        only list system_job_templates with the specified type
  --created CREATED     only list system_job_templates with the specified created
  --modified MODIFIED   only list system_job_templates with the specified modified
  --name TEXT           only list system_job_templates with the specified name
  --description TEXT    only list system_job_templates with the specified description
  --last_job_run LAST_JOB_RUN
                        only list system_job_templates with the specified last_job_run
  --last_job_failed BOOLEAN
                        only list system_job_templates with the specified last_job_failed
  --next_job_run NEXT_JOB_RUN
                        only list system_job_templates with the specified next_job_run
  --status {new,pending,waiting,running,successful,failed,error,canceled,never updated,
↪ok,missing,none,updating}
                        only list system_job_templates with the specified status
  --execution_environment ID
                        only list system_job_templates with the specified execution_
↪environment
  --job_type {,cleanup_jobs,cleanup_activitystream,cleanup_sessions,cleanup_tokens}
                        only list system_job_templates with the specified job_type

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

system_job_templates get

```
usage: awx system_job_templates get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--
↪conf.color BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx system_job_templates get: the following arguments are required: id
```

system_jobs

```
usage: awx system_jobs [-h] action ...

positional arguments:
  action
    list
    get
    delete

options:
  -h, --help  show this help message and exit
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
awx system_jobs: the following arguments are required: action
```

system_jobs list

```
usage: awx system_jobs list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,jq,human}]␣
↪[--filter TEXT] [--conf.color BOOLEAN] [-v] [--type {system_job}]
                            [--created CREATED] [--modified MODIFIED] [--name TEXT] [--
↪description TEXT] [--unified_job_template ID]
                            [--launch_type {manual,relaunch,callback,scheduled,
↪dependency,workflow,webhook,sync,scm}]
                            [--status {new,pending,waiting,running,successful,failed,
↪error,canceled}] [--execution_environment ID] [--failed BOOLEAN]
                            [--started STARTED] [--finished FINISHED] [--canceled_on␣
↪CANCELED_ON] [--elapsed ELAPSED] [--job_explanation TEXT] [--execution_node TEXT]
                            [--work_unit_id TEXT] [--system_job_template ID] [--job_type
↪{,cleanup_jobs,cleanup_activitystream,cleanup_sessions,cleanup_tokens}]
                            [--extra_vars TEXT]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --type {system_job}   only list system_jobs with the specified type
  --created CREATED     only list system_jobs with the specified created
  --modified MODIFIED   only list system_jobs with the specified modified
  --name TEXT           only list system_jobs with the specified name
  --description TEXT    only list system_jobs with the specified description
  --unified_job_template ID
                        only list system_jobs with the specified unified_job_template
  --launch_type {manual,relaunch,callback,scheduled,dependency,workflow,webhook,sync,scm}
                        only list system_jobs with the specified launch_type
  --status {new,pending,waiting,running,successful,failed,error,canceled}
                        only list system_jobs with the specified status
  --execution_environment ID
                        only list system_jobs with the specified execution_environment
  --failed BOOLEAN      only list system_jobs with the specified failed
  --started STARTED     only list system_jobs with the specified started
  --finished FINISHED   only list system_jobs with the specified finished
  --canceled_on CANCELED_ON
                        only list system_jobs with the specified canceled_on
  --elapsed ELAPSED     only list system_jobs with the specified elapsed
  --job_explanation TEXT
                        only list system_jobs with the specified job_explanation
  --execution_node TEXT
                        only list system_jobs with the specified execution_node
  --work_unit_id TEXT   only list system_jobs with the specified work_unit_id
  --system_job_template ID
                        only list system_jobs with the specified system_job_template
  --job_type {,cleanup_jobs,cleanup_activitystream,cleanup_sessions,cleanup_tokens}
                        only list system_jobs with the specified job_type
  --extra_vars TEXT     only list system_jobs with the specified extra_vars

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

system_jobs get

```
usage: awx system_jobs get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--conf.color␣
↪BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx system_jobs get: the following arguments are required: id
```

system_jobs delete

```
usage: awx system_jobs delete [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx system_jobs delete: the following arguments are required: id
```

teams

```
usage: awx teams [-h] action ...

positional arguments:
  action
    list
    create
    get
    modify
    delete
    grant
    revoke

options:
  -h, --help  show this help message and exit

awx teams: the following arguments are required: action
```

teams list

```
usage: awx teams list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,jq,human}] [--
↪filter TEXT] [--conf.color BOOLEAN] [-v] [--type {team}] [--created CREATED]
                      [--modified MODIFIED] [--name TEXT] [--description TEXT] [--
↪organization ID]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --type {team}         only list teams with the specified type
  --created CREATED     only list teams with the specified created
  --modified MODIFIED   only list teams with the specified modified
  --name TEXT           only list teams with the specified name
  --description TEXT    only list teams with the specified description
  --organization ID     only list teams with the specified organization

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

teams create

```
usage: awx teams create [-h] --name TEXT [--description TEXT] --organization ID

required arguments:
  --name TEXT         Name of this team.
  --organization ID   the ID of the associated organization

options:
  -h, --help          show this help message and exit
  --description TEXT  Optional description of this team.

awx teams create: the following arguments are required: --name, --organization
```

teams get

```
usage: awx teams get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--conf.color␣
↪BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx teams get: the following arguments are required: id
```

teams modify

```
usage: awx teams modify [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
awx teams modify: the following arguments are required: id
```

teams delete

```
usage: awx teams delete [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx teams delete: the following arguments are required: id
```

teams grant

```
usage: awx teams grant [-h]
                       (--organization ID | --project ID | --inventory ID | --team ID | -
↪-credential ID | --job_template ID | --workflow_job_template ID | --instance_group ID)
                       --role
                       {project_admin,admin,member,notification_admin,execute,update,
↪credential_admin,workflow_admin,approval,adhoc,read,job_template_admin,execution_
↪environment_admin,use,auditor,inventory_admin}
                       id

positional arguments:
  id                    The ID (or name) of the teams to grant access to/from

options:
  -h, --help            show this help message and exit
  --organization ID     The ID (or name) of the target organization
  --project ID          The ID (or name) of the target project
  --inventory ID        The ID (or name) of the target inventory
  --team ID             The ID (or name) of the target team
  --credential ID       The ID (or name) of the target credential
  --job_template ID     The ID (or name) of the target job_template
  --workflow_job_template ID
                        The ID (or name) of the target workflow_job_template
  --instance_group ID   The ID (or name) of the target instance_group
  --role {project_admin,admin,member,notification_admin,execute,update,credential_admin,
↪workflow_admin,approval,adhoc,read,job_template_admin,execution_environment_admin,use,
↪auditor,inventory_admin}
                        The name of the role to grant

awx teams grant: the following arguments are required: id, --role
```

teams revoke

```
usage: awx teams revoke [-h]
                        (--organization ID | --project ID | --inventory ID | --team ID |␣
↪--credential ID | --job_template ID | --workflow_job_template ID | --instance_group ID)
                        --role
                        {execute,read,credential_admin,job_template_admin,member,
↪approval,adhoc,project_admin,workflow_admin,inventory_admin,execution_environment_
↪admin,admin,use,auditor,update,notification_admin}
                        id

positional arguments:
  id                    The ID (or name) of the teams to revoke access to/from
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
options:
  -h, --help            show this help message and exit
  --organization ID     The ID (or name) of the target organization
  --project ID          The ID (or name) of the target project
  --inventory ID        The ID (or name) of the target inventory
  --team ID             The ID (or name) of the target team
  --credential ID       The ID (or name) of the target credential
  --job_template ID     The ID (or name) of the target job_template
  --workflow_job_template ID
                        The ID (or name) of the target workflow_job_template
  --instance_group ID   The ID (or name) of the target instance_group
  --role {execute,read,credential_admin,job_template_admin,member,approval,adhoc,project_
↪admin,workflow_admin,inventory_admin,execution_environment_admin,admin,use,auditor,
↪update,notification_admin}
                        The name of the role to revoke

awx teams revoke: the following arguments are required: id, --role
```

tokens

```
usage: awx tokens [-h] action ...

positional arguments:
  action
    list
    create
    get
    modify
    delete

options:
  -h, --help  show this help message and exit

awx tokens: the following arguments are required: action
```

tokens list

```
usage: awx tokens list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,jq,human}] [--
↪filter TEXT] [--conf.color BOOLEAN] [-v] [--type {o_auth2_access_token}]
                       [--created CREATED] [--modified MODIFIED] [--description TEXT] [--
↪user ID] [--token TEXT] [--application ID] [--expires EXPIRES] [--scope TEXT]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --type {o_auth2_access_token}
                        only list tokens with the specified type
  --created CREATED     only list tokens with the specified created
  --modified MODIFIED   only list tokens with the specified modified
  --description TEXT    only list tokens with the specified description
  --user ID             only list tokens with the specified user
  --token TEXT          only list tokens with the specified token
  --application ID      only list tokens with the specified application
  --expires EXPIRES     only list tokens with the specified expires
  --scope TEXT          only list tokens with the specified scope

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

tokens create

```
usage: awx tokens create [-h] [--description TEXT] [--application ID] [--scope TEXT]

options:
  -h, --help          show this help message and exit
  --description TEXT  Optional description of this access token.
  --application ID    the ID of the associated application
  --scope TEXT        Allowed scopes, further restricts user's permissions. Must be a␣
↪simple space-separated string with allowed scopes ['read', 'write'].
```

tokens get

```
usage: awx tokens get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--conf.color␣
↪BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx tokens get: the following arguments are required: id
```

tokens modify

```
usage: awx tokens modify [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx tokens modify: the following arguments are required: id
```

tokens delete

```
usage: awx tokens delete [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx tokens delete: the following arguments are required: id
```

unified_job_templates

```
usage: awx unified_job_templates [-h] action ...

positional arguments:
  action
    list
    get

options:
  -h, --help  show this help message and exit

awx unified_job_templates: the following arguments are required: action
```

unified_job_templates list

```
usage: awx unified_job_templates list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,
↪jq,human}] [--filter TEXT] [--conf.color BOOLEAN] [-v]
                                      [--type {project,inventory_source,job_template,
↪system_job_template,workflow_job_template}] [--created CREATED]
                                      [--modified MODIFIED] [--name TEXT] [--description␣
↪TEXT] [--last_job_run LAST_JOB_RUN] [--last_job_failed BOOLEAN]
                                      [--next_job_run NEXT_JOB_RUN]
                                      [--status {new,pending,waiting,running,successful,
↪failed,error,canceled,never updated,ok,missing,none,updating}]
                                      [--execution_environment ID]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --type {project,inventory_source,job_template,system_job_template,workflow_job_
↪template}
                        only list unified_job_templates with the specified type
  --created CREATED     only list unified_job_templates with the specified created
  --modified MODIFIED   only list unified_job_templates with the specified modified
  --name TEXT           only list unified_job_templates with the specified name
  --description TEXT    only list unified_job_templates with the specified description
  --last_job_run LAST_JOB_RUN
                        only list unified_job_templates with the specified last_job_run
  --last_job_failed BOOLEAN
                        only list unified_job_templates with the specified last_job_
↪failed
  --next_job_run NEXT_JOB_RUN
                        only list unified_job_templates with the specified next_job_run
  --status {new,pending,waiting,running,successful,failed,error,canceled,never updated,
↪ok,missing,none,updating}
                        only list unified_job_templates with the specified status
  --execution_environment ID
                        only list unified_job_templates with the specified execution_
↪environment

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

unified_job_templates get

```
usage: awx unified_job_templates get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--
↪conf.color BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx unified_job_templates get: the following arguments are required: id
```

unified_jobs

```
usage: awx unified_jobs [-h] action ...

positional arguments:
  action
    list
    get

options:
  -h, --help  show this help message and exit

awx unified_jobs: the following arguments are required: action
```

unified_jobs list

```
usage: awx unified_jobs list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,jq,human}
↪] [--filter TEXT] [--conf.color BOOLEAN] [-v]
                             [--type {project_update,inventory_update,job,ad_hoc_command,
↪system_job,workflow_job}] [--created CREATED] [--modified MODIFIED]
                             [--name TEXT] [--description TEXT] [--unified_job_template␣
↪ID]
                             [--launch_type {manual,relaunch,callback,scheduled,
↪dependency,workflow,webhook,sync,scm}]
                             [--status {new,pending,waiting,running,successful,failed,
↪error,canceled}] [--execution_environment ID] [--failed BOOLEAN]
                             [--started STARTED] [--finished FINISHED] [--canceled_on␣
↪CANCELED_ON] [--elapsed ELAPSED] [--job_explanation TEXT]
                             [--execution_node TEXT] [--controller_node TEXT] [--work_
↪unit_id TEXT]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --type {project_update,inventory_update,job,ad_hoc_command,system_job,workflow_job}
                        only list unified_jobs with the specified type
  --created CREATED     only list unified_jobs with the specified created
  --modified MODIFIED   only list unified_jobs with the specified modified
  --name TEXT           only list unified_jobs with the specified name
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  --description TEXT    only list unified_jobs with the specified description
  --unified_job_template ID
                        only list unified_jobs with the specified unified_job_template
  --launch_type {manual,relaunch,callback,scheduled,dependency,workflow,webhook,sync,scm}
                        only list unified_jobs with the specified launch_type
  --status {new,pending,waiting,running,successful,failed,error,canceled}
                        only list unified_jobs with the specified status
  --execution_environment ID
                        only list unified_jobs with the specified execution_environment
  --failed BOOLEAN      only list unified_jobs with the specified failed
  --started STARTED     only list unified_jobs with the specified started
  --finished FINISHED   only list unified_jobs with the specified finished
  --canceled_on CANCELED_ON
                        only list unified_jobs with the specified canceled_on
  --elapsed ELAPSED     only list unified_jobs with the specified elapsed
  --job_explanation TEXT
                        only list unified_jobs with the specified job_explanation
  --execution_node TEXT
                        only list unified_jobs with the specified execution_node
  --controller_node TEXT
                        only list unified_jobs with the specified controller_node
  --work_unit_id TEXT   only list unified_jobs with the specified work_unit_id

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

unified_jobs get

```
usage: awx unified_jobs get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--conf.color␣
↪BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx unified_jobs get: the following arguments are required: id
```

users

```
usage: awx users [-h] action ...

positional arguments:
  action
    list
    create
    get
    modify
    delete
    grant
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
    revoke

options:
  -h, --help  show this help message and exit

awx users: the following arguments are required: action
```

users list

```
usage: awx users list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,jq,human}] [--
↪filter TEXT] [--conf.color BOOLEAN] [-v] [--type {user}] [--username TEXT]
                      [--first_name TEXT] [--last_name TEXT] [--email TEXT] [--is_
↪superuser BOOLEAN] [--password TEXT] [--last_login LAST_LOGIN]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --type {user}         only list users with the specified type
  --username TEXT       only list users with the specified username
  --first_name TEXT     only list users with the specified first_name
  --last_name TEXT      only list users with the specified last_name
  --email TEXT          only list users with the specified email
  --is_superuser BOOLEAN
                        only list users with the specified is_superuser
  --password TEXT       only list users with the specified password
  --last_login LAST_LOGIN
                        only list users with the specified last_login

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

users create

```
usage: awx users create [-h] --username TEXT [--first_name TEXT] [--last_name TEXT] [--
↪email TEXT] [--is_superuser BOOLEAN] [--is_system_auditor BOOLEAN] --password
                        TEXT

required arguments:
  --username TEXT       Required. 150 characters or fewer. Letters, digits and @/./+/-/_␣
↪only.
  --password TEXT       Field used to change the password.

options:
  -h, --help            show this help message and exit
  --first_name TEXT
  --last_name TEXT
  --email TEXT
  --is_superuser BOOLEAN
                        Designates that this user has all permissions without explicitly␣
↪assigning them.
  --is_system_auditor BOOLEAN

awx users create: the following arguments are required: --username, --password
```

users get

```
usage: awx users get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--conf.color␣
↪BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx users get: the following arguments are required: id
```

users modify

```
usage: awx users modify [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx users modify: the following arguments are required: id
```

users delete

```
usage: awx users delete [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx users delete: the following arguments are required: id
```

users grant

```
usage: awx users grant [-h]
                       (--organization ID | --project ID | --inventory ID | --team ID | -
↪-credential ID | --job_template ID | --workflow_job_template ID | --instance_group ID)
                       --role
                       {update,workflow_admin,job_template_admin,auditor,use,read,
↪inventory_admin,approval,credential_admin,member,adhoc,admin,notification_admin,
↪execute,execution_environment_admin,project_admin}
                       id

positional arguments:
  id                    The ID (or name) of the users to grant access to/from

options:
  -h, --help            show this help message and exit
  --organization ID     The ID (or name) of the target organization
  --project ID          The ID (or name) of the target project
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  --inventory ID        The ID (or name) of the target inventory
  --team ID             The ID (or name) of the target team
  --credential ID       The ID (or name) of the target credential
  --job_template ID     The ID (or name) of the target job_template
  --workflow_job_template ID
                        The ID (or name) of the target workflow_job_template
  --instance_group ID   The ID (or name) of the target instance_group
  --role {update,workflow_admin,job_template_admin,auditor,use,read,inventory_admin,
↪approval,credential_admin,member,adhoc,admin,notification_admin,execute,execution_
↪environment_admin,project_admin}
                        The name of the role to grant

awx users grant: the following arguments are required: id, --role
```

users revoke

```
usage: awx users revoke [-h]
                        (--organization ID | --project ID | --inventory ID | --team ID |␣
↪--credential ID | --job_template ID | --workflow_job_template ID | --instance_group ID)
                        --role
                        {project_admin,job_template_admin,inventory_admin,use,admin,
↪update,member,approval,adhoc,execute,read,execution_environment_admin,auditor,
↪credential_admin,workflow_admin,notification_admin}
                        id

positional arguments:
  id                    The ID (or name) of the users to revoke access to/from

options:
  -h, --help            show this help message and exit
  --organization ID     The ID (or name) of the target organization
  --project ID          The ID (or name) of the target project
  --inventory ID        The ID (or name) of the target inventory
  --team ID             The ID (or name) of the target team
  --credential ID       The ID (or name) of the target credential
  --job_template ID     The ID (or name) of the target job_template
  --workflow_job_template ID
                        The ID (or name) of the target workflow_job_template
  --instance_group ID   The ID (or name) of the target instance_group
  --role {project_admin,job_template_admin,inventory_admin,use,admin,update,member,
↪approval,adhoc,execute,read,execution_environment_admin,auditor,credential_admin,
↪workflow_admin,notification_admin}
                        The name of the role to revoke

awx users revoke: the following arguments are required: id, --role
```

workflow_approvals

```
usage: awx workflow_approvals [-h] action ...

positional arguments:
  action
    list
    get
    delete

options:
  -h, --help  show this help message and exit

awx workflow_approvals: the following arguments are required: action
```

workflow_approvals list

```
usage: awx workflow_approvals list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,jq,
↪human}] [--filter TEXT] [--conf.color BOOLEAN] [-v]
                                   [--type {workflow_approval}] [--created CREATED] [--
↪modified MODIFIED] [--name TEXT] [--description TEXT]
                                   [--unified_job_template ID] [--launch_type {manual,
↪relaunch,callback,scheduled,dependency,workflow,webhook,sync,scm}]
                                   [--status {new,pending,waiting,running,successful,
↪failed,error,canceled}] [--execution_environment ID] [--failed BOOLEAN]
                                   [--started STARTED] [--finished FINISHED] [--canceled_
↪on CANCELED_ON] [--elapsed ELAPSED] [--job_explanation TEXT]
                                   [--work_unit_id TEXT] [--timed_out BOOLEAN]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --type {workflow_approval}
                        only list workflow_approvals with the specified type
  --created CREATED     only list workflow_approvals with the specified created
  --modified MODIFIED   only list workflow_approvals with the specified modified
  --name TEXT           only list workflow_approvals with the specified name
  --description TEXT    only list workflow_approvals with the specified description
  --unified_job_template ID
                        only list workflow_approvals with the specified unified_job_
↪template
  --launch_type {manual,relaunch,callback,scheduled,dependency,workflow,webhook,sync,scm}
                        only list workflow_approvals with the specified launch_type
  --status {new,pending,waiting,running,successful,failed,error,canceled}
                        only list workflow_approvals with the specified status
  --execution_environment ID
                        only list workflow_approvals with the specified execution_
↪environment
  --failed BOOLEAN      only list workflow_approvals with the specified failed
  --started STARTED     only list workflow_approvals with the specified started
  --finished FINISHED   only list workflow_approvals with the specified finished
  --canceled_on CANCELED_ON
                        only list workflow_approvals with the specified canceled_on
  --elapsed ELAPSED     only list workflow_approvals with the specified elapsed
  --job_explanation TEXT
                        only list workflow_approvals with the specified job_explanation
  --work_unit_id TEXT   only list workflow_approvals with the specified work_unit_id
  --timed_out BOOLEAN   only list workflow_approvals with the specified timed_out

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

workflow_approvals get

```
usage: awx workflow_approvals get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--conf.
↪color BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx workflow_approvals get: the following arguments are required: id
```

workflow_approvals delete

```
usage: awx workflow_approvals delete [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx workflow_approvals delete: the following arguments are required: id
```

workflow_job_nodes

```
usage: awx workflow_job_nodes [-h] action ...

positional arguments:
  action
    list
    get

options:
  -h, --help  show this help message and exit

awx workflow_job_nodes: the following arguments are required: action
```

workflow_job_nodes list

```
usage: awx workflow_job_nodes list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,jq,
↪human}] [--filter TEXT] [--conf.color BOOLEAN] [-v]
                                   [--type {workflow_job_node}] [--created CREATED] [--
↪modified MODIFIED] [--extra_data JSON/YAML] [--inventory ID]
                                   [--execution_environment ID] [--job ID] [--workflow_
↪job ID] [--unified_job_template ID] [--all_parents_must_converge BOOLEAN]
                                   [--do_not_run BOOLEAN] [--identifier TEXT]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --type {workflow_job_node}
                        only list workflow_job_nodes with the specified type
  --created CREATED     only list workflow_job_nodes with the specified created
  --modified MODIFIED   only list workflow_job_nodes with the specified modified
  --extra_data JSON/YAML
                        only list workflow_job_nodes with the specified extra_data
  --inventory ID        only list workflow_job_nodes with the specified inventory
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  --execution_environment ID
                        only list workflow_job_nodes with the specified execution_
↪environment
  --job ID              only list workflow_job_nodes with the specified job
  --workflow_job ID     only list workflow_job_nodes with the specified workflow_job
  --unified_job_template ID
                        only list workflow_job_nodes with the specified unified_job_
↪template
  --all_parents_must_converge BOOLEAN
                        only list workflow_job_nodes with the specified all_parents_must_
↪converge
  --do_not_run BOOLEAN  only list workflow_job_nodes with the specified do_not_run
  --identifier TEXT     only list workflow_job_nodes with the specified identifier

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

workflow_job_nodes get

```
usage: awx workflow_job_nodes get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--conf.
↪color BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx workflow_job_nodes get: the following arguments are required: id
```

workflow_job_template_nodes

```
usage: awx workflow_job_template_nodes [-h] action ...

positional arguments:
  action
    list
    create
    get
    modify
    delete

options:
  -h, --help  show this help message and exit

awx workflow_job_template_nodes: the following arguments are required: action
```

workflow_job_template_nodes list

```
usage: awx workflow_job_template_nodes list [-h] [--all] [--order_by ORDER_BY] [-f {json,
↪yaml,jq,human}] [--filter TEXT] [--conf.color BOOLEAN] [-v]
                                            [--type {workflow_job_template_node}] [--
↪created CREATED] [--modified MODIFIED] [--extra_data JSON/YAML] [--inventory ID]
                                            [--execution_environment ID] [--workflow_job_
↪template ID] [--unified_job_template ID]
                                            [--all_parents_must_converge BOOLEAN] [--
↪identifier TEXT]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --type {workflow_job_template_node}
                        only list workflow_job_template_nodes with the specified type
  --created CREATED     only list workflow_job_template_nodes with the specified created
  --modified MODIFIED   only list workflow_job_template_nodes with the specified modified
  --extra_data JSON/YAML
                        only list workflow_job_template_nodes with the specified extra_
↪data
  --inventory ID        only list workflow_job_template_nodes with the specified␣
↪inventory
  --execution_environment ID
                        only list workflow_job_template_nodes with the specified␣
↪execution_environment
  --workflow_job_template ID
                        only list workflow_job_template_nodes with the specified␣
↪workflow_job_template
  --unified_job_template ID
                        only list workflow_job_template_nodes with the specified unified_
↪job_template
  --all_parents_must_converge BOOLEAN
                        only list workflow_job_template_nodes with the specified all_
↪parents_must_converge
  --identifier TEXT     only list workflow_job_template_nodes with the specified␣
↪identifier

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

workflow_job_template_nodes create

```
usage: awx workflow_job_template_nodes create [-h] [--extra_data JSON/YAML] [--inventory␣
↪ID] [--scm_branch TEXT] [--job_type {None,,run,check}] [--job_tags TEXT]
                                              [--skip_tags TEXT] [--limit TEXT] [--diff_
↪mode BOOLEAN] [--verbosity {None,0,1,2,3,4,5}] [--execution_environment ID]
                                              [--forks INTEGER] [--job_slice_count␣
↪INTEGER] [--timeout INTEGER] --workflow_job_template ID [--unified_job_template ID]
                                              [--all_parents_must_converge BOOLEAN] [--
↪identifier TEXT]

required arguments:
  --workflow_job_template ID
                        the ID of the associated workflow_job_template
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
options:
  -h, --help            show this help message and exit
  --extra_data JSON/YAML
                        a JSON or YAML string. You can optionally specify a file path e.
↪g., @path/to/file.yml
  --inventory ID        Inventory applied as a prompt, assuming job template prompts for␣
↪inventory
  --scm_branch TEXT
  --job_type {None,,run,check}
  --job_tags TEXT
  --skip_tags TEXT
  --limit TEXT
  --diff_mode BOOLEAN
  --verbosity {None,0,1,2,3,4,5}
  --execution_environment ID
                        The container image to be used for execution.
  --forks INTEGER
  --job_slice_count INTEGER
  --timeout INTEGER
  --unified_job_template ID
                        the ID of the associated unified_job_template
  --all_parents_must_converge BOOLEAN
                        If enabled then the node will only run if all of the parent␣
↪nodes have met the criteria to reach this node
  --identifier TEXT     An identifier for this node that is unique within its workflow.␣
↪It is copied to workflow job nodes corresponding to this node.

awx workflow_job_template_nodes create: the following arguments are required: --workflow_
↪job_template
```

workflow_job_template_nodes get

```
usage: awx workflow_job_template_nodes get [-h] [-f {json,yaml,jq,human}] [--filter␣
↪TEXT] [--conf.color BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx workflow_job_template_nodes get: the following arguments are required: id
```

workflow_job_template_nodes modify

```
usage: awx workflow_job_template_nodes modify [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx workflow_job_template_nodes modify: the following arguments are required: id
```

workflow_job_template_nodes delete

```
usage: awx workflow_job_template_nodes delete [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx workflow_job_template_nodes delete: the following arguments are required: id
```

workflow_job_templates

```
usage: awx workflow_job_templates [-h] action ...

positional arguments:
  action
    list
    create
    get
    modify
    delete
    launch
    associate
    disassociate

options:
  -h, --help    show this help message and exit

awx workflow_job_templates: the following arguments are required: action
```

workflow_job_templates list

```
usage: awx workflow_job_templates list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,
↪jq,human}] [--filter TEXT] [--conf.color BOOLEAN] [-v]
                                       [--type {workflow_job_template}] [--created␣
↪CREATED] [--modified MODIFIED] [--name TEXT] [--description TEXT]
                                       [--last_job_run LAST_JOB_RUN] [--last_job_failed␣
↪BOOLEAN] [--next_job_run NEXT_JOB_RUN]
                                       [--status {new,pending,waiting,running,successful,
↪failed,error,canceled,never updated,ok,missing,none,updating}]
                                       [--extra_vars JSON/YAML] [--organization ID] [--
↪survey_enabled BOOLEAN] [--allow_simultaneous BOOLEAN]
                                       [--ask_variables_on_launch BOOLEAN] [--inventory␣
↪ID] [--ask_inventory_on_launch BOOLEAN] [--ask_credential_on_launch BOOLEAN]
                                       [--ask_scm_branch_on_launch BOOLEAN] [--ask_limit_
↪on_launch BOOLEAN] [--webhook_service {,gitflic,github,gitlab,bitbucket_dc}]
                                       [--webhook_credential ID] [--ask_labels_on_launch␣
↪BOOLEAN] [--ask_skip_tags_on_launch BOOLEAN] [--ask_tags_on_launch BOOLEAN]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --type {workflow_job_template}
                        only list workflow_job_templates with the specified type
  --created CREATED     only list workflow_job_templates with the specified created
  --modified MODIFIED   only list workflow_job_templates with the specified modified
  --name TEXT           only list workflow_job_templates with the specified name
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  --description TEXT    only list workflow_job_templates with the specified description
  --last_job_run LAST_JOB_RUN
                        only list workflow_job_templates with the specified last_job_run
  --last_job_failed BOOLEAN
                        only list workflow_job_templates with the specified last_job_
↪failed
  --next_job_run NEXT_JOB_RUN
                        only list workflow_job_templates with the specified next_job_run
  --status {new,pending,waiting,running,successful,failed,error,canceled,never updated,
↪ok,missing,none,updating}
                        only list workflow_job_templates with the specified status
  --extra_vars JSON/YAML, -e JSON/YAML
                        only list workflow_job_templates with the specified extra_vars
  --organization ID     only list workflow_job_templates with the specified organization
  --survey_enabled BOOLEAN
                        only list workflow_job_templates with the specified survey_
↪enabled
  --allow_simultaneous BOOLEAN
                        only list workflow_job_templates with the specified allow_
↪simultaneous
  --ask_variables_on_launch BOOLEAN
                        only list workflow_job_templates with the specified ask_
↪variables_on_launch
  --inventory ID        only list workflow_job_templates with the specified inventory
  --ask_inventory_on_launch BOOLEAN
                        only list workflow_job_templates with the specified ask_
↪inventory_on_launch
  --ask_credential_on_launch BOOLEAN
                        only list workflow_job_templates with the specified ask_
↪credential_on_launch
  --ask_scm_branch_on_launch BOOLEAN
                        only list workflow_job_templates with the specified ask_scm_
↪branch_on_launch
  --ask_limit_on_launch BOOLEAN
                        only list workflow_job_templates with the specified ask_limit_on_
↪launch
  --webhook_service {,gitflic,github,gitlab,bitbucket_dc}
                        only list workflow_job_templates with the specified webhook_
↪service
  --webhook_credential ID
                        only list workflow_job_templates with the specified webhook_
↪credential
  --ask_labels_on_launch BOOLEAN
                        only list workflow_job_templates with the specified ask_labels_
↪on_launch
  --ask_skip_tags_on_launch BOOLEAN
                        only list workflow_job_templates with the specified ask_skip_
↪tags_on_launch
  --ask_tags_on_launch BOOLEAN
                        only list workflow_job_templates with the specified ask_tags_on_
↪launch

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

workflow_job_templates create

```
usage: awx workflow_job_templates create [-h] --name TEXT [--description TEXT] [--extra_
↪vars JSON/YAML] [--organization ID] [--survey_enabled BOOLEAN]
                                         [--allow_simultaneous BOOLEAN] [--ask_variables_
↪on_launch BOOLEAN] [--inventory ID] [--limit TEXT] [--scm_branch TEXT]
                                         [--ask_inventory_on_launch BOOLEAN] [--ask_
↪credential_on_launch BOOLEAN] [--ask_scm_branch_on_launch BOOLEAN]
                                         [--ask_limit_on_launch BOOLEAN] [--webhook_
↪service {,gitflic,github,gitlab,bitbucket_dc}] [--webhook_credential ID]
                                         [--ask_labels_on_launch BOOLEAN] [--ask_skip_
↪tags_on_launch BOOLEAN] [--ask_tags_on_launch BOOLEAN] [--skip_tags TEXT]
                                         [--job_tags TEXT]

required arguments:
  --name TEXT           Name of this workflow job template.

options:
  -h, --help            show this help message and exit
  --description TEXT    Optional description of this workflow job template.
  --extra_vars JSON/YAML, -e JSON/YAML
                        a JSON or YAML string. You can optionally specify a file path e.
↪g., @path/to/file.yml
  --organization ID     The organization used to determine access to this template.
  --survey_enabled BOOLEAN
  --allow_simultaneous BOOLEAN
  --ask_variables_on_launch BOOLEAN
  --inventory ID        Inventory applied as a prompt, assuming job template prompts for␣
↪inventory
  --limit TEXT
  --scm_branch TEXT
  --ask_inventory_on_launch BOOLEAN
  --ask_credential_on_launch BOOLEAN
  --ask_scm_branch_on_launch BOOLEAN
  --ask_limit_on_launch BOOLEAN
  --webhook_service {,gitflic,github,gitlab,bitbucket_dc}
                        Service that webhook requests will be accepted from
  --webhook_credential ID
                        Personal Access Token for posting back the status to the service␣
↪API
  --ask_labels_on_launch BOOLEAN
  --ask_skip_tags_on_launch BOOLEAN
  --ask_tags_on_launch BOOLEAN
  --skip_tags TEXT
  --job_tags TEXT

awx workflow_job_templates create: the following arguments are required: --name
```

workflow_job_templates get

```
usage: awx workflow_job_templates get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--
↪conf.color BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  -v, --verbose         print debug-level logs, including requests made

awx workflow_job_templates get: the following arguments are required: id
```

workflow_job_templates modify

```
usage: awx workflow_job_templates modify [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx workflow_job_templates modify: the following arguments are required: id
```

workflow_job_templates delete

```
usage: awx workflow_job_templates delete [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx workflow_job_templates delete: the following arguments are required: id
```

workflow_job_templates launch

```
usage: awx workflow_job_templates launch [-h] [--monitor] [--action-timeout ACTION_
↪TIMEOUT] [--wait] [--interval INTERVAL] [--extra_vars JSON/YAML] [--inventory ID]
                                         [--limit TEXT] [--scm_branch TEXT] [--labels␣
↪LABELS] [--credentials [ID, ID, ...]] [--skip_tags TEXT] [--job_tags TEXT]
                                         id

positional arguments:
  id

options:
  -h, --help            show this help message and exit
  --monitor             If set, prints stdout of the launched job until it finishes.
  --action-timeout ACTION_TIMEOUT
                        If set with --monitor or --wait, time out waiting on job␣
↪completion.
  --wait                If set, waits until the launched job finishes.
  --interval INTERVAL   If set with --monitor or --wait, amount of time to wait in␣
↪seconds between api calls. Minimum value is 2.5 seconds to avoid overwhelming the
                        api
  --extra_vars JSON/YAML, -e JSON/YAML
                        a JSON or YAML string. You can optionally specify a file path e.
↪g., @path/to/file.yml
  --inventory ID        the ID of the associated inventory
  --limit TEXT
  --scm_branch TEXT
  --labels LABELS
  --credentials [ID, ID, ...]
                        a list of comma-delimited credentials to associate (IDs or␣
↪unique names)
  --skip_tags TEXT
  --job_tags TEXT
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
awx workflow_job_templates launch: the following arguments are required: id
```

workflow_job_templates associate

```
usage: awx workflow_job_templates associate [-h] (--start_notification  | --success_
↪notification  | --failure_notification  | --approval_notification ) id

positional arguments:
  id

options:
  -h, --help            show this help message and exit
  --start_notification
                        The ID (or name) of the notification_template to associate
  --success_notification
                        The ID (or name) of the notification_template to associate
  --failure_notification
                        The ID (or name) of the notification_template to associate
  --approval_notification
                        The ID (or name) of the notification_template to associate

awx workflow_job_templates associate: the following arguments are required: id
```

workflow_job_templates disassociate

```
usage: awx workflow_job_templates disassociate [-h] (--start_notification  | --success_
↪notification  | --failure_notification  | --approval_notification ) id

positional arguments:
  id

options:
  -h, --help            show this help message and exit
  --start_notification
                        The ID (or name) of the notification_template to disassociate
  --success_notification
                        The ID (or name) of the notification_template to disassociate
  --failure_notification
                        The ID (or name) of the notification_template to disassociate
  --approval_notification
                        The ID (or name) of the notification_template to disassociate

awx workflow_job_templates disassociate: the following arguments are required: id
```

workflow_jobs

```
usage: awx workflow_jobs [-h] action ...

positional arguments:
  action
    list
    get
    delete
    monitor

options:
  -h, --help  show this help message and exit

awx workflow_jobs: the following arguments are required: action
```

workflow_jobs list

```
usage: awx workflow_jobs list [-h] [--all] [--order_by ORDER_BY] [-f {json,yaml,jq,human}
↪] [--filter TEXT] [--conf.color BOOLEAN] [-v] [--type {workflow_job}]
                              [--created CREATED] [--modified MODIFIED] [--name TEXT] [--
↪description TEXT] [--unified_job_template ID]
                              [--launch_type {manual,relaunch,callback,scheduled,
↪dependency,workflow,webhook,sync,scm}]
                              [--status {new,pending,waiting,running,successful,failed,
↪error,canceled}] [--failed BOOLEAN] [--started STARTED] [--finished FINISHED]
                              [--canceled_on CANCELED_ON] [--elapsed ELAPSED] [--job_
↪explanation TEXT] [--work_unit_id TEXT] [--workflow_job_template ID]
                              [--extra_vars JSON/YAML] [--allow_simultaneous BOOLEAN] [--
↪job_template ID] [--is_sliced_job BOOLEAN] [--inventory ID]
                              [--webhook_service {,gitflic,github,gitlab,bitbucket_dc}]␣
↪[--webhook_credential ID] [--webhook_guid TEXT]

options:
  -h, --help            show this help message and exit
  --all                 fetch all pages of content from the API when returning results␣
↪(instead of just the first page)
  --order_by ORDER_BY   order results by given field name, prefix the field name with a␣
↪dash (-) to sort in reverse eg --order_by='-name',multiple sorting fields may
                        be specified by separating the field names with a comma (,)
  --type {workflow_job}
                        only list workflow_jobs with the specified type
  --created CREATED     only list workflow_jobs with the specified created
  --modified MODIFIED   only list workflow_jobs with the specified modified
  --name TEXT           only list workflow_jobs with the specified name
  --description TEXT    only list workflow_jobs with the specified description
  --unified_job_template ID
                        only list workflow_jobs with the specified unified_job_template
  --launch_type {manual,relaunch,callback,scheduled,dependency,workflow,webhook,sync,scm}
                        only list workflow_jobs with the specified launch_type
  --status {new,pending,waiting,running,successful,failed,error,canceled}
                        only list workflow_jobs with the specified status
  --failed BOOLEAN      only list workflow_jobs with the specified failed
  --started STARTED     only list workflow_jobs with the specified started
  --finished FINISHED   only list workflow_jobs with the specified finished
  --canceled_on CANCELED_ON
                        only list workflow_jobs with the specified canceled_on
  --elapsed ELAPSED     only list workflow_jobs with the specified elapsed
  --job_explanation TEXT
                        only list workflow_jobs with the specified job_explanation
  --work_unit_id TEXT   only list workflow_jobs with the specified work_unit_id
  --workflow_job_template ID
                        only list workflow_jobs with the specified workflow_job_template
  --extra_vars JSON/YAML
                        only list workflow_jobs with the specified extra_vars
  --allow_simultaneous BOOLEAN
                        only list workflow_jobs with the specified allow_simultaneous
  --job_template ID     only list workflow_jobs with the specified job_template
  --is_sliced_job BOOLEAN
                        only list workflow_jobs with the specified is_sliced_job
  --inventory ID        only list workflow_jobs with the specified inventory
  --webhook_service {,gitflic,github,gitlab,bitbucket_dc}
                        only list workflow_jobs with the specified webhook_service
  --webhook_credential ID
                        only list workflow_jobs with the specified webhook_credential
  --webhook_guid TEXT   only list workflow_jobs with the specified webhook_guid

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made
```

workflow_jobs get

```
usage: awx workflow_jobs get [-h] [-f {json,yaml,jq,human}] [--filter TEXT] [--conf.
↪color BOOLEAN] [-v] id

positional arguments:
  id                    the ID (or unique name) of the resource

options:
  -h, --help            show this help message and exit

input/output formatting:
  -f {json,yaml,jq,human}, --conf.format {json,yaml,jq,human}
                        specify a format for the input and output
  --filter TEXT         specify an output filter (only valid with jq or human format)
  --conf.color BOOLEAN  Display colorized output. Defaults to True
  -v, --verbose         print debug-level logs, including requests made

awx workflow_jobs get: the following arguments are required: id
```

workflow_jobs delete

```
usage: awx workflow_jobs delete [-h] id

positional arguments:
  id          the ID (or unique name) of the resource

options:
  -h, --help  show this help message and exit

awx workflow_jobs delete: the following arguments are required: id
```

workflow_jobs monitor

```
usage: awx workflow_jobs monitor [-h] id

positional arguments:
  id

options:
  -h, --help  show this help message and exit

awx workflow_jobs monitor: the following arguments are required: id
```

11.1.6 Утилита awx-manage

awx-manage является утилитой командной строки, используемой для получения подробной информации о контроллере и управления им.

Предупреждение

Эта утилита не рекомендуется к регулярному применению, поскольку производит настройки на низком уровне. ЕЕ применение локально, то есть она действует на том узле, к которому вы подключились через SSH.

Команды утилиты awx-manage необходимо запускать на управляющем узле контроллера от имени пользователя awx или суперпользователя. Если управляющих узлов несколько, то запускать команду можно на любом из них.

Важно

Запуск команд awx-manage через сценарии Ansible не поддерживается.

Вызов утилиты имеет следующий вид:

```
sudo awx-manage <command> [<arguments>]
```

Для вывода справочной информации воспользуйтесь командой:

```
sudo awx-manage --help
```

Импорт инвентаря

Утилита awx-manage предоставляет администраторам возможность импортировать инвентарь в Automation Controller с помощью следующих шагов:

1. В контроллере создайте инвентарь, который будет служить местом назначения для импорта.
2. Подключитесь по SSH к управляющему узлу контроллера и выполните команду:

   ```
   sudo awx-manage inventory_import --source=<path_to_inventory> --inventory-id=<id>
   ```

   Здесь:

   - <path_to_inventory> – путь к файлу инвентаря, который будет использован в качестве источника.
   - <id> – идентификатор инвентаря, созданного ранее.

Команда inventory_import позволяет синхронизировать существующий инвентарь в контроллере с текстовым файлом инвентаря в формате INI или YAML, динамическим инвентарем или каталогом, содержащим такие файлы и инвентари.

При импорте можно использовать аргументы --overwrite и --overwrite-vars, чтобы управлять тем, как данные объединяются или заменяются.

По умолчанию данные из внешнего источника перезаписывают имеющиеся. Данные, которые отсутствуют во внешнем источнике, но присутствуют в контроллере, сохраняются.

При запуске утилиты с аргументом --overwrite используются только данные из внешнего источника. Данные, отсутствующие во внешнем источнике, будут удалены.

При запуске утилиты с аргументом --overwrite-vars используются переменные исключительно из внешнего источника. Узлы, которые отсутствуют во внешнем источнике, но присутствуют в контроллере, сохраняются.

Удаление устаревших данных

Команды cleanup_jobs и cleanup_activitystream используются для удаления данных о выполненных заданиях и данных ленты активности старше количества дней, указанного в аргументе --days:

- Удаление данных о выполненных заданиях:

  ```
  sudo awx-manage cleanup_jobs --days <n>
  ```

- Удаление данных ленты активности:

  ```
  sudo awx-manage cleanup_activitystream --days <n>
  ```

Удаление узла

Команда deprovision_instance используется для удаления узлов:

```
sudo awx-manage deprovision_instance --hostname=<hostname>
```

Здесь <hostname> – название удаляемого узла в инвентаре установщика платформы.

Примечание

Узел необходимо удалить из инвентаря установщика платформы, если он не должен появиться в кластере при повторном запуске сценария настройки.

Метрики узлов

Утилита awx-manage предоставляет возможность генерировать данные метрик узлов в формате CSV.

Создание файлов с данными метрик узла и информацией о кластере:

```
sudo awx-manage host_metric --csv
```

Чтобы упаковать все файлы в один архив, воспользуйтесь командой:

```
sudo awx-manage host_metric --tarball
```

Чтобы указать количество строк (<N>) для вывода в каждый файл, используйте аргумент --rows_per_file:

```
sudo awx-manage host_metric --rows_per_file <N>
```

11.1.7 Среда исполнения

Automation Controller использует *среды исполнения* для запуска заданий на исполняющих и гибридных узлах.

При развертывании контроллера в нем автоматически создаются две среды исполнения:

- Control Plane Execution Environment:
  - Используется при развертывании и обновлении платформы.
  - Используется для выполнения служебных заданий Control Plane.
  - Образ среды невозможно изменить через пользовательский интерфейс.
- Default execution environment:
  - Используется по умолчанию для выполнения заданий из шаблонов.
  - Его образ можно заменить на другой.

Для создания дополнительных сред исполнения можно использовать как образы из Automation Hub[193] (рекомендуется), так и собранные самостоятельно.

Поддерживается использование *полномочий* для доступа к реестрам образов, требующим аутентификацию пользователей.

Вместо среды по умолчанию можно назначить другую среду на следующих уровнях (в порядке возрастания приоритета использования):

1. контроллер;
2. организация;
3. проект;

[193] https://hub.astra-automation.ru/

4. шаблон задания.

При создании среды исполнения необходимо указать следующие параметры:

- Название. Должно быть уникальным в рамках контроллера.
- Образ. Ссылка на образ в реестре образов. Указывается в следующем формате:

```
<registry>/<name>:<tag>
```

  где:

  - <registry> – URL реестра образов;
  - <name> – название образа;
  - <tag> – тег, указывающий версию образа.

Для замены Control Plane Execution Environment на уже установленном контроллере воспользуйтесь утилитой `aa-setup` с дополнительной переменной `control_plane_execution_environment`, например:

```
./aa-setup -- -e control_plane_execution_environment=<image_name>
```

где <image_name> – полное название образа контейнера, как оно представлено в Private Automation Hub, например, `hub.example.com/aa-1.2/aa-full-ee:latest`.

После выполнения команды контроллер будет недоступен 2-3 минуты.

Параметры загрузки

Automation Controller позволяет задать настройки загрузки образа для каждой среды исполнения. Поддерживаются три варианта управления загрузкой:

- Перед запуском контейнера образ всегда загружается заново.

  Эта настройка может быть полезна при использовании в ссылке на образ тега `latest`.
- Перед запуском контейнера образ загружается только в том случае, если он отсутствует.
- Никогда – образ загружается только один раз, в момент создания среды исполнения.

Получение сведений о составе образа

Для получения сведений о версиях компонентов воспользуйтесь инструкцией из секции *Получение сведений об образах среды исполнения*.

ГЛАВА 12

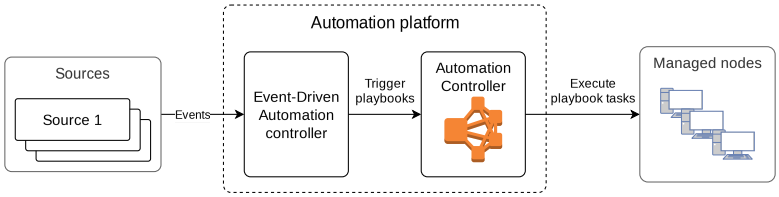
Обработка событий

Event-Driven Automation (EDA) является средством автоматизации обработки событий, создаваемых различными источниками, например, системами мониторинга. Этот инструмент позволяет отслеживать, идентифицировать и автоматически документировать различные события, либо реагировать на них в соответствии с заданными настройками.

12.1 Структура управления

Event-Driven Automation используется для автоматизации реакций на определенные события.

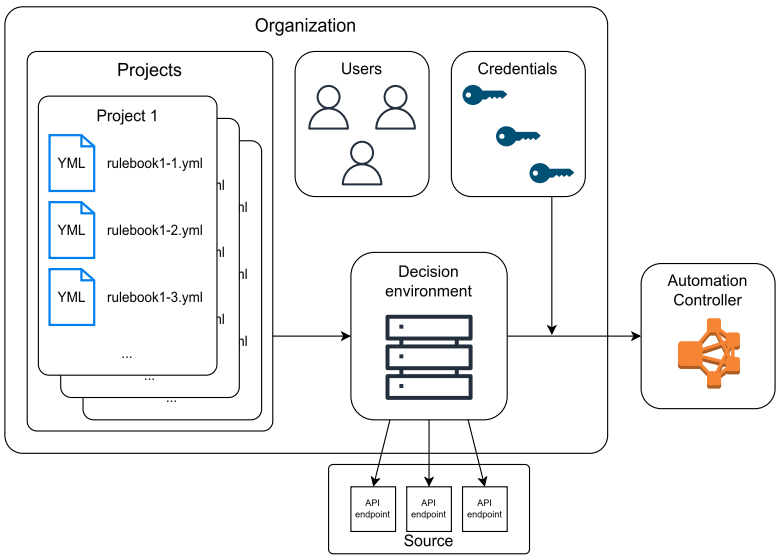
Типичный порядок обработки событий в Astra Automation показан на схеме:



Обработка событий происходит следующим образом:

1. Event-Driven Automation controller (далее – контроллер EDA) получает из источника уведомление об изменениях.
2. Контроллер EDA анализирует произошедшее изменение и принимает решение о реагировании на событие.
3. Если реакция на событие предполагает запуск набора сценариев, контроллер EDA запускает его в Automation Controller.
4. Automation Controller исполняет набор сценариев, тем самым реализуя реакцию платформы на событие, произошедшее во внешней системе.

Для обработки событий контроллер EDA использует компоненты, показанные на схеме:



Сюда входят:

- *Проекты* (projects) – наборы сводов правил, хранящихся в одном репозитории системы контроля версий Git[194].
- *Своды правил* (rulebooks) – файлы формата YAML, содержащие список наборов правил и сведения о среде принятия решений.
- *Наборы правил* (ruleset) – списки записей о том, что считать событиями и как на них реагировать.
- *Среды принятия решений* (DE, decision environment) – это контейнер, аналогичный *среде исполнения*, который включает все зависимости, коллекции и настройки, необходимые для выполнения сводов правил.
- *Полномочия* (credentials) – сведения, позволяющие получать доступ к сторонним ресурсам, например, репозиториям с кодом проектов, Automation Controller и так далее.

Перечисленные компоненты взаимодействуют между собой следующим образом:

1. Пользователь загружает проект, содержащий своды правил, в контроллер EDA и, при необходимости, загружает образ DE для этого проекта.
2. Пользователь активизирует необходимый свод правил в указанном DE.
3. DE периодически выполняет следующие проверки:
   - состояние каждого источника событий, указанного в наборах правил;
   - выполнение условий, означающих наступление события.
4. Если условия наступления события выполнены, контроллер EDA запускает соответствующий обработчик.

   Обработчик может, например, потребовать выполнение набора сценариев в Automation Controller. В этом случае контроллер Event-Driven Automation использует для подключения к Automation Controller API полномочие соответствующего типа.

12.1.1 Разграничение доступа

В соответствии с *ролевой моделью доступа* привилегии пользователям Event-Driven Automation предоставляют путем назначения им напрямую или через команды пользователей определенных ролей на следующие типы ресурсов:

- **Активации сводов правил** (Rulebook Activations).
- **Среды принятия решений** (Decision Environments).

[194] https://git-scm.com/

- **Полномочия** (Credentials).
- **Потоки событий** (Event Streams).
- **Проекты** (Projects).

Графический интерфейс позволяет назначать пользователю роли по отношению к отдельному ресурсу следующими способами:

- В окне конкретного ресурса. Инструкции по предоставлению доступа см. в описании конкретного ресурса. Например, чтобы назначить пользователю доступ к активации сводов правил, выполните действия, представленные в *описании активации сводов правил*.
- В окне *Пользователи (Users)*.

12.1.2 Полномочия

Полномочия используются для доступа к внешним ресурсам, управления узлами и проверки целостности содержимого. Полномочия могут содержать следующие данные:

- названия учетных записей пользователей;
- пароли;
- токены;
- приватные ключи SSH;
- публичные ключи GPG;
- и другие данные, которые можно использовать для получения доступа к ресурсам.

Конфиденциальные данные, используемые полномочием, называются *секретами*. Все секреты хранятся в базе данных контроллера EDA.

При создании любых полномочий обязательно должны быть указаны их название и тип. Если ни один из *встроенных типов полномочий* не подходит для выполнения поставленных задач, можно создать *собственный тип полномочий*.

Полномочие может быть связано с организацией, и тогда использовать его могут только пользователи этой организации. Если связи полномочия с организацией нет, то оно доступно для использования всеми пользователями контроллера EDA.

Встроенные типы полномочий

В Event-Driven Automation встроены следующие типы полномочий:

- *Astra Ansible Automation Platform*;
- *Basic Event Stream*;
- *Container Registry*;
- *Dynatrace Event Stream*;
- *ECDSA Event Stream*;
- *GitHub Event Stream*;
- *GitLab Event Stream*;
- *GPG Public Key*;
- *HMAC Event Stream*;
- *OAuth2 Event Stream*;
- *OAuth2 JWT Event Stream*;
- *Postgres*;
- *ServiceNow Event Stream*;
- *Source Control*;

- *Token Event Stream*;
- *Vault*.

Astra Ansible Automation Platform

Полномочия этого типа используются для доступа к *Автоматизация процессов*.

Поддерживается защита подключения с помощью TLS/SSL.

Basic Event Stream

Полномочия этого типа используются для управления потоками событий, использующих базовую аутентификацию.

Container Registry

Полномочия этого типа используются для доступа к различным хранилищам образов, например:

- *Управление контентом*;
- Docker[195];
- Quay[196];
- GitLab[197].

Поддерживается защита подключения с помощью TLS/SSL.

Dynatrace Event Stream

Полномочия этого типа используются для управления потоками событий типа Dynatrace.

ECDSA Event Stream

Полномочия этого типа используются для управления потоками событий, использующих ECDSA (Elliptic Curve Digital Signature Algorithm).

GitHub Event Stream

Полномочия этого типа используются для управления потоками событий GitHub, использующих HMAC для аутентификации.

GitLab Event Stream

Полномочия этого типа используются для управления потоками событий GitLab, использующих аутентификацию по токену (заголовок HTTP `X-Gitlab-Token`).

GPG Public Key

Полномочия этого типа используются для проверки подписи содержимого, полученного из внешних источников.

HMAC Event Stream

Полномочия этого типа используются для управления потоками событий, использующих HMAC. Для проверки требуется общий секретный ключ у отправителя и получателя. Подпись может быть отправлена в виде строки в формате hex или base64. Большинство отправителей используют специальный заголовок HTTP для отправки данных подписи.

[195] https://www.docker.com/
[196] https://quay.io/
[197] https://docs.gitlab.com/user/packages/container_registry/

OAuth2 Event Stream

Полномочия этого типа используются для управления потоками событий, использующих OAuth2. Для проверки необходимы идентификатор клиента и его учетные данные, а также доступ к серверу авторизации, чтобы анализировать отправляемый токен.

OAuth2 JWT Event Stream

Полномочия этого типа используются для управления потоками событий, использующих OAuth2 с JWT. Для этого требуется URL-адрес JWKS, который будет использоваться для получения открытого ключа и проверки входящего токена.

Postgres

Важно

Эта часть документации находится в стадии разработки.

ServiceNow Event Stream

Полномочия этого типа используются для управления потоками событий ServiceNow, использующих токен для аутентификации.

Source Control

Полномочия этого типа используются для доступа к системам контроля версий на основе Git и Subversion.

Поддерживается доступ с помощью ключа SSH или с использованием имени пользователя и пароля.

Token Event Stream

Полномочия этого типа используются для управления потоками событий, использующих аутентификацию по токену.

Обычно токен отправляется в HTTP-заголовке `Authorization`. Некоторые отправители используют для отправки токена специальный HTTP-заголовок.

Vault

Полномочия этого типа предоставляют доступ к секретам, защищенным с помощью Ansible Vault.

Если необходим доступ к нескольким хранилищам, необходимо предоставить соответствующие им идентификаторы Vault.

Собственные типы полномочий

Event-Driven Automation позволяет создавать собственные типы полномочий.

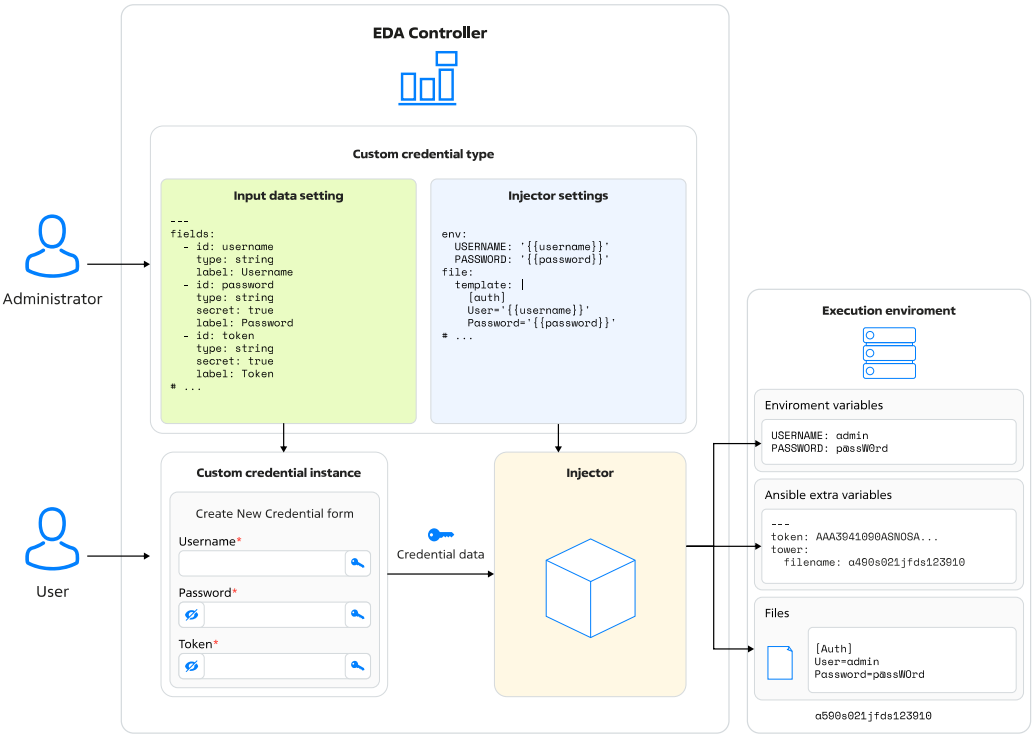
При создании собственного типа полномочий используются следующие понятия:

- *Входная настройка* определяет типы параметров, которые вводит пользователь при создании полномочия. Параметрами могут быть названия учетных записей, пароли, приватные ключи SSH и GPG, токены и тому подобное.
- *Инжектор* – механизм, выполняющий преобразование параметров полномочия в нужный формат.
- *Внедрение данных (инжекция)* – процесс передачи настроенного полномочия в среду принятия решений при активации свода правил.

Создание собственного типа полномочий имеет следующие особенности:

- Каждый тип полномочий должен содержать описание своих параметров (переменных) и параметры инжектора.
- Контроллер EDA не проверяет наличие коллизий названий переменных окружения, дополнительных переменных Ansible и пространств имен.
- Следует избегать использования названий переменных, начинающихся с EDA_, поскольку они зарезервированы.

Работа контроллера EDA с собственными типами полномочий показана на схеме:



1. При создании собственного типа полномочий администратор задает типы параметров и настройку инжектора.
2. Пользователи создают необходимые экземпляры полномочий определенного типа.
3. При запуске заданий введенные пользователями данные полномочия передаются в инжектор.
4. Инжектор передает данные полномочия в среду принятия решений.

   Возможны следующие способы передачи данных полномочия в среду принятия решений:

   - переменная окружения;
   - дополнительная переменная (extra vars) Ansible;
   - текстовый файл, в том числе с возможностью генерации содержимого с использованием шаблонизатора Jinja.

   Способ передачи данных определяется настройками инжектора. Одни и те же данные полномочия могут быть переданы в среду принятия решений сразу несколькими способами, например, в переменной окружения, дополнительной переменной Ansible и содержимом текстового файла.

Входная настройка

Входная настройка типа полномочий определяет параметры полей ввода, используемых при создании полномочия.

Типовые настройки входных параметров состоят из обязательного блока `fields` и необязательного блока `required`.

В блоке fields содержится список записей с описанием полей ввода данных полномочия. Каждая запись может состоять из следующих полей:

- id – идентификатор поля. Значение, указанное в этом поле, используется для передачи данных в инжектор.

  Обязательное поле.

- label – название поля при заполнении сведений о полномочии через графический интерфейс.

  Обязательное поле.

- type – тип поля:

  - string – строка;
  - boolean – логическое.

  Значение по умолчанию: string.

- format – дополнительная проверка корректности значения, указанного в поле типа string.

  Поддерживаются следующие значения:

  - ssh_private_key – приватный ключ SSH;
  - url – URI.

- multiline – логическое значение для полей типа string. При значении true поле позволяет вводить многострочный текст.
- choices – массив доступных значений для полей типа string. Если массив не пуст, при заполнении поля через графический интерфейс значение нужно выбрать из списка, а не вводить вручную.
- default – значение по умолчанию.
- secret – если равно true, вводимое значение является секретом.

  Для защиты секретов в полях ввода используется маскировка символов, а в таблице базы данных контроллера EDA – защитное преобразование.

- help_text – подсказка, которая выводится рядом с полем при заполнении сведений о полномочии через графический интерфейс.

В блоке required содержится список идентификаторов полей, обязательных для заполнения.

Конфигурация внедрения данных

Для доступа к значению, полученному из входных данных, следует в двойных фигурных скобках указать идентификатор соответствующего поля, например:

```
---
extra_vars:
  auth_token: '{{ auth_token }}'
```

Дополнительной переменной Ansible auth_token присваивается значение поля входных данных с идентификатором auth_token.

Конфигурация внедрения данных может содержать следующие блоки:

- file – шаблон, на основе которого генерируется содержимое временного файла.

  Имя файла генерируется случайным образом. Оно хранится в переменной Ansible tower.filename.

  Если используется несколько полей для загрузки файлов, их следует описать в отдельных переменных, названия которых начинаются с префикса template.. Для получения имени файла следует обратиться к соответствующему значению словаря tower.filename, например:

```
---
file:
  template.cert_file: '{{ tls_cert }}'
  template.key_file: '{{ tls_key }}'
env:
  TLS_CERT_FILE_NAME: '{{ tower.filename.cert_file }}'
  TLS_KEY_FILE_NAME: '{{ tower.filename.key_file }}'
```

Здесь для хранения сертификата и соответствующего ему ключа создаются два временных файла, имена которых сохраняются в переменных `tower.filename.cert_file` и `tower.filename.key_file` соответственно.

- env – список названий переменных окружения и соответствующих им значений.
- `extra_vars` – список названий дополнительных переменных Ansible и соответствующих им значений.

Важно

- Блок `extra_vars` обязателен к указанию в конфигурации внедрения данных.
- При создании `extra_vars` не используйте `eda` или `ansible` в качестве названий ключей, так как это может привести к внутренним конфликтам.

Примеры

Пусть для работы с Kafka (источник событий) и Slack (целевая система для отправки уведомлений) нам потребуются следующие учетные данные:

- URL брокера Kafka.

  Значение передается в среду принятия решений через переменную окружения `KAFKA_BROKER_URL`.

- Название топика Kafka.

  Значение передается в среду принятия решений через переменную окружения `KAFKA_TOPIC`.

- Ключ API для Kafka.

  Значение передается в среду принятия решений через файл, название которого хранится в переменной окружения `KAFKA_API_KEY_FILE`.

- URL веб-обработчика Slack.

  Значение передается в среду принятия решений через файл, название которого хранится в переменной окружения `SLACK_WEBHOOK_URL_FILE`.

Параметр	Идентификатор	Тип	Обязательное
Адрес Kafka Broker	`kafka_broker_url`	`string`	Да
Название Kafka Topic	`kafka_topic`	`string`	Да
API Key для Kafka	`kafka_api_key`	`string`	Да
Webhook URL для Slack	`slack_webhook_url`	`string`	Да

Настройки собственного типа полномочий в этом случае можно задать следующим образом:

Список 1: Входная настройка

```
---
fields:
  - id: kafka_broker_url
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
    type: string
    label: Kafka Broker URL
    help_text: Enter the URL of the Kafka broker.

  - id: kafka_topic
    type: string
    label: Kafka Topic
    help_text: Enter the name of the Kafka topic to consume events from.

  - id: kafka_api_key
    type: string
    label: Kafka API Key
    secret: true
    help_text: Enter the API key for authenticating with Kafka.

  - id: slack_webhook_url
    type: string
    label: Slack Webhook URL
    secret: true
    format: url
    help_text: Enter the Slack webhook URL for sending messages.

required:
  - kafka_broker_url
  - kafka_topic
  - kafka_api_key
  - slack_webhook_url
```

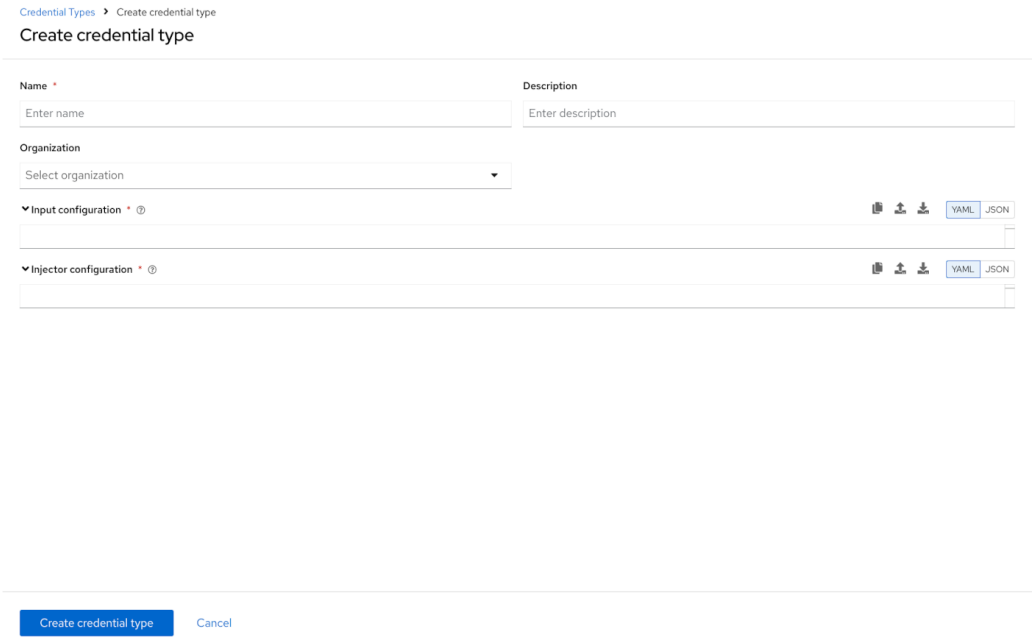
Список 2: Конфигурация внедрения данных

```
---
env:
  KAFKA_BROKER_URL: '{{ kafka_broker_url }}'
  KAFKA_TOPIC: '{{ kafka_topic }}'
  KAFKA_API_KEY_FILE: '{{ tower.filename.kafka_api_key }}'
  SLACK_WEBHOOK_URL_FILE: '{{ tower.filename.slack_webhook_url }}'

file:
  template.kafka_api_key: '{{ kafka_api_key }}'
  template.slack_webhook_url: '{{ slack_webhook_url }}'

extra_vars:
  kafka_broker_url: '{{ kafka_broker_url }}'
  kafka_topic: '{{ kafka_topic }}'
  kafka_api_key: '{{ kafka_api_key }}'
  slack_webhook_url: '{{ slack_webhook_url }}'
```

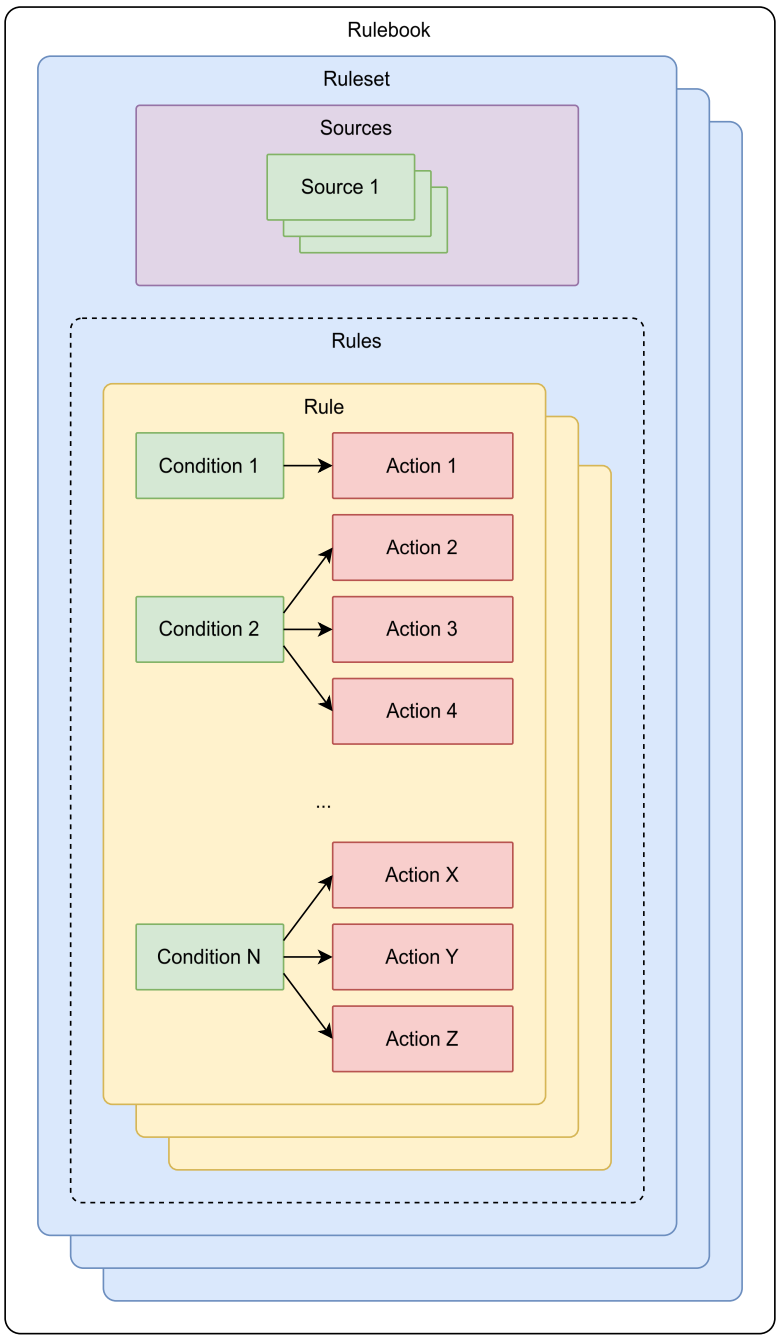
В графическом интерфейсе EDA форма для создания полномочий этого типа имеет следующий вид:



12.2 Своды правил

Свод правил (rulebook) – это файл формата YAML, содержащий список *наборов правил* (ruleset).

Структура типового свода правил показана на схеме:



Свод правил содержит несколько *наборов правил* (ruleset). Каждый набор правил содержит сведения об *источниках* событий (source) и *правила* (rule). Каждое правило состоит из списка *условий* (conditions), выполнение которых приводит к запуску тех или иных *действий* (actions).

Пример файла, содержащего свод правил:

```
---
- name: Example ruleset 1
  hosts: all
  sources:
    - name: Source 1
      ansible.eda.url_check:
        urls:
          - https://example.com/heartbeat/
        delay: 10

  rules:
    - name: Rule 1
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
      condition: event.status == "down"
      action:
        run_job_template:
          name: "Reload server"
          organization: "Default"

- name: Example ruleset 2
  hosts: database
  sources:
    - name: PG Status
      ansible.eda.webhook:
        host: https://source.example.com/api/webhook/
        port: 443
        keyfile: /path/to/keyfile

  rules:
    - name: Rule 2
      conditinon:
        any:
          - event.payload.connections_free <= 10
          - event.payload.mem_free < 2048
      action:
        run_job_template:
          name: "Restart PostgreSQL server"
          organization: "Default"
```

Этот свод правил содержит два набора правил.

- Example ruleset 1

  Состоит из источника «Source 1» и правила «Rule 1».

  Источник «Source 1» каждые 10 секунд проверяет код возврата при обращении к указанному URI. Если код возврата отличается от 200, либо его вовсе не удается получить за указанное время, контроллер EDA создает событие, в свойствах которого заполняет поле `status` значением `down`.

  Реакция на указанное событие описана в правиле «Rule 1». Оно проверяет поля `status` на равенство значению `down`. Если условие выполнено, EDA подключается к Automation Controller и запускает задание на основе шаблона Reload server, принадлежащего организации Default.

- Example ruleset 2

  Состоит из источника «PG Status» и правила «Rule 2».

  Источник «PG Status» получает данные через веб-обработчик, отслеживающий обращения к точке доступа API `https://source.example.com/api/webhook/`. Содержимое ответа сохраняется в поле `payload`.

  Правило «Rule 2» проверяет выполнение двух условий:

  - значение `payload.connections_free` меньше или равно 10 (исчерпание свободных подключений к серверу PostgreSQL);
  - значение `payload.mem_free` меньше 2048 (исчерпание свободной оперативной памяти).

  Если выполняется хотя бы одно из указанных условий, контроллер EDA подключается к Automation Controller и запускает задание на основе шаблона Restart PostgreSQL server, принадлежащего организации Default.

Важно

Предполагается, что контроллер EDA интегрирован с Automation Controller, в котором существуют организация Default и принадлежащие ей шаблоны заданий Reload server и

Restart PostgreSQL server.

12.2.1 Наборы правил

Каждый набор правил содержит следующие поля:

- `name` – уникальное название набора правил.

  Это поле обязательно для заполнения.

  Важно

  Названия наборов правил должны быть уникальны на уровне свода. Это необходимо для обмена сообщениями между наборами правил во время их выполнения.

  Каждый набор правил выполняется в отдельной сессии механизма исполнения правил. События и факты хранятся отдельно для каждого набора правил правил. Во время выполнения набор правил может отправить событие или факт сам себе или другому набору правил из этого же свода правил.

- `sources` – список *источников* событий.

  Это поле обязательно для заполнения.

- `rules` – список *правил*.

  Это поле обязательно для заполнения.

- `hosts` – список узлов, к которым применяются действия, описанные в блоке `rules.action` или `rules.actions`.

  Это поле обязательно для заполнения.

- `gather_facts` – сбор фактов Ansible при запуске правила.

  Контроллер EDA собирает факты Ansible в момент запуска свода правил. Собранные данные сохраняются и могут быть использованы в соответствующих выражениях.

  Данные о каждом узле хранятся отдельно. Название узла хранится в свойстве `fact.meta.hosts`, например:

```
- name: Ruleset example
  hosts: all
  gather_facts: true
  sources:
    # ...
  rules:
    - name: Host specific rule
      condition:
        all:
          - fact.ansible_os_family == "linux"
          - fact.meta.hosts == "node1.example.com"
          - event.i == 4
  # ...
```

  По умолчанию факты не собираются, то есть `gather_facts: false`.

- `default_events_ttl` – длительность периода, в течение которого хранятся данные о частично наступивших событиях.

  Значение этого поля задается в следующем формате:

```
<count> <unit>
```

  Здесь:

  - <count> – количество;

  - <unit> – единица измерения:
    * `seconds` – секунды;
    * `minutes` – минуты;
    * `hours` – часы;
    * `days` – дни.

  Значение по умолчанию: `2 hours` (2 часа).

- `execution_strategy` – стратегия выполнения действий:
  - `sequential` – действия выполняются в порядке очереди;
  - `parallell` – действия выполняются параллельно.

  Значение по умолчанию: `sequential`.

Набор правил должен содержать хотя бы одно расширение для работы с источниками событий. Настройки каждого расширения должны быть указаны после типа расширения. Расширение источника может быть настроено с помощью фильтра событий, который позволяет изменить данные перед передачей их на исполнение правил. Утилита `ansible-rulebook` запускает расширение источника и исполняет его в фоновом режиме, помещая события в очередь исполнения. Когда расширение источника завершает работу, оно автоматически создает соответствующее действие и прерывает выполнение набора правил. Это, в свою очередь, приводит к завершению работы `ansible-rulebook`.

Набор правил должен содержать хотя бы одно правило. Контроллер EDA выполняет все необходимые проверки для правил, основанных на входящих событиях. Если условия в правиле выполняются, контроллер EDA выполняет соответствующие действия.

Действием может быть запуск набора сценариев или модулей, возбуждение другого события или создание факта в том же самом или другом наборе правил.

Исполнение набора правил завершается в следующих случаях:

- расширение источника создает событие завершения работы;
- одно из выполненных правил создает событие завершения работы.

12.2.2 Распространение

Рекомендуется распространять своды правил в составе коллекций. В этом случае свод правил должен быть размещен в каталоге `extensions/eda/rulebooks`, а ссылка на него указана в аргументах CLI.

12.2.3 Источники

Для обработки событий, поступающих из различных типов источников, контроллер EDA использует расширения. Эти расширения могут храниться локально, однако, более эффективно их распространение в составе коллекций.

Коллекция ansible.eda[198] предоставляет следующие расширения для работы с различными типами источников:

- `alertmanager` – получение событий через веб-обработчики Prometheus Alertmanager[199];
- `azure_service_bus` – получение событий от служб Microsoft Azure;
- `file_watch` – отслеживание состояния файловой системы;
- `file` – загрузка фактов из файла YAML и их обновление при изменении содержимого файла;
- `kafka` – обработка событий, хранящихся в топиках Apache Kafka;
- `range` – генерация событий с индексами из указанного диапазона;

[198] https://github.com/ansible/event-driven-ansible
[199] https://prometheus.io/docs/alerting/latest/alertmanager/

- `tick` – генерация событий с индексами, которые задает бесконечно увеличивающийся счетчик;
- `url_check` – периодическая проверка HTTP-статусов указанных URL;
- `webhook` – использование веб-обработчиков для получения сведений о событиях.

Примечание

Как правило, типы источников `tick`, `file` и `range` используются при разработке и тестировании.

Список 3: Пример использования источника `ansible.eda.webhook`

```
- name: Listen port 5000 with webhook
  hosts: all
  sources:
    - ansible.eda.webhook:
        host: 192.168.56.1
        port: 5000
  # ...
```

Независимо от типа используемого расширения, полученные им данные передаются в правила через объект `event`.

Особенности использования различных типов источников описаны в *справочнике*.

12.2.4 Правила

Каждое правило содержит следующие поля:

- `name` – название правила.

  Это поле обязательно для заполнения.

  Важно

  Названия правил должны быть уникальны на уровне свода.

  При формировании названий правил допускается использовать шаблонизатор Jinja2.

- `condition` – условие или список условий, выполнение которых считается наступлением события.

  Это поле обязательно для заполнения.

- `action` – действие, которое должно быть выполнено при наступлении события, например:

```
rules:
  - name: Rule with one action
    condition: event.outage == true
    action:
      run_job_templtae:
        name: "Job template example"
        organization: "Default"
```

- `actions` – список действий, которые должны быть выполнены при наступлении события, например:

```
rules:
  - name: Rule with multiple actions
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
    condition: event.outage == true
    actions:
      - run_playbook:
          name: playbook.yml
      - print_event:
          pretty: true
```

- `enabled` – разрешение на использование правила.

  Значение по умолчанию: `true` (использование правила разрешено).

Особенности использования действий описаны в *справочнике*.

Условия

Условие – выражение, результатом вычисления которого могут быть только логические значения `true` (условие выполнено) и `false` (условие не выполнено).

Условие может содержать:

- одно условие:

```
condition: event.host == 'localhost'
```

- несколько условий, которые должны быть выполнены одновременно:

```
condition:
  all:
    - event.target_os == "linux"
    - event.tracking_id == 123
```

- несколько условий, из которых должно быть выполнено хотя бы одно:

```
condition:
  any:
    - event.target_os == "linux"
    - event.target_os == "windows"
```

- несколько условий, из которых хотя бы одно *не* выполняется:

```
condition:
  not_all:
    - event.target_os == "linux"
    - event.target_os == "freebsd"
```

Условия могут использовать информацию из следующих источников:

- полученное событие;
- событие, сохранное в правиле ранее;
- долговременные факты о системе;
- переменные, переданные пользователем при запуске правила.

Для установки значений переменных или получения данных используйте следующие префиксы:

- `event` – для доступа к данным текущего события;
- `events` – для установки значений переменных и доступа к данным внутри правила;
- `fact` – для доступа к данным, заданным с помощью `set_facts` в своде правил;
- `facts` – для установки значений переменных и доступа к данным внутри правила;
- `vars` – для доступа к значениям переменных, заданных в опциях CLI `--vars` и `--env-vars`.

Важно

Выражения не могут содержать замены с использованием шаблонов Jinja при доступе к переменным, переданным в опциях CLI, так как при этом теряется информация о типе данных, и контроллер EDA не может обработать условие. Вместо этого используйте поле `vars` для доступа к данным, переданным в опциях CLI.

В одном выражении нельзя использовать данные из `fact` и `event`. Вместо этого необходимо предварительно передать данные в `facts` с помощью `set_fact`, например:

```
---
rules:
  - name: Using fact and event together
    condition:
      all:
        - facts.storage << fact.custom.free_space is defined
        - event.space <= facts.storage.custom.free_space
```

Обращение к нужным полям может быть записано через точку (точечная нотация) или в квадратных скобках (скобочная нотация). Следующие два правила идентичны:

```
---
rules:
  - name: Rule with dot notation
    condition: event.data.nested == true
    action:
      debug:

  - name: Rule with bracket notation
    condition: event.data["nested"] == true
    action:
      debug:
```

Использование скобочной нотации рекомендуется для случаев, когда название поля содержит точки и другие специальные символы, например:

```
---
rules:
  - name: Rule for special characters field
    condition: event.resource.metadata.labels["app.kubernetes.io/name"] == "example"
    action:
      debug
```

Также скобочная нотация может быть использована для обращения к элементам списков. Нумерация элементов списка при этом начинается с нуля:

```
---
rules:
  - name: Rule for checking second list item
    condition: event.packages[1] == "2-st package"
    action:
      debug:
```

Отрицательные значения позволяют обращаться к элементам списка, начиная отсчет с конца, как это принято в Python.

Типы данных

Возможно использование следующих типов данных:

- целые числа (integers);
- строки (strings);
- логические значения (booleans);

- вещественные числа (float), записанные в точечной или научной нотации, например, `10.5` или `105E-1`;
- `null`.

Операторы

Операторы используются для создания логических выражений. Для некоторых операторов, работающих со строками, поддерживается флаг <ignorecase>. Он управляет игнорированием регистра символов при поиске и по умолчанию равен `true` (регистр символов не имеет значения).

При описании выражений допускается использовать следующие операторы:

- == – равенство строк, чисел или логических значений.
- `!=` – неравенство строк, чисел или логических значений.
- > – число слева больше числа справа.
- < – число слева меньше числа справа.
- >= – число слева больше или равно числу справа.
- <= – число слева меньше или равно числу справа.
- + – сложение чисел.
- `-` – разность чисел.
- * – умножение чисел.
- and – логическое «и», используется для создания сложных выражений.
- `or` – не исключающее «или», используется для создания сложных выражений.
- `in` – проверка присутствия значения из левой части выражения в списке из правой части.
- `not in` – проверка отсутствия значения из левой части выражения в списке из правой части.
- `contains` – проверка присутствия в списке из левой части выражения значения, указанного в правой части.
- `not contains` – проверка отсутствия в списке из левой части выражения значения, указанного в правой части.
- `is defined` – проверка существования переменной.
- `is not defined` – проверка отсутствия переменной.
- `is match(<pattern>, <ignorecase>)` – проверка начала строки на совпадение с указанным шаблоном <pattern>.

  Допускается использование регулярных выражений.
- `is not match(<pattern>, <ignorecase>)` – проверка начала строки на несовпадение с указанным шаблоном <pattern>.

  Допускается использование регулярных выражений.
- `is search(<pattern>, <ignorecase>)` – проверка строки на наличие подстроки, удовлетворяющей шаблону <pattern>.

  Допускается использование регулярных выражений.
- `is not search(<pattern>, <ignorecase>)` – проверка строки на отсутствие подстроки, удовлетворяющей шаблону <pattern>.

  Допускается использование регулярных выражений.
- `is regex(<regexp>, <ignorecase>)` – проверка строки на наличие подстроки, удовлетворяющей регулярному выражению <regexp>.

- `is not regex(<regexp>, <ignorecase>)` – проверка строки на отсутствие подстроки, удовлетворяющей регулярному выражению <regexp>.
- `is select(<operator>, <value>)` – проверка наличия в списке из левой части выражения хотя бы одного элемента, для которого выполняется условие, состоящее из оператора <operator> и значения <value>.
- `is not select(<operator>, <value>` – проверка отсутствия в списке из левой части выражения элементов, для которых выполняется условие, состоящее из оператора <operator> и значения <value>.
- `is selectattr(<key>, <operator>, <value>)` – проверка наличия в списке из левой части выражения хотя бы одного элемента, у которого для значения поля <key> выполняется условие, состоящее из оператора <operator> и значения <value>.
- `is not selectattr(<key>, <operator>, <value>)` – проверка отсутствия в списке из левой части выражения элементов, у которых для значения поля <key> выполняется условие, состоящее из оператора <operator> и значения <value>.
- << – оператор присваивания, используемый для сохранения событий или фактов.
- `not` – оператор инвертирования логического выражения.

12.2.5 Примеры

В этом примере рассматривается несложный свод правил:

Список 4: Пример свода правил

```
---
- name: Rulebook example
  hosts: all
  sources:
    - ansible.eda.url_check:
        urls:
          - https://192.168.56.1/api/heartbeat/
        delay: 10
        verify_ssl: false

  rules:
    - name: Check headers status
      condition: event.url_check.status == "down"
      action:
        run_job_template:
          name: "Restart service"
          organization: "Default"
```

Для подключения к источнику используется расширение `ansible.eda.url_check` со следующими параметрами:

- URI источника – `https:/192.168.56.1/api/heartbeat/`;
- периодичность проверки – каждые 10 секунд;
- проверка сертификата SSL – выключена.

Расширение `ansible.eda.url_check` добавляет в объект `event` свойство `status`, которое может принимать только два значения:

- `up` – если HTTP-код ответа равен 200 при обращении к указанному URI;
- `down` – во всех остальных случаях, в том числе при отсутствии какого-либо ответа со стороны источника.

В правиле `Check header status` выполняется проверка статуса. Если он равен `down`, значит, нужный URI недоступен. Для восстановления работоспособности сервера, на котором он размещен, контроллера EDA подключается к Automation Controller и запускает задание на основе шаблона Restart service, принадлежащего организации Default.

12.3 Применение Event-Driven Automation

Важно

Эта часть документации находится в стадии разработки.

12.4 Графический интерфейс

Раздел **Обработка событий** (Automation Decisions) предназначен для настройки и управления автоматизации на платформе Astra Automation на основе событий.

С помощью этого раздела можно выполнять следующие действия:

- создавать правила обработки событий и управлять ими;
- запускать и контролировать активации сводов правил;
- управлять проектами;
- создавать и настраивать среды принятия решений;
- организовывать потоки событий.

Ниже описаны основные разделы интерфейса **Обработка событий** (Automation Decisions).

12.4.1 Аудит правил

Раздел **Аудит правил** (Rule Audit) позволяет просматривать историю срабатывания всех правил, активированных в системе.

В этом разделе отображается следующая информация:

- список всех случаев, когда входящее событие соответствовало условию правила;
- дата и время срабатывания;
- к какой активации свода правил относится данное срабатывание.

Этот раздел помогает анализировать корректность работы правил и отслеживать, какие действия выполнялись платформой.

12.4.2 Активации сводов правил

Раздел **Активации сводов правил** (Rulebook Activations) содержит все наборы (своды) правил, запущенные в средах принятия решений. Каждая активация – это фоновый процесс, исполняющий конкретный свод правил.

В данном разделе доступны следующие возможности:

- создание, редактирование и удаление активаций сводов правил;
- просмотр состояния и параметров активации (связанные проекты, своды правил, среды принятия решений, политики перезапуска, переменные);
- управление жизненным циклом активации.

Это основной инструмент запуска и контроля автоматизации процессов на основе событий.

12.4.3 Проекты

Раздел **Проекты** (Projects) отображает коллекции сценариев и сводов правил, используемые для автоматизации на основе событий. Проекты должны быть связаны с репозиториями исходного кода (*VCS*).

Для каждого проекта доступен следующий набор действий:

- синхронизация содержимого;
- настройка командного и пользовательского доступа на доступ к проекту.

12.4.4 Среды принятия решений

Раздел **Среды принятия решений** (Decision Environments) предназначен для управления контейнерными образами, внутри которых исполняются своды правил. Среды принятия решений обеспечивают стандартизацию зависимостей и воспроизводимость выполнения.

В данном разделе доступны следующие действия:

- выбор и использование готовых образов;
- создание собственных образов с необходимыми зависимостями;
- настройка командного и пользовательского доступа на доступ к образу.

12.4.5 Потоки событий

Раздел **Потоки событий** (Event Streams) связан с оркестрацией событий и последовательностей действий реагирования системы на различные события.

С помощью потоков событий выстраиваются сложные сценарии:

- своды правил задают цепочку условий и действий;
- при срабатывании конкретного события запускается сценарий, шаблон рабочего процесса или другое действие.

Этот раздел позволяет управлять сценариями реагирования на внешние или внутренние события.

12.4.6 Инфраструктура

Раздел **Инфраструктура** (infrastructure) обеспечивает управление доступом к автоматизации на основе событий. В разделе можно управлять полномочиями (Credentials) и их типами (Credential Types) для обеспечения безопасного доступа к инфраструктуре и внешним сервисам.

Аудит правил

Окно **Аудит правил** (Rule Audit) позволяет в реальном времени отслеживать и анализировать выполнение правил. Данные аудита правил будут заполнены после запуска активации *свода правил*. Для перехода к окну выберите на панели навигации *Обработка событий ▸ Аудит правил* (*Automation Decision ▸ Rule Audit*).

Таблица аудита правил

Внешний вид окна **Аудит правил** (Rule Audit) представлен на схеме:

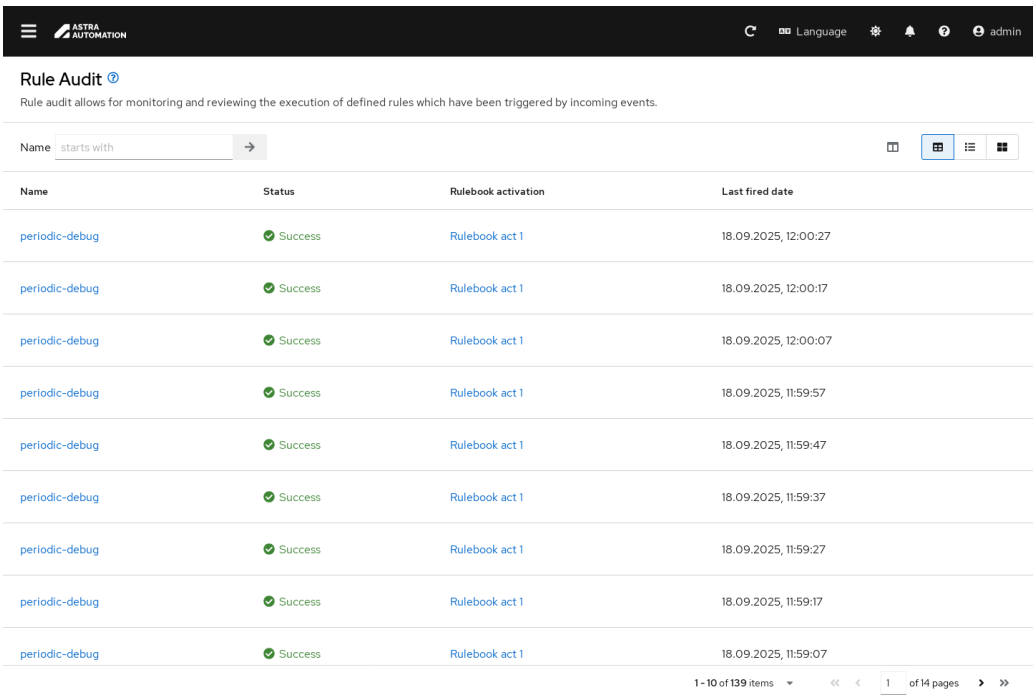


Таблица аудита правил состоит из следующих колонок:

- **Название** (Name) – название правила, которое было определено в *своде правил*.
- **Статус** (Status) – статус сработавшего правила.
- **Активация свода правил** (Rulebook activation) – название активации свода правил.
- **Дата срабатывания** (Fired date) – дата и время срабатывания правила.

Просмотр

Для получения подробных сведений о правиле нажмите на ссылку с его названием в таблице. Окно сведений о правиле состоит из следующих вкладок:

- **Подробности** (Details) – общие сведения о правиле.
- **События** (Events) – события, вызвавшие срабатывание правила.

  Для подробного просмотра события и его журнала нажмите на ссылку в названии события.
- **Действия** (Actions) – действия, выполненные после срабатывания правила.

Активации сводов правил

Окно **Активации сводов правил** (Rulebook Activations) используется для управления процессами выполнения сводов правил в Event-Driven Automation с помощью следующих операций:

- запуск и остановка активации сводов правил;
- определение параметров запуска, источников событий и сред исполнения;
- контроль состояния активных процессов и их производительности.

Для перехода к окну **Активации сводов правил** (Rulebook Activations) выберите на панели навигации *Обработка событий* ▸ *Активация сводов правил* (*Automation Decision* ▸ *Rulebook Activations*).

Таблица активаций сводов правил

Внешний вид окна **Активации сводов правил** (Rulebook Activations) представлен на схеме:

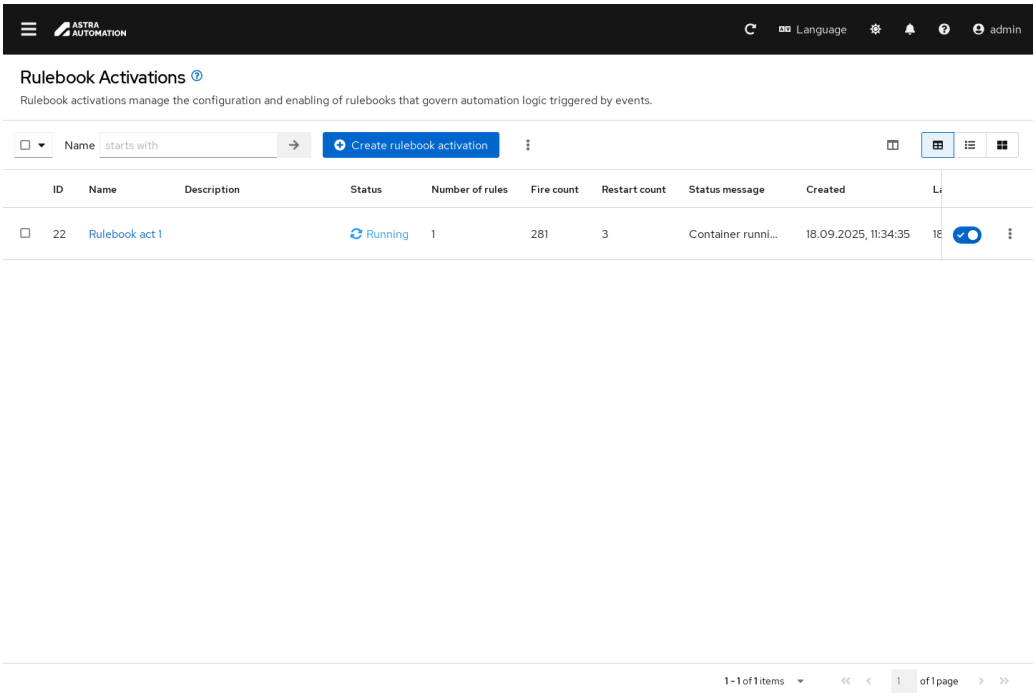


Таблица активаций сводов правил состоит из столбцов согласно настройке с помощью кнопки *Настроить столбцы* (Manage columns). Полный список столбцов:

- Переключатель подробности вывода основных сведений о своде правил.
- Флаги для выбора нескольких записей.
- **Идентификатор** (ID) – идентификатор активации свода правил.
- **Название** (Name) – ссылка для перехода в окно просмотра сведений об активации свода правил.
- **Описание** (Description) – описание активации свода правил.
- **Статус** (Status) – текущий статус активации свода правил.
- **Количество правил** (Number of rules) – количество правил в своде.
- **Количество запусков** (Fire count) – количество срабатываний правил.
- **Количество перезапусков** (Restart count) – количество перезапусков свода правил.
- **Сообщение о состоянии** (Status message) – статус выполнения активации свода правил.
- **Дата создания** (Created) – дата и время создания активации свода правил.
- **Последнее изменение** (Last modified) – дата и время последнего изменения активации свода правил.
- Переключатель активности свода правил.
- Список часто выполняемых действий:
  - **Удалить активацию свода правил** (Delete rulebook activation).

Просмотр

Для получения подробных сведений об активации свода правил нажмите ее название в таблице. Окно сведений об активации свода правил состоит из следующих вкладок:

- **Подробности** (Details) – общие сведения об активации свода правил.

- **История** (History) – история выполнения активации свода правил.
- **Командный доступ** (Team Access) – настройки доступа команд (групп) пользователей.
- **Доступ пользователей** (User Access) – настройки доступа отдельных пользователей.

Создание

Для создания активации свода правил выполните следующие действия:

1. На панели инструментов нажмите кнопку *Создать активацию свода правил* (Create rulebook activation).
2. Заполните форму **Создать активацию свода правил** (Create rulebook activation):
   - **Название** (Name) – название активации свода правил. Не допускается создание активаций с одним и тем же названием.
   - **Описание** (Description) – описание активации свода правил.
   - **Организация** (Organization) – организация, которой будет принадлежать активация свода правил.
   - **Проект** (Project) – проект с нужным сводом правил.
   - **Свод правил** (Rulebook) – свод правил, который необходимо активировать.
   - **Потоки событий** (Event streams) – webhooks для подключения источников событий к активациям сводов правил.
   - **Полномочие** (Credential) – полномочия для активации свода правил.
   - **Среда принятия решений** (Decision environment) – среда принятия решений, в которой будет работать свод правил.
   - **Политика перезапуска** (Restart policy) – политика, определяющая, как должна перезапускаться активация после завершения процесса контейнера, выполняющего код расширения (plugin) источника:
     - **Всегда** (Always) – всегда перезапускать активацию свода правил независимо от того, завершилась она успешно или нет.
     - **Никогда** (Never) – никогда не перезапускать активацию свода правил.
     - **При техническом отказе** (On failure) – перезапускать активацию свода правил только в случае сбоя процесса контейнера.
   - **Уровень логирования** (Log level) – события, которые будут записываться в журнал:
     - **Ошибка** (Error) – только ошибки.
     - **Информация** (Info) – информационные сообщения.
     - **Отладка** (Debug) – детальная отладочная информация.
   - **Название сервиса** (Service name) – название сервиса Kubernetes, к которому относиться событие.
   - **Активация свода правил включена?** (Rulebook activation enabled?) – автоматическое включение активации свода правил после создания.
   - **Переменные** (Variables) – переменные для свода правил в формате JSON или YAML.
   - **Пропустить события аудита** (Skip audit events) – события аудита не будут отображаться в разделе *Обработка событий ▸ Аудит правил* (*Automation Decision ▸ Rule Audit*).

Важно

Будьте внимательны при заполнении формы – изменить данные после создания активации свода правил невозможно.

3. Нажмите кнопку *Создать активацию свода правил* (Create rulebook activation).

Настройка доступа

Во вкладках **Доступ команд** (Team Access) и **Доступ пользователей** (User Access) выводится информация о ролях, назначенных командам и пользователям соответственно.

Назначение ролей командам

Чтобы настроить доступ к активации сводов правил для всех участников одной или нескольких команд, выполните следующие действия:

1. Во вкладке **Командный доступ** (Team Access) нажмите кнопку *Добавить роль* (Add roles).
2. Выберите команды, участникам которых хотите назначить роли, и нажмите кнопку *Далее* (Next).
3. Выберите роли, которые хотите назначить выбранным на предыдущем шаге командам.
4. Нажмите кнопку *Завершить* (Finish).

Назначение ролей отдельным пользователям

Чтобы настроить доступ к активации сводов правил для отдельных пользователей, выполните следующие действия:

1. Во вкладке **Доступ пользователей** (User Access) нажмите кнопку *Добавить роль* (Add roles).
2. Выберите пользователей, которым хотите назначить роли, и нажмите кнопку *Далее* (Next).
3. Выберите роли, которые хотите назначить выбранным на предыдущем шаге пользователям.
4. Нажмите кнопку *Завершить* (Finish).

Отзыв ролей

Чтобы отозвать пользовательские или командные роли, выполните следующие действия:

1. Во вкладке **Доступ пользователей** (User Access) или **Командный доступ** (Team Access) установите флаги в строках с пользователями или командами, соответственно, у которых следует отозвать роли.
2. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите пункт *Удалить роли* (Remove roles).
3. Подтвердите отзыв роли.

Перезапуск

Чтобы перезапустить активации сводов правил, выполните следующие действия:

1. В таблице активаций сводов правил установите флаги в строках с активациями, которые следует перезапустить.
2. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите пункт *Перезапустить активации сводов правил* (Restart rulebook activations).

3. Подтвердите перезапуск.

Удаление

Чтобы удалить активации сводов правил, выполните следующие действия:

1. В таблице активаций сводов правил установите флаги в строках с активациями, которые следует удалить.
2. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите пункт *Удалить активации сводов правил* (Delete rulebook activations).
3. Подтвердите удаление.

Проекты

Окно **Проекты** (Projects) служит для работы с репозиториями сводов правил с помощью следующих операций:

- подключение и синхронизация репозиториев со сводами правил;
- управление версиями и обновлениями контента, применяемого в активациях;
- поддержка связи с системами CI/CD и Git-репозиториями.

Для перехода к окну **Проекты** (Projects) выберите на панели навигации *Обработка событий* ▸ *Проекты* (*Automation Decision* ▸ *Projects*).

Таблица проектов

Внешний вид окна **Проекты** (Projects) представлен на схеме:

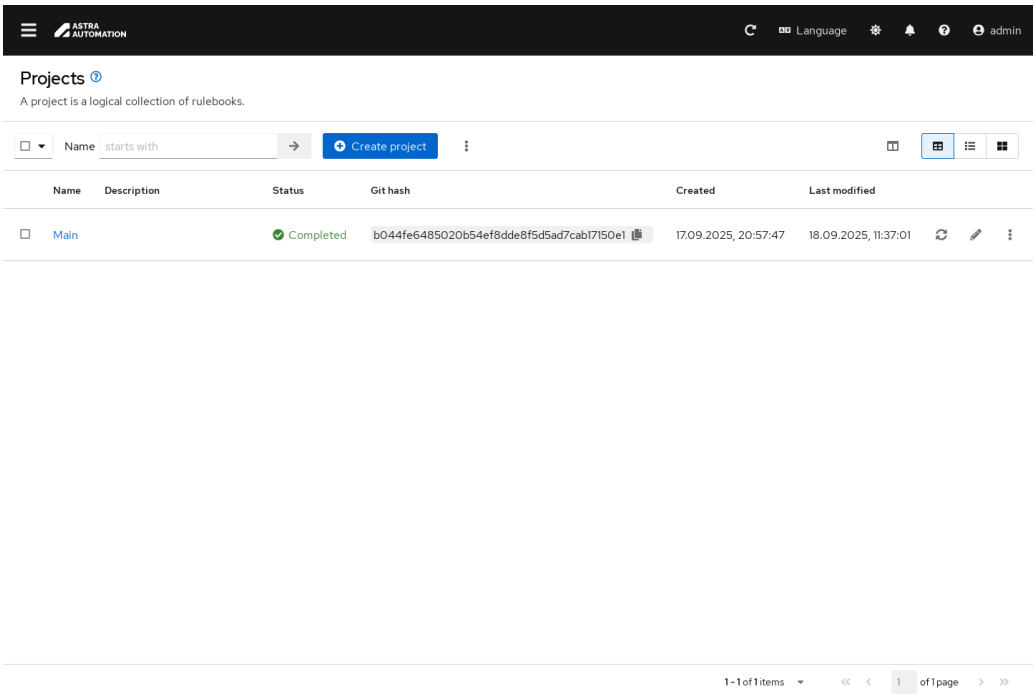


Таблица проектов состоит из столбцов согласно настройке с помощью кнопки *Настроить столбцы* (Manage columns). Полный список столбцов:

- Переключатель подробности вывода основных сведений о проекте.
- Флаги для выбора нескольких записей.
- **Название** (Name) – ссылка для перехода в окно просмотра сведений о проекте.
- **Описание** (Description) – описание проекта.
- **Статус** (Status) – текущий статус проекта.

- **Git-хеш** (Git hash) – хеш коммита, используемого проектом.
- **Дата создания** (Created) – дата создания проекта.
- **Последние изменение** (Last modified) – дата последнего изменения или синхронизации проекта.
- Кнопки для вызова часто выполняемых действий:
  - синхронизация проекта;
  - редактирование проекта;
  - удаление проекта.

Просмотр

Для получения подробных сведений о проекте нажмите на его название в окне проектов.

Окно сведений о проекте состоит из следующих вкладок:

- **Подробности** (Details) – общие сведения о проекте. Здесь же выводится информация об ошибках при синхронизации проекта.
- **Командный доступ** (Team Access) – настройки доступа команд (групп) пользователей.
- **Доступ пользователей** (User Access) – настройки доступа отдельных пользователей.

Создание

Для создания проекта выполните следующие действия:

1. На панели инструментов нажмите кнопку *Создать проект* (Create project).
2. Заполните форму **Создать проект** (Create project):
   - **Название** (Name) – название проекта.

     Особенности заполнения поля:
     - В рамках одной организации не допускается создание проектов с одним и тем же названием.
     - Название проекта не может состоять из одних пробельных символов (пробелы, табуляции и так далее).
     - Допускается использование символов кириллицы и специальных символов.
   - **Описание** (Description) – описание проекта, например, его назначение или особенности использования.
   - **Организация** (Organization) – организация, которой принадлежит проект.
   - **Тип системы управления исходным кодом** (Source control type) – значение этого поля нельзя изменить.
   - **URL системы управления исходным кодом** (Source control URL) – ссылка на репозиторий, в котором хранится код проекта.
   - **Прокси** (Proxy) – если для доступа к репозиторию с кодом проекта используется прокси-сервер, укажите параметры подключения к нему в этом поле.
   - **Ветвь/Тег/Коммит системы управления исходным кодом** (Source control branch/tag/commit) – если требуется использовать версию кода, отличную от последнего коммита в основной ветке, укажите в этом поле название необходимой ветки, название тега или хеш коммита.
   - **Специальная ссылка системы управления исходным кодом (refspec)** (Source control refspec) – если требуется использовать специфичную версию кода, укажите в этом поле правила получения необходимых ссылок из репозитория Git.

- **Полномочия на систему управления исходным кодом** (Source control credential) – если для доступа к репозиторию требуется авторизация, выберите в этом поле соответствующее полномочие типа *Система управления версиями (Source Control)*.
- **Полномочия для проверки подписи контента** (Content signature validation credential) – если код проекта подписан цифровой подписью, для ее проверки выберите в этом поле соответствующее полномочие типа *Открытый ключ GPG (GPG Public Key)*.
- **Проверка SSL** (Verify SSL) – если этот флаг включен, при подключении к репозиторию с кодом проекта выполняется строгая проверка сертификата.

3. Нажмите кнопку *Создать проект* (Create project).

Синхронизация

Чтобы обновить код проекта, в строке с необходимой записью нажмите кнопку *Синхронизировать проект* (Sync project).

Настройка доступа

Во вкладках **Командный доступ** (Team Access) и **Доступ пользователей** (User Access) выводится информация о ролях, назначенных командам и пользователям соответственно.

Назначение ролей командам

Чтобы настроить доступ к проекту для всех участников одной или нескольких команд, выполните следующие действия:

1. Во вкладке **Командный доступ** (Team access) нажмите кнопку *Добавить роль* (Add roles).
2. В открывшемся окне выберите команды, которым хотите назначить роли, и нажмите кнопку *Далее* (Next).
3. Выберите роли, которые хотите назначить команде, и нажмите кнопку *Далее* (Next).
4. Проверьте корректность введенных данных и нажмите кнопку *Завершить* (Finish).

Назначение ролей отдельным пользователям

Чтобы настроить доступ к проекту для одного или нескольких пользователей, выполните следующие действия:

1. Во вкладке **Доступ пользователей** (User access) нажмите кнопку *Добавить роль* (Add roles).
2. В открывшемся окне выберите пользователей, которым хотите назначить роли, и нажмите кнопку *Далее* (Next).
3. Выберите роли, которые хотите назначить пользователям, и нажмите кнопку *Далее* (Next).
4. Проверьте корректность введенных данных и нажмите кнопку *Завершить* (Finish).

Отзыв ролей

Чтобы отозвать пользовательские или командные роли, выполните следующие действия:

1. Во вкладке **Доступ пользователей** (User Access) или **Командный доступ** (Team Access) установите флаги в строках с пользователями или командами, соответственно, у которых следует отозвать роли.
2. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите пункт *Удалить роли* (Remove roles).
3. Подтвердите отзыв роли.

Удаление

Предупреждение

При удалении проекта удаляются все связанные с ним своды правил.

Чтобы удалить проекты, выполните следующие действия:

1. На панели проектов включите флаги на записях, подлежащих удалению.
2. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите **Удалить проекты** (Delete projects).
3. Подтвердите удаление проектов.

Среды принятия решений

Окно **Среды принятия решений** (Decision Environments) позволяет управлять контейнеризованными окружениями, в которых выполняются своды правил Event-Driven Automation, с помощью следующих операций:

- просмотр доступных сред и их версии;
- настройка контейнерных образов с нужными библиотеками и зависимостями;
- обеспечение совместимости сводов правил с корпоративными политиками безопасности.

Для перехода к окну **Среды принятия решений** (Decision Environments) выберите на панели навигации *Обработка событий* ▸ *Среды принятия решений* (*Automation Decision* ▸ *Decision Environments*).

Таблица сред принятия решений

Внешний вид окна **Среды принятия решений** (Decision Environments) представлен на схеме:

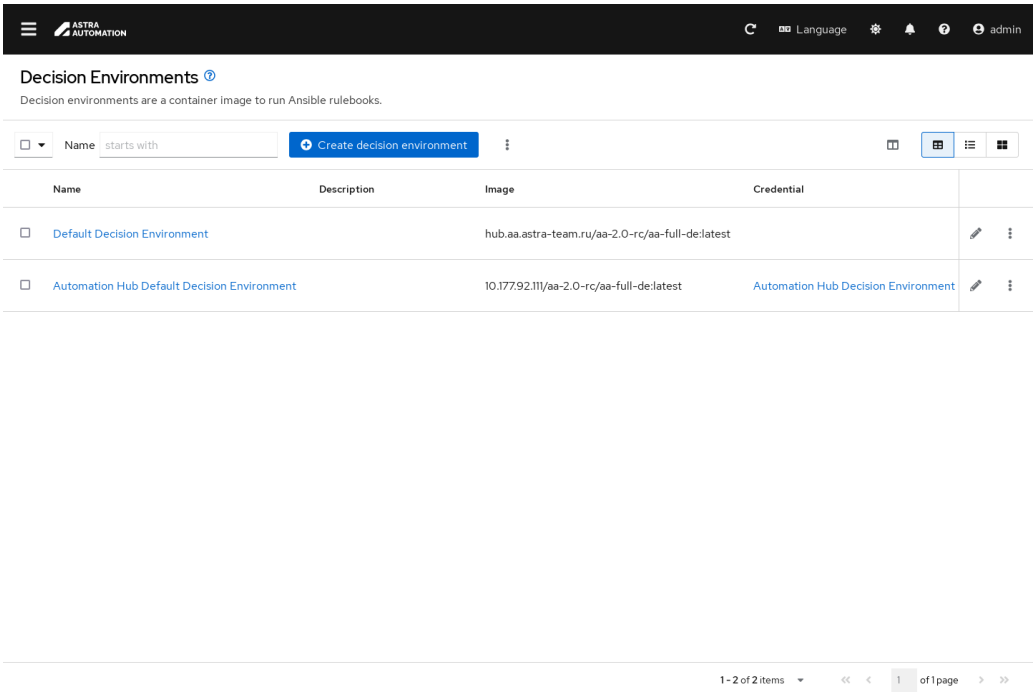


Таблица сред принятия решений состоит из столбцов согласно настройке с помощью кнопки *Настроить столбцы* (Manage columns). Полный список столбцов:

- Переключатель подробности вывода основных сведений о среде принятия решений.

- Флаги для выбора нескольких записей.
- **Название** (Name) – ссылка для перехода в окно просмотра сведений о среде принятия решений.
- **Описание** (Description) - дополнительные сведения о среде принятия решений.
- **Образ** (Image) – название образа из реестра образов.
- **Полномочие** (Credential) – ссылка на полномочие, используемое для авторизации в реестре образов.
- **Дата создания** (Created) – дата создания среды принятия решений.
- **Последнее изменение** (Last modified) – дата последнего изменения среды принятия решений.
- Кнопки для вызова часто выполняемых действий:
  - редактирование среды принятия решений;
  - удаление среды принятия решений.

Просмотр

Для получения подробных сведений о среде принятия решений нажмите на ее название в окне сред принятия решений.

Окно сведений о среде принятия решений состоит из следующих вкладок:

- **Подробности** (Details) – общие сведения о среде принятия решений.
- **Командный доступ** (Team Access) – список команд (групп) и назначенных им ролей на доступ к среде принятия решений.
- **Доступ пользователей** (User Access) – список пользователей и назначенных им ролей на доступ к среде принятия решений.

Добавление

Для добавления среды принятия решений выполните следующие действия:

1. В окне **Среды принятия решений** (Decision environments) нажмите кнопку *Создать среду принятия решений* (Create decision environment).
2. Заполните форму **Создать среду принятия решений** (Create decision environment):
   - **Название** (Name) – название среды принятия решений.

     Примечание

     Название должно быть уникальным.

   - **Описание** (Description) – описание среды принятия решений.
   - **Организация** (Organization) – выберите организацию, которой принадлежит среда принятия решений.

     Если это поле не заполнено, среде принятия решений автоматически будет назначена организация по умолчанию.
   - **Образ** (Image) – ссылка на образ в реестре образов.
   - **Полномочие** (Credential) – если для доступа к реестру образов требуется аутентификация, выберите в этом поле полномочие типа «Реестр контейнеров» (Container Registry).
3. Нажмите кнопку *Создать среду принятия решений* (Create decision environment).

Настройка доступа

Во вкладках **Командный доступ** (Team Access) и **Доступ пользователей** (User Access) выводится информация о ролях, назначенных командам и пользователям соответственно.

Назначение ролей командам

Чтобы настроить доступ к среде принятия решений для всех участников одной или нескольких команд, выполните следующие действия:

1. Во вкладке **Командный доступ** (Team access) нажмите кнопку *Добавить роль* (Add roles).
2. В открывшемся окне выберите команды, которым хотите назначить роли, и нажмите кнопку *Далее* (Next).
3. Выберите роли, которые хотите назначить команде, и нажмите кнопку *Далее* (Next).
4. Проверьте корректность введенных данных и нажмите кнопку *Завершить* (Finish).

Назначение ролей отдельным пользователям

Чтобы настроить доступ к среде принятия решений для одного или нескольких пользователей, выполните следующие действия:

1. Во вкладке **Доступ пользователей** (User access) нажмите кнопку *Добавить роль* (Add roles).
2. В открывшемся окне выберите пользователей, которым хотите назначить роли, и нажмите кнопку *Далее* (Next).
3. Выберите роли, которые хотите назначить пользователям, и нажмите кнопку *Далее* (Next).
4. Проверьте корректность введенных данных и нажмите кнопку *Завершить* (Finish).

Отзыв ролей

Чтобы отозвать пользовательские или командные роли, выполните следующие действия:

1. Во вкладке **Доступ пользователей** (User Access) или **Командный доступ** (Team Access) установите флаги в строках с пользователями или командами, соответственно, у которых следует отозвать роли.
2. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите пункт *Удалить роли* (Remove roles).
3. Подтвердите отзыв роли.

Удаление

Чтобы удалить среды принятия решений, выполните следующие действия:

1. В окне сред принятия решений включите флаги на записях, подлежащих удалению.
2. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите **Удалить среды принятия решений** (Delete decision environments).
3. Подтвердите удаление сред принятия решений.

Потоки событий

Окно **Потоки событий** (Event Streams) предоставляет интерфейс для управления серверными веб-перехватчиками и интеграциями Event-Driven Automation с внешними источниками событий с помощью следующих операций:

- создание и настройка веб-перехватчиков для приема событий;
- управление обработкой входящих событий и их маршрутизацией к правилам;

- обеспечение изоляции и масштабирования потоков событий для разных бизнес-процессов.

Для перехода к окну **Потоки событий** (Event Streams) выберите на панели навигации *Обработка событий* ▸ *Потоки событий* (*Automation Decision* ▸ *Event Stream*).

Таблица потоков событий

Внешний вид окна **Потоки событий** (Event Streams) представлен на схеме:

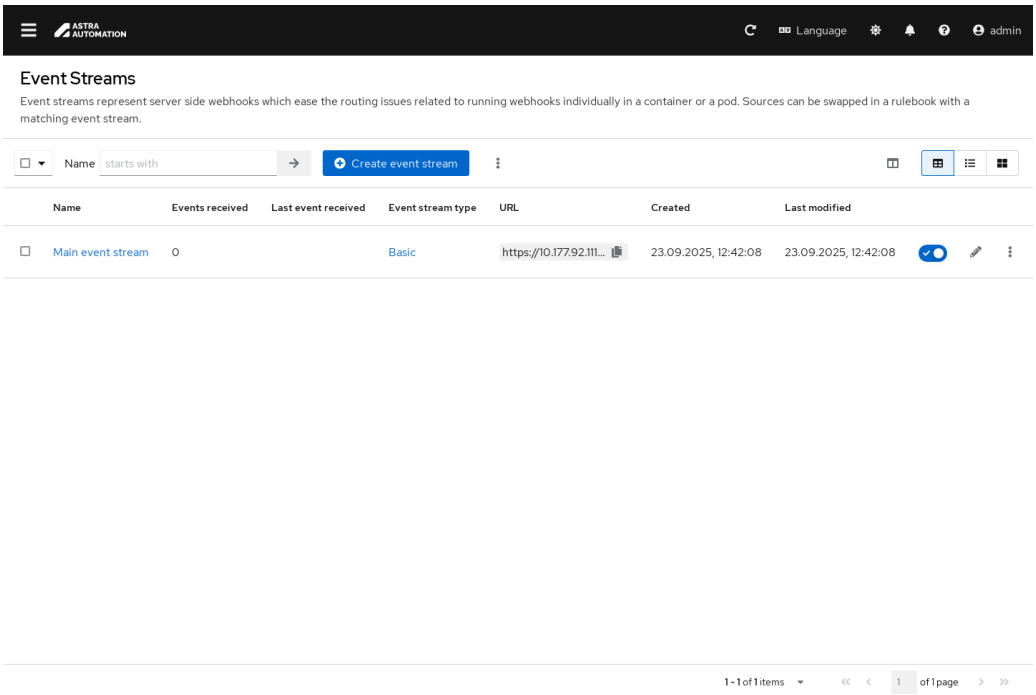


Таблица потоков событий состоит из столбцов согласно настройке с помощью кнопки *Настроить столбцы* (Manage columns). Полный список столбцов:

- Переключатель подробности вывода основных сведений о потоке событий.
- Флаги для выбора нескольких записей.
- **Название** (Name) – ссылка для перехода в окно просмотра сведений о потоке событий.
- **Полученные события** (Events received) – общее количество событий, полученных потоком событий.
- **Последнее полученное событие** (Last event received) – последнее событие, полученное поток событий.
- **Тип потока событий** (Event stream type) – тип потока событий.
- **URL** (URL) – ссылка на поток событий для удаленной системы, отправляющей события.
- **Дата создания** (Created) – дата создания потока событий.
- **Последние изменение** (Last modified) – дата последнего изменения потока событий.
- Кнопки для вызова часто выполняемых действий:
  - переключатель пересылки событий в активацию свода правил;
  - редактирование потока событий;
  - удаление потока событий.

Просмотр

Для получения подробных сведений о потоке событий нажмите на его название в окне потоков событий. Окно сведений о потоке событий состоит из следующих вкладок:

- **Подробности** (Details) – общие сведения о потоке событий.
- **Активации** (Activations) – история срабатываний потока событий.
- **Командный доступ** (Team Access) – настройки доступа команд (групп) пользователей.
- **Доступ пользователей** (User Access) – настройки доступа отдельных пользователей.

Создание

Для создания потока событий выполните следующие действия:

1. На панели инструментов нажмите кнопку *Создать поток событий* (Create event stream).
2. Заполните форму **Создать поток событий** (Create event stream):
   - **Название** (Name) – название потока событий.

     Особенности заполнения поля:

     - В рамках одной организации не допускается создание потоков событий с одним и тем же названием.
     - Название потока событий не может состоять из одних пробельных символов (пробелы, табуляции и так далее).
     - Допускается использование символов кириллицы и специальных символов.
   - **Организация** (Organization) – организация, которой принадлежит поток событий.

     Если это поле не заполнено, среде принятия решений автоматически будет назначена организация по умолчанию.
   - **Тип потока событий** (Event stream type) – тип потока событий.
   - **Полномочие** (Credential) – полномочие для выполнения потока событий.
   - **Заголовки** (Headers) – HTTP-заголовки, которые необходимо включить в полезную нагрузку события.

     Чтобы включить все заголовки, оставьте поле пустым.
   - **Перенаправлять события на активизацию rulebook** (Forward events to rulebook activation) – перенаправление событий для активации сводов правил.
3. Нажмите кнопку *Создать поток событий* (Create event stream).

Настройка доступа

Во вкладках **Командный доступ** (Team Access) и **Доступ пользователей** (User Access) выводится информация о ролях, назначенных командам и пользователям соответственно.

Назначение ролей командам

Чтобы настроить доступ к потоку событий для всех участников одной или нескольких команд, выполните следующие действия:

1. Во вкладке **Командный доступ** (Team access) нажмите кнопку *Добавить роль* (Add roles).
2. В открывшемся окне выберите команды, которым хотите назначить роли, и нажмите кнопку *Далее* (Next).
3. Выберите роли, которые хотите назначить команде, и нажмите кнопку *Далее* (Next).

4. Проверьте корректность введенных данных и нажмите кнопку *Завершить* (Finish).

Назначение ролей отдельным пользователям

Чтобы настроить доступ к потоку событий для одного или нескольких пользователей, выполните следующие действия:

1. Во вкладке **Доступ пользователей** (User access) нажмите кнопку *Добавить роль* (Add roles).
2. В открывшемся окне выберите пользователей, которым хотите назначить роли, и нажмите кнопку *Далее* (Next).
3. Выберите роли, которые хотите назначить пользователям, и нажмите кнопку *Далее* (Next).
4. Проверьте корректность введенных данных и нажмите кнопку *Завершить* (Finish).

Отзыв ролей

Чтобы отозвать пользовательские или командные роли, выполните следующие действия:

1. Во вкладке **Доступ пользователей** (User Access) или **Командный доступ** (Team Access) установите флаги в строках с пользователями или командами, соответственно, у которых следует отозвать роли.
2. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите пункт *Удалить роли* (Remove roles).
3. Подтвердите отзыв роли.

Удаление

Чтобы удалить потоки событий, выполните следующие действия:

1. В окне поток событий включите флаги на записях, подлежащих удалению.
2. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите **Удалить потоки событий** (Delete selected event streams).
3. Подтвердите удаление потоков событий.

Инфраструктура

Раздел **Инфраструктура** (Infrastructure) содержит подразделы, используемые для управления полномочиями и их типами.

Полномочия

Окно **Полномочия** (Credentials) предоставляет централизованные механизмы управления учетными данными для Event-Driven Automation с помощью следующих операций:

- создание и хранение учетных данных для доступа к внешним системам и брокерам событий;
- назначение и контроль доступа к секретам согласно политике *RBAC*;
- обеспечение безопасного использования полномочий автоматизированными процессами.

Для перехода к окну **Полномочия** (Credentials) выберите на панели навигации *Обработка событий* ▸ *Инфраструктура* ▸ *Полномочия* (*Automation Decision* ▸ *Infrastructure* ▸ *Credentials*).

Таблица полномочий

Внешний вид окна **Полномочия** (Credentials) представлен на схеме:

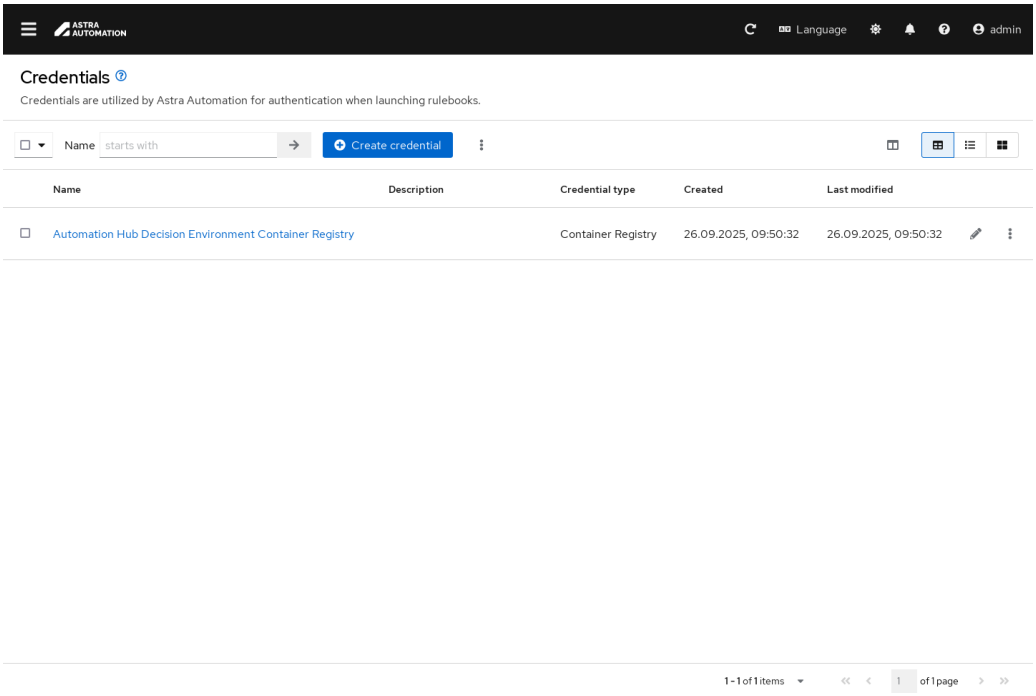


Таблица полномочий состоит из столбцов согласно настройке с помощью кнопки *Настроить столбцы* (Manage columns). Полный список столбцов:

- Флаги для выбора нескольких записей.
- **Название** (Name) – название полномочия.
- **Описание** (Description) – дополнительная информация о полномочии, например, цель его создания или правила использования.
- **Тип полномочий** (Credential type) – тип полномочий.
- **Дата создания** (Created) – дата и время создания полномочия.
- **Последнее изменение** (Last modified) – дата и время последнего изменения полномочия.
- Кнопки для вызова часто выполняемых действий:
  - редактирование полномочия;
  - удаление полномочия.

Просмотр сведений

Для получения подробных сведений о полномочии нажмите на его название в таблице полномочий.

Окно со сведениями о полномочии содержит следующие вкладки:

- **Подробности** (Details) – общие сведения о полномочии;
- **Командный доступ** (Team Access) – таблица команд и назначенных им *ролей*;
- **Доступ пользователей** (User Access) – таблица пользователей и назначенных им *ролей*.

Создание

Для создания полномочия выполните следующие действия:

1. В окне **Полномочия** (Credentials) нажмите кнопку *Создать полномочие* (Create credential).
2. Заполните форму **Создать полномочие** (Create credential):
   - **Название** (Name) – уникальное название полномочия.
   - **Описание** (Description) – дополнительная информация о полномочии, например, цель его создания или правила использования.
   - **Организация** (Organization) – организация, которой принадлежит полномочие.

     Если это поле не заполнено, полномочию автоматически будет назначена организация по умолчанию.
   - **Тип полномочий** (Credential type) – тип полномочий.

     При выборе значения в этом поле в форме появляются соответствующие поля.
3. Заполните поля, соответствующие выбранному типу полномочия.

   Особенности заполнения полей для каждого типа полномочия приведены в следующих секциях.
4. Нажмите кнопку *Создать полномочие* (Create credential).

Astra Ansible Automation Platform

При создании полномочия этого типа доступны следующие поля:

- **Astra Ansible Automation Platform** – URL платформы для аутентификации.
- **Название учетной записи** (Username) – название учетной записи пользователя для аутентификации.

  Примечание

  Не заполняется, если используется OAuth-токен.

- **Пароль** (Password) – пароль пользователя для аутентификации.
- **Токен OAuth** (OAuth Token) – токен для аутентификации.

  Примечание

  Не заполняется, если используется аутентификация по паролю.

- **Проверка SSL** (Verify SSL) – если этот флаг включен, при подключении выполняется проверка SSL-сертификата сервера.
- **Request Timeout** – время ожидания, которое Ansible должен использовать при запросах к узлу.

Basic Event Stream

При создании полномочия этого типа доступны следующие поля:

- **Название учетной записи** (Username) – название учетной записи пользователя для аутентификации;
- **Пароль** (Password) – пароль пользователя для аутентификации.

Container Registry

При создании полномочия этого типа доступны следующие поля:

- **URL аутентификации** (Authentication URL) – URL-адрес реестра контейнеров;
- **Название учетной записи** (Username) – название учетной записи для аутентификации;
- **Пароль или токен** (Password or Token) – ключ пользователя для аутентификации (может быть паролем или токеном API);
- **Проверка SSL** (Verify SSL) – если этот флаг включен, при подключении выполняется проверка SSL-сертификата сервера.

Dynatrace Event Stream

При создании полномочия этого типа доступны следующие поля:

- **Название учетной записи** (Username) – название учетной записи пользователя для аутентификации;
- **Пароль** (Password) – пароль пользователя для аутентификации.

ECDSA Event Stream

При создании полномочия этого типа доступны следующие поля:

- **Primary Header Key** – HTTP-заголовок для передачи подписи.
- **Public Key** – открытый ключ для проверки данных.

  Открытый ключ будет предоставлен отправителем после того, как вы создадите поток событий на их стороне с помощью URL, полученном при создании потока событий в Astra Automation.
- **Additional Prefix Header Key** – дополнительный ключ заголовка для ECDSA.
- **Signature Encoding** – тип строки, в который должна быть преобразована сигнатура полезной нагрузки перед добавлением в HTTP-заголовок.
- **Hash algorithm** – алгоритм, с помощью которого отправитель потока событий хеширует сообщение.

GitHub Event Stream

При создании полномочия этого типа укажите в поле **HMAC Secret** симметричный общий секретный ключ между Event-Driven Automation и сервером передачи событий.

Рекомендуется сохранить указанное значение, так как оно понадобится на передачи событий.

GitLab Event Stream

При создании полномочия этого типа укажите в поле **Токен** (Token) симметричный общий секретный ключ между Event-Driven Automation и сервером GitLab.

Рекомендуется сохранить указанное значение, так как оно понадобится на сервере передачи событий.

GPG Public Key

При создании полномочия этого типа укажите в поле **Открытый ключ GPG** (GPG Public Key) содержимое ключа GPG, который контроллер EDA должен использовать для проверки подписи содержимого.

HMAC Event Stream

При создании полномочия этого типа доступны следующие поля:

- **Secret** – симметричный общий секретный ключ между Event-Driven Automation и сервером передачи событий.

  Рекомендуется сохранить указанное значение, так как оно понадобится на сервере передачи событий.
- **Hash algorithm** – алгоритм, с помощью которого отправитель потока событий хеширует сообщение.
- **HMAC Header Key** – название HTTP-заголовка, в котором отправитель потока событий передает подпись полезной нагрузки.
- **Signature Encoding** – формат, в котором должна быть закодирована подпись полезной нагрузки перед добавлением в HTTP-заголовок.
- **Signature Prefix** – необязательный префикс, который отправитель может добавлять к значению подписи.

OAuth2 Event Stream

При создании полномочия этого типа доступны следующие поля:

- **ID клиента** (Client ID) – идентификатор клиента на сервере авторизации.
- **Client Secret** – секретный ключ клиента на сервере авторизации.
- **Introspection URL** – URL интроспекции на сервере авторизации.

  Подробности см. в **RFC 7662**[200].

OAuth2 JWT Event Stream

При создании полномочия этого типа доступны следующие поля:

- **JWKS URL** – URL-адрес набора веб-ключей JSON для получения открытых ключей для проверки токена JWT, например:

  ```
  https://<auth_server>/.well-known/jwks.json
  ```
- **Audience** – идентификатор получателя токена. Если поле заполнено, то контроллер будет проверять его совпадение со значением audience (ключ aud) в поле запроса токена JWT.

Postgres

Важно

Эта часть документации находится в стадии разработки.

ServiceNow Event Stream

При создании полномочия этого типа укажите в поле **Токен** (Token) симметричный общий секретный ключ между Event-Driven Automation и сервером ServiceNow.

Рекомендуется сохранить указанное значение, так как оно понадобится на сервере передачи событий.

[200] https://datatracker.ietf.org/doc/html/rfc7662.html

Source Control

При создании полномочия этого типа доступны следующие поля:

- **Название учетной записи** (Username) – название учетной записи пользователя для аутентификации;
- **Пароль** (Password) – пароль пользователя для аутентификации;
- **Приватный ключ SCM** (SCM Private Key) – содержимое приватного ключа SSH;
- **Пароль от приватного ключа** (Private Key Passphrase) – если приватный ключ SSH защищен паролем, укажите его в этом поле.

Token Event Stream

При создании полномочия этого типа доступны следующие поля:

- **Токен** (Token) – симметричный общий секретный ключ между Event-Driven Automation и сервером передачи событий.

  Рекомендуется сохранить указанное значение, так как оно понадобится на сервере передачи событий.
- **HTTP Header Key** – название заголовка HTTP, в котором отправитель потока событий передает токен.

Vault

При создании полномочия этого типа доступны следующие поля:

- **Пароль Vault** (Vault Password) – пароль для доступа к хранилищу Vault.
- **Идентификатор Vault** (Vault Identitier) – идентификатор хранилища. Заполнение данного поля эквивалентно указанию параметра `--vault-id` при работе с несколькими паролями Vault.

  Примечание

  Для корректной работы этого полномочия версия Ansible Core в среде принятия решений должна быть не ниже 2.4.

Редактирование

Для редактирования полномочия выполните следующие действия:

1. Нажмите на название полномочия в таблице полномочий.
2. Нажмите кнопку *Редактировать полномочия* (Edit credential).
3. Отредактируйте необходимые поля.

   Для редактирования доступны все поля, кроме **Тип полномочий** (Type credential).
4. Нажмите кнопку *Сохранение полномочия* (Save credential).

Удаление

Для удаления полномочий выполните следующие действия:

1. В окне полномочий установите флаги в строках с полномочиями, которые необходимо удалить.
2. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите пункт *Удалить полномочия* (Delete credentials).
3. Подтвердите удаление.

Типы полномочий

Окно **Типы полномочий** (Credentials types) используется для просмотра встроенных и собственных типов полномочий, создания собственных типов полномочий и управления ими. Для перехода в окно выберите на панели навигации *Обработка событий* ▸ *Инфраструктура* ▸ *Типы полномочий* (*Automation Decision* ▸ *Infrastructure* ▸ *Credential Types*).

Таблица типов полномочий

Внешний вид окна **Типы полномочий** (Credentials types) представлен на схеме:

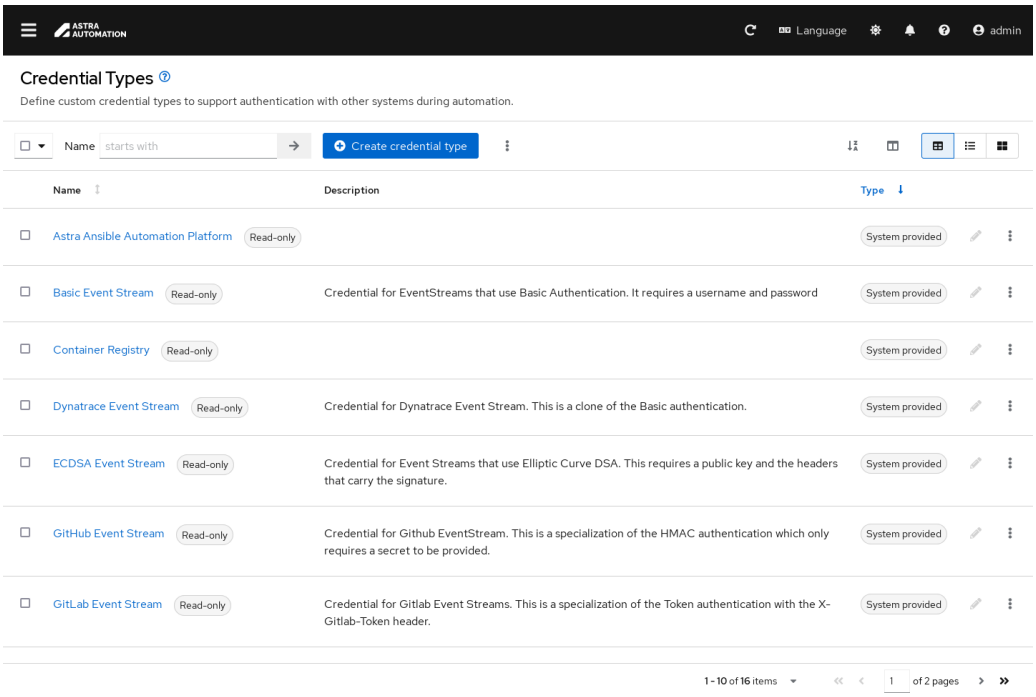


Таблица типов полномочий состоит из следующих колонок:

- Флаги для выбора нескольких записей.
- **Название** (Name) – название типа полномочий.

  Рядом с названиями встроенных типов полномочий выводится метка **Доступен только для чтения** (Read-only).
- **Описание** (Description) – дополнительная информация о типе полномочий, например, цель его создания или правила использования.
- **Тип** (Type) – создатель полномочия.

  Возможные значения:

  - **System provided** – системный тип полномочий, недоступен для редактирования.
  - Пустое значение – пользовательский тип полномочий.
- Кнопки для вызова часто выполняемых действий:
  - редактирование типа полномочий;
  - удаление типа полномочий.

Примечание

Встроенные типы полномочий нельзя изменить или удалить.

Просмотр

Для получения подробной информации о типе полномочий нажмите на ссылку с его названием в таблице типов полномочий.

Окно просмотра сведений о типе полномочий состоит из следующих вкладок:

- **Подробности** (Details) – основные сведения о типе полномочий: название, организация-владелец, конфигурации входных данных и внедрения данных.
- **Полномочия** (Credentials) – таблица связанных полномочий.

Создание

Для создания типа полномочий выполните следующие действия:

1. В окне **Типы полномочий** (Credential types) нажмите кнопку *Создать тип полномочий* (Create credential type).
2. Заполните форму **Создать тип полномочий** (Create credential type):
   - **Название** (Name) – уникальное название типа полномочий.
   - **Описание** (Description) – краткое описание типа полномочий, например, особенности его использования.
   - **Входная настройка** (Input configuration) – параметры входных данных, которые необходимо указать при использовании полномочий этого типа.
   - **Конфигурация внедрения данных** (Injector configuration) – данные, которые передаются во внешнюю систему аутентификации при использовании полномочий этого типа.

   Примечание

   Особенности заполнения полей **Входная настройка** (Input configuration) и **Конфигурация внедрения данных** (Injector configuration) см. в секции *Собственные типы полномочий*.

3. Нажмите кнопку *Создать тип полномочий* (Create credential type).

Удаление

Предупреждение

При удалении типа полномочия удаляются все связанные с ним полномочия.

Чтобы удалить типы полномочий, выполните следующие действия:

1. На панели типов полномочий включите флаги на записях, подлежащих удалению.
2. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите **Удалить типы полномочий** (Delete credential types).
3. Подтвердите удаление типов полномочий.

12.5 API

Event-Driven Automation API представляет собой RESTful API, предоставляющий программный доступ ко всем функциям системы автоматизации на основе обработки событий в Astra Automation. API позволяет организациям с помощью приложений управлять автоматизацией на основе событий путем глубокой интеграции с внешними системами мониторинга, которые служат источниками событий. Он предоставляет различные методы авторизации,

включая session authentication через platform gateway и token-based authentication, обеспечивая безопасный доступ для инструментов командной строки и внешних систем для управления rulebook activations, event streams и credentials.

Доступ к API осуществляется через множество точек доступа (endpoints), начиная с базового URI `/api/eda/v1/` контроллера EDA, и включает полную поддержку методов GET, POST, PUT, PATCH и DELETE для управления ресурсами, такими как projects, decision environments, rulebook activations, credentials и event streams.

Примечание

Подробное формальное описание HTTP API представлено в *спецификации*.

12.5.1 Спецификация API

12.6 Справочные данные

Для получения и обработки событий из различных источников необходима подробная информация. В частности, как настроить своды правил для приема и обработки данных от этих источников. Этому помогут следующие подробные формальные описания:

- сведения о расширениях из состава коллекции `ansible.eda`, которые необходимы для использования различных источников событий;
- сведения о действиях, которые может выполнять контроллер Event-Driven Automation при наступлении событий.

12.6.1 Источники

В состав коллекции `ansible.eda` входят следующие расширения, позволяющие работать с источниками событий:

- *alertmanager*;
- *azure_service_bus*;
- *file*;
- *file_watch*;
- *kafka*;
- *range*;
- *tick*;
- *url_check*;
- *webhook*.

alertmanager

Источник `alertmanager` позволяет обращаться к точкам доступа API системы мониторинга Prometheus Alertmanager[201].

Источник предоставляет следующие настройки:

- `host` – IP-адрес или URL веб-обработчика (webhook) Alertmanager.

  Значение по умолчанию: `localhost`.

- `port` – номер порта веб-обработчика Alertmanager.

  Значение по умолчанию: `5000`.

[201] https://prometheus.io/docs/alerting/latest/alertmanager/

- `data_alerts_path` – выражение в формате JSONPath[202] для поиска данных для оповещения.

  Значение по умолчанию: `alerts`.

- `data_host_path` – JSONPath внутри данных оповещения для поиска узла, являющегося источником данных.

  Значение по умолчанию: `labels.instance`.

- `data_path_separator` – разделитель, используемый при разборе `data_alerts_path` и `data_host_path`.

  Значение по умолчанию: `.`.

  Пример использования `data_path_separator`

  Допустим, Alertmanager отправляет следующий JSON:

```
{
  "alerts": [
    {
      "labels": {
        "instance": "server1.example.com",
        "severity": "critical"
      },
      "annotations": {
        "summary": "Disk space is low"
      }
    }
  ]
}
```

  Если `data_alerts_path = "alerts"` и `data_host_path = "labels.instance"`, а `data_path_separator = "."`, то система извлечет узел `server1.example.com` из пути `alerts[0].labels.instance`. Если бы `data_path_separator` был, например, `/`, то тот же путь записывался бы как `alerts/0/labels/instance`.

  В `data_path_separator` можно задать любой символ, который будет использоваться для разделения путей внутри JSON:

```
data_alerts_path: "alerts"
data_host_path: "labels/instance"
data_path_separator: "/"
```

- `skip_original_data` – управляет добавлением исходных данных в очередь.

  Возможные значения:

  - `true` – в очередь добавляются только данные оповещения.
  - `false` – в очередь последовательно добавляются как полученные исходные данные, так и каждый разобранный элемент оповещения.

  Значение по умолчанию: `false`.

azure_service_bus

Источник `azure_service_bus` позволяет получать события из службы Azure Service Bus[203].

Источник предоставляет следующие настройки:

- `conn_str` – строка подключения к Azure Service Bus.

  Обязательный параметр.

[202] https://en.wikipedia.org/wiki/JSONPath
[203] https://azure.microsoft.com/en-us/products/service-bus

- queue_name – название очереди сообщений.

  Обязательный параметр.

- logging_enable – запись в журналы.
  - true – включено;
  - false – включено.

  Значение по умолчанию: true.

file

Источник file загружает факты Ansible из файлов YAML при запуске и при изменении содержимого файлов. Список файлов укажите в настройке files.

file_watch

Источник file_watch отслеживает изменения в файловой системе.

Источник предоставляет следующие настройки:

- path – каталог для наблюдения за изменениями.

  Обязательный параметр.

- ignore_regexes – список регулярных выражений для игнорирования изменений. Файлы и каталоги, имена которых удовлетворяют указанному выражению, не отслеживаются.
- recursive – отслеживание изменений не только в указанном каталоге, но и в его подкаталогах.

  Обязательный параметр.

kafka

Источник kafka позволяет получать события из брокера Kafka.

Источник предоставляет следующие настройки:

- host – IP-адрес сервера Kafka.

  Обязательный параметр.

- port – порт сервера Kafka.

  Обязательный параметр.

- cafile – путь к файлу сертификата, используемого для подписи сертификатов брокеров Kafka.
- certfile – путь к файлу клиентского сертификата, содержащему сертификат клиента и сертификаты CA, необходимые для подтверждения подлинности.
- keyfile – путь к файлу приватного ключа клиента.
- password – пароль для загрузки цепочки сертификатов.
- check_hostname – включение проверки имени узла для SSL.

  Значение по умолчанию: true.

- verify_mode – режим проверки сертификатов других участников.

  Возможные значения:

  - CERT_NONE – проверка сертификатов отключена. Соединение устанавливается без проверки подлинности сертификата сервера.
  - CERT_OPTIONAL – если сервер предоставляет сертификат, то он будет проверен. Если сертификат недействителен или отсутствует, соединение все равно будет установлено.

  - CERT_REQUIRED – сертификат сервера должен быть предоставлен и проверен. Если сертификат отсутствует или недействителен, соединение разрывается.

  Значение по умолчанию: CERT_REQUIRED.

- encoding — кодировка сообщений.

  Значение по умолчанию: utf-8.

- topic – название топика.

- group_id – идентификатор группы Kafka.

  Значение по умолчанию: null.

- offset – начальная позиция чтения.

  Возможные значения:

  - latest – контроллер EDA начнет чтение с последнего сообщения.
  - earliest – контроллер EDA начнет чтение с самого раннего сообщения.

  Значение по умолчанию: latest.

- security_protocol – протокол безопасности для связи с брокерами.

  Возможные значения:

  - PLAINTEXT – незашифрованный протокол для связи с Kafka-брокерами, не обеспечивающий аутентификацию или шифрование данных.
  - SSL – протокол с шифрованием данных с использованием SSL/TLS, обеспечивающий конфиденциальность.
  - SASL_PLAINTEXT – протокол с аутентификацией через SASL, но без шифрования данных.
  - SASL_SSL – протокол с аутентификацией через SASL и шифрованием данных через SSL/TLS, обеспечивающий максимальную безопасность соединения.

  Значение по умолчанию: PLAINTEXT.

- sasl_mechanism – механизм аутентификации при настройке security_protocol.

  Возможные значения:

  - PLAIN – название учетной записи и пароль передаются в виде обычного текста.
  - GSSAPI – использование Kerberos для централизованного управления доступом.
  - SCRAM-SHA-256 – для защиты паролей используется хеширование по алгоритму SHA-256.
  - SCRAM-SHA-512 – для защиты паролей используется хеширование по алгоритму SHA-512.
  - OAUTHBEARER – использование OAuth 2.0 и токенов доступа для интеграции с внешними системами.

  Значение по умолчанию: PLAIN.

- sasl_plain_username – название учетной записи для аутентификации SASL PLAIN.
- sasl_plain_password – пароль для аутентификации SASL PLAIN.

range

Источник range генерирует события с увеличивающимся индексом.

Источник предоставляет единственную настройку limit, в которой необходимо указать верхний предел диапазона индекса.

Значение по умолчанию: 0.

tick

Источник `tick` генерирует события с увеличивающимся индексом без завершения.

Источник предоставляет единственную настройку `delay`, в которой необходимо указать задержку (в секундах) между событиями.

Значение по умолчанию: 1.

url_check

Источник `url_check` проверяет URL-адреса и создает события: `event.successful = "up"` при успехе (код 200) и `event.successful = "down"` при ошибке.

Источник предоставляет следующие настройки:

- `urls` – список URL-адресов для опроса.

  Обязательный параметр.

- `delay` – задержка (в секундах) между опросами.

  Значение по умолчанию: 1.

- `verify_ssl` – проверка SSL-сертификата.

  Значение по умолчанию: `true`.

webhook

Источник `webhook` получает события через веб-обработчики. Содержимое, присланное через `webhook`, доступно в событии через поле `payload`.

Источник предоставляет следующие настройки:

- `host` – IP-адрес сетевого интерфейса, на котором контейнер с источником ожидает входящие соединения.

  Значение по умолчанию: `0.0.0.0` (источник слушает все сетевые интерфейсы).

- `port` – порт для прослушивания.

  Обязательный параметр.

- `token` – токен аутентификации, ожидаемый от клиента.
- `certfile` – путь к файлу сертификата для включения TLS.
- `keyfile` – путь к файлу сертификата, используемому вместе с `certfile`.
- `password` – пароль для загрузки цепочки сертификатов.
- `cafile` – путь к файлу с сертификатами удостоверяющего центра для проверки клиентских сертификатов.
- `capath` – путь к каталогу с сертификатами удостоверяющего центра (используется для mTLS).
- `hmac_secret` – секретный ключ HMAC для проверки полезной нагрузки клиента.
- `hmac_algo` – алгоритм HMAC для вычисления хеша полезной нагрузки.

  Примечание

  Полный список возможных значений определяется используемой версией библиотеки `hashlib` и хранится в переменной `algorighms_available`.

  Значение по умолчанию: `sha256`.

- `hmac_header` – заголовок HMAC, отправляемый клиентом с подписью полезной нагрузки.

  Значение по умолчанию: `x-hub-signature-256`.

- `hmac_format` – формат подписи HMAC.

  Возможные значения:

  - hex;
  - base64.

  Значение по умолчанию: hex.

12.6.2 Действия

При наступлении событий контроллер EDA может выполнять следующие действия:

- *debug*;
- *none*;
- *post_event*;
- *print_event*;
- *retract_fact*;
- *run_job_template*;
- *run_module*;
- *run_playbook*;
- *run_workflow_template*;
- *set_fact*;
- *shutdown*.

debug

Действие debug используется для вывода отладочной информации.

Настройки:

- `msg` – строка или массив строк для вывода в stdout. Может содержать ссылки на события.
- `var` – переменная для вывода в stdout. Может содержать ссылки на события.

  Использование выражения Jinja `{{ }}` необязательно.

Если debug вызывается без аргументов, он выводит события, соответствующие текущей задаче.

Важно

Параметры `msg` и `var` взаимоисключающие – используйте только один из них.

Примеры использования debug

- Отладка с одиночным сообщением.

  Пример вывода простого отладочного сообщения при условии `event.i >= 5`:

```
name: debug with single message
condition: event.i >= 5
action:
  debug:
    msg: Simple debug message
```

- Отладка с несколькими сообщениями.

  Здесь debug выводит массив строк, включая ссылку на событие:

```
name: debug with multiple messages
condition: event.i >= 5
action:
  debug:
    msg:
      - "Message 1 {{ event }}"
      - Second Message
```

- Отладка с переменной.

  Этот вариант выводит значение `event.i`:

```
name: debug with var
condition: event.i >= 5
action:
  debug:
    var: event.i
```

none

Действие none не выполняет никаких операций. Оно полезно при написании тестов, когда необходимо определить условие без последующих действий.

Данное действие не имеет настроек.

Пример использования none

Проверка срабатывания правила без выполнения реального действия:

```
name: test condition
condition: event.status == "ready"
action:
  none: {}
```

post_event

Действие post_event отправляет событие в активированный набор правил.

Настройки:

- event – словарь событий для отправки.

  Обязательный параметр.

- ruleset – название набора правил, в который будет отправлено событие.

  Если не указано, используется текущий набор правил.

Важно

Префиксы events и facts имеют область видимости только внутри правил и не могут быть использованы за их пределами. При доступе к данным события используйте выражение Jinja {{ }}.

Примеры использования post_event

- Пример 1.

  В активный набор правил передается переменная j со значением 4.

```
action:
  post_event:
    ruleset: Test rules4
    event:
      j: 4
```

- Пример 2. Использование данных, сохраненных с помощью оператора присваивания.

```
name: multiple conditions
condition:
  all:
    - events.first << event.i == 0
    - events.second << event.i == 1
    - events.third << event.i == events.first.i + 2
action:
  post_event:
    ruleset: Test rules4
    event:
      data: "{{events.third}}"
```

print_event

Действие `print_event` выводит данные события в стандартный поток вывода (stdout).

Настройки:

- `pretty` – форматирование вывода:
  - `true` – в удобочитаемом виде (отформатированный JSON);
  - `false` – без форматирования.

  Значение по умолчанию: `true`.
- `var_root` – ключ в глубоко вложенном словаре, значение которого будет заменять исходные данные события при выводе. Может принимать словарь, если необходимо обработать несколько событий.

Примеры использования `print_event`

- Пример 1. Форматированный вывод только данных, хранящихся в ключе `i`

```
action:
  print_event:
    pretty: true
    var_root: i
```

- Пример 2.

  На этапе проверки условий в переменные `events.webhook` и `events.kafka` записываются значения ключей `event.webhook.payload.url` и `event.kafka.message.channel`, которые затем выводятся в стандартный поток вывода.

```
name: Multiple events with var_root
condition:
  all:
    - events.webhook << event.webhook.payload.url == "http://www.example.com"
    - events.kafka << event.kafka.message.channel == "red"
action:
  print_event:
    var_root:
      webhook.payload: webhook
      kafka.message: kafka
```

retract_fact

Действие `retract_fact` удаляет факт из рабочего набора правил.

Настройки:

- `fact` – удаляемый словарь фактов.

  Обязательный параметр.

- `ruleset` – название набора правил, из которого будет удален факт.

  Если не указано, используется текущий набор правил.

- `partial` – является ли удаляемый факт неполным (содержит не все ключи).

  Значение по умолчанию: `true`.

Пример использования `retract_fact`

Удаление факта `{ j: 3 }` из указанного набора правил:

```
action:
  retract_fact:
    ruleset: Test rules4
    fact:
      j: 3
```

run_job_template

Действие `run_job_template` запускает шаблон задания.

Важно

- Для использования этого действия необходимо передать параметры командной строки:
  - `--controller-url`;
  - `--controller-token` (или `--controller-username` вместе с `--controller-password`).
- Чтобы получить доступ к информации о событии в пространстве имен `ansible_eda`, в настройках шаблона задания включите флаг **Запрос при запуске** (Prompt on launch) для поля **Переменные** (Variables). В качестве альтернативы можно создать опрос, включающий переменную `ansible_eda`.

  Если необходимо ограничить список управляемых узлов на основе информации о событии, в настройках шаблона задания включите флаг **Запрос при запуске** (Prompt on launch) для поля **Лимит** (Limit).

Настройки:

- `name` – название шаблона задания.

  Обязательный параметр.

- `organization` – название организации, которой принадлежит шаблон задания.

  Обязательный параметр.

- `set_facts` – сохранение артефактов выполнения задания на основе шаблона в наборе правил как фактов.
- `post_events` – сохранение артефактов выполнения задания на основе шаблона в наборе правил как событий.
- `ruleset` – название набора правил, в который будут добавлены факты или события.

  Значение по умолчанию: текущий набор правил.

- `retry` – перезапуск задания в случае неудачи.

  Значение по умолчанию: `false`.

- `retries` – количество повторных попыток перезапуска задания на основе шаблона в случае неудачи.
- `delay` – интервал между попытками перезапуска (в секундах).

- var_root – ключ в глубоко вложенном словаре, значение которого заменяет исходные данные события. Может принимать словарь для нескольких совпадающих событий.
- job_args – дополнительные аргументы для API запуска задания на основе шаблона. Данные опроса и другие дополнительные переменные передаются через extra_vars. События и факты автоматически добавляются в extra_vars.

Пример использования retract_fact

Запуск задания на основе шаблона Deploy App, принадлежащего организации DevOps. Automation Controller попытается выполнить задание три раза. Интервал между попытками – 10 секунд.

```
action:
  run_job_template:
    name: Deploy App
    organization: DevOps
    retry: true
    retries: 3
    delay: 10
```

run_module

Действие run_module выполняет модуль Ansible с указанными параметрами.

Настройки:

- name – название модуля Ansible в формате *FQCN*.

  Обязательный параметр.
- module_args – аргументы, передаваемые модулю.
- retry – перезапуск модуля в случае неудачи.

  Значение по умолчанию: false.
- retries – количество повторных попыток перезапуска модуля в случае неудачи.

  Значение по умолчанию: 0.
- delay – интервал между попытками повторного запуска (в секундах).
- verbosity – уровень детализации вывода (от 1 до 4).
- extra_vars – дополнительные переменные, передаваемые в набор сценариев.
- json_mode – отправка данных событий, произошедших при выполнении сценариев, в stdout в формате JSON.
- set_facts – сохранение артефактов выполнения модуля в наборе правил как фактов.
- post_events – сохранение артефактов выполнения модуля в наборе правил как событий.
- ruleset – название набора правил, в который будут добавлены факты или события.

  Значение по умолчанию: текущий набор правил.
- var_root – ключ в глубоко вложенном словаре, значение которого заменит исходные данные события.

Пример использования run_module

Три попытки выполнения команды uptime с интервалом 5 секунд:

```
action:
  run_module:
    name: ansible.builtin.command
    module_args:
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
      cmd: "uptime"
    retry: true
    retries: 3
    delay: 5
```

run_playbook

Действие run_playbook выполняет набор сценариев с заданными параметрами.

Настройки:

- name – название набора сценариев в формате *FQCN* или путь к файлу. Относительный путь указывайте по отношению к каталогу, в котором выполняется команда ansible-rulebook.

  Обязательный параметр.
- set_facts – сохранение артефактов выполнения сценариев в наборе правил как фактов.
- post_events – сохранение артефактов выполнения сценариев в наборе правил как событий.
- ruleset – название набора правил, в который будут добавлены факты или события.

  Значение по умолчанию: текущий набор правил.
- retry – перезапуск набора сценариев в случае неудачи.
- retries – количество повторных попыток перезапуска набора сценариев в случае неудачи.
- delay – интервал между попытками повторного запуска (в секундах).
- verbosity – уровень детализации вывода (от 1 до 4).
- var_root – ключ в глубоко вложенном словаре, значение которого заменит исходные данные события.
- extra_vars – дополнительные переменные, передаваемые в сценарии.
- json_mode – отправка данных событий, произошедших при выполнении сценариев, в stdout в формате JSON.
- copy_files – копирование локального файла сценариев в каталог проекта ansible-runner. Не требуется, если набор сценариев выполняется из коллекции Ansible.

Пример использования run_playbook

Запуск набора сценариев с дополнительными переменными и повторными попытками:

```
action:
  run_playbook:
    name: /home/user/ansible/playbook.yml
    extra_vars:
      my_var: "Some value"
    retry: true
    retries: 2
    delay: 10
    verbosity: 3
```

run_workflow_template

Действие run_workflow_template запускает поток заданий на основе шаблона.

Важно

Для использования этого действия необходимо передать параметры командной строки:

- `--controller-url`;
- `--controller-token` (или `--controller-username` вместе с `--controller-password`).

Настройки:

- `name` – название шаблона потока заданий.

  Обязательный параметр.

- `organization` – название организации, которой принадлежит шаблон потока заданий.

  Обязательный параметр.

- `set_facts` – сохранение артефактов выполнения заданий в наборе правил как фактов.
- `post_events` – сохранение артефактов выполнения заданий в наборе правил как событий.
- `ruleset` – название набора правил, в который будут добавлены факты или события.

  Значение по умолчанию: текущий набор правил.

- `retry` – перезапуск потока заданий в случае неудачи.
- `retries` – количество повторных попыток перезапуска потока заданий в случае неудачи.
- `delay` – интервал между попытками повторного запуска (в секундах).
- `var_root` – ключ в глубоко вложенном словаре, значение которого заменит исходные данные события.
- `job_args` – дополнительные аргументы для API запуска потока заданий. События и факты автоматически добавляются в `extra_vars`.

Пример использования `run_workflow_template`

Запуск потока заданий резервного копирования базы данных:

```
action:
  run_workflow_template:
    name: Database Backup
    organization: Operations
    job_args:
      extra_vars:
        backup_location: "/mnt/backups"
        retention_days: 7
```

set_fact

Действие `set_fact` используется для добавления фактов в рабочий набор правил.

Настройки:

- `fact` – словарь фактов для добавления.

  Обязательный параметр.

- `ruleset` – название набора правил, в который добавляется факт.

  Значение по умолчанию: текущий набор правил.

Примеры использования set_fact

- Добавление факта в указанный набор правил.

```
action:
  set_fact:
    ruleset: Test rules4
    fact:
      j: 1
```

- Добавление факта на основе сохраненных данных события.

```
name: multiple conditions
condition:
  all:
    - events.first << event.i == 0
    - events.second << event.i == 1
    - events.third << event.i == events.first.i + 2
action:
  set_fact:
    ruleset: Test rules4
    fact:
      data: "{{events.first}}"
```

- Добавление факта при выполнении единственного условия.

```
name: single condition
condition: event.i == 23
action:
  set_fact:
    fact:
      myfact: "{{event.i}}"
```

Примечание

- Используйте выражение Jinja {{ }} для подстановки значений из событий.
- Факт можно добавить только в один набор правил. Нельзя одновременно добавить его в несколько наборов.

shutdown

Действие shutdown используется для завершения работы свода правил. Оно генерирует событие завершения работы, которое останавливает выполнение всех запущенных наборов правил.

Важно

Если в своде правил выполняются несколько наборов правил, выполнение команды shutdown приведет к завершению всех остальных наборов правил. Необходимо учитывать работающие сценарии, которые могут быть затронуты, когда один из наборов правил завершит работу. Сообщение о завершении работы передается всем выполняющимся наборам правил.

Настройки:

- delay – время задержки перед завершением работы (в секундах).

  Значение по умолчанию: 60.

- message – сообщение, связанное с завершением работы, например, причины.
- kind – тип завершения:

- `graceful` – мягкое: работа процессов завершается корректно;
- `now` – немедленное завершение: работа процессов прерывается без ожидания.

Значение по умолчанию: `graceful`.

Пример использования `shutdown`

Завершение работы с задержкой после 5 событий:

```
name: shutdown after 5 events
condition: event.i >= 5
action:
  shutdown:
    delay: 0.125
    message: Shutting down after 5 events
```

ГЛАВА 13

Управление доступом

В Astra Automation управление доступом основано на объединенном интерфейсе и наборе инструментов, обеспечивающих аутентификацию, авторизацию и разграничение прав пользователей. Из единого интерфейса можно настраивать глобальные и системные параметры, включать различные механизмы контроля доступа и управлять интеграцией с внешними сервисами идентификации. Система построена на ролевой модели доступа и поддерживает администрирование как через графический интерфейс, так и через REST API.

13.1 Основные механизмы и инструменты

Модель управления доступом в Astra Automation строится на следующих механизмах:

- *Ролевая модель доступа* (*RBAC*).

  Роли определяют, какие действия могут выполнять пользователи и команды. С их помощью регулируется доступ к объектам платформы: запуск заданий, редактирование проектов, управление инвентарями и учетными данными.

- Аутентификация пользователей.

  Поддерживаются как локальные учетные записи, так и интеграция с внешними провайдерами (LDAP, SAML и другие). Для внешних приложений и сервисов применяется аутентификация по токенам (OAuth 2.0).

- Карты аутентификаторов и отображение атрибутов (Authenticator Maps).

  Система отображения атрибутов позволяет автоматически назначать роли и привилегии пользователям на основании данных, получаемых от внешнего провайдера (например, групп LDAP или утверждений SAML).

- Команды и организации.

  Пользователей можно объединять в команды и организации для централизованного управления доступом к рабочим областям и объектам платформы. Администратор системы может делегировать полномочия по управлению этими командами или организациями.

- Авторизация по токенам (OAuth 2.0).

  Для интеграции со сторонними системами и инструментами DevOps применяются токены доступа. Они позволяют безопасно делегировать права и ограничивать доступ заданными разрешениями.

13.2 Возможности управления доступом в UI

Графический интерфейс (UI) предоставляет удобный способ настройки и администрирования доступа. Через UI администратор может выполнять следующие действия:

- управлять пользователями, командами и организациями;
- назначать и отзывать роли;
- создавать и настраивать методы аутентификации (LDAP, SAML, локальные и другие);
- задавать правила отображения (mapping rules) для автоматического назначения прав на основании атрибутов и групп.

13.3 Управление через API

Помимо графического интерфейса администрировать доступ можно с помощью REST API.

API предоставляет полный набор операций для управления доступом: создание, удаление и редактирование пользователей, ролей, организаций и карт аутентификаторов. Это позволяет интегрировать платформу с внешними системами IAM, CI/CD и другими инструментами DevOps.

13.3.1 Пример структуры управления доступом

Модель управления доступом в Astra Automation строится по иерархическому принципу и может быть представлена следующим образом:

- Организация → Команды → Пользователи.
- Каждой сущности назначаются роли через UI или API.
- Для сторонних приложений и сервисов настраиваются токены доступа OAuth 2.0.

Методы аутентификации

Astra Automation может использовать для аутентификации пользователей внешние системы, называемые *поставщиками идентификационных данных* (Identity Provider, IdP). Далее с целью сокращения они называются провайдерами.

При первоначальной настройке необходимо установить доверительные отношения между шлюзом Astra Automation и провайдером. Набор данных, используемых для идентификации контроллера на стороне провайдера аутентификации, зависит от типа провайдера и его собственных настроек.

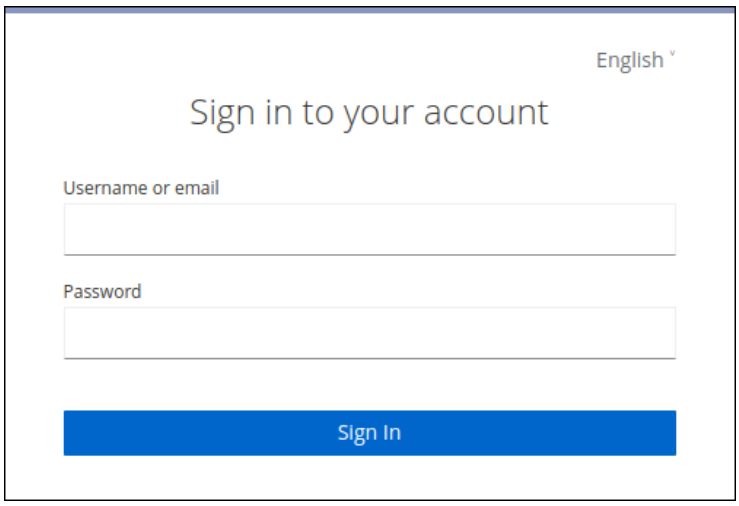
После настройки аутентификация пользователя Astra Automation через провайдер выполняется по следующему алгоритму:

1. В Astra Automation на странице аутентификации становится доступна кнопка аутентификации с использованием провайдера.

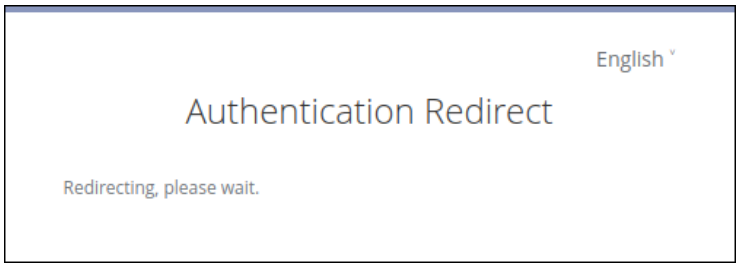
   Примечание

   Количество кнопок и их внешний вид зависят от количества и типов настроенных провайдеров.

2. При нажатии на кнопку Astra Automation перенаправляет пользователя на страницу аутентификации, заданную в настройках провайдера.



3. Пользователь вводит свои учетные данные на указанной странице. Если идентификация успешна, провайдер перенаправляет пользователя на страницу аутентификации Astra Automation.



   Также провайдер передает в Astra Automation сведения о пользователе и токен, подтверждающий подлинность данных.

4. Astra Automation проверяет токен и наличие в своей базе учетной записи пользователя.

   Если такой записи нет, она создается на основе данных, полученных от провайдера аутентификации. Запись о пользователе Astra Automation может содержать следующие поля:

   - `username` – название учетной записи;
   - `first_name` – фамилия;
   - `last_name` – имя;
   - `email` – адрес электронной почты.

   Если в настройках Astra Automation заданы правила ассоциации пользователей с организациями и командами, Astra Automation выполняет следующие действия:

   1. Создает организации и команды, отсутствующие в Astra Automation.
   2. Включает пользователя в организации и команды в соответствии с заданными настройками.

      На уровне организации и команды пользователю автоматически назначается роль «Администратор» или «Участник». Прочие роли при необходимости должны быть назначены вручную.

Список провайдеров аутентификации, которые может использовать Astra Automation для реализации *SSO*:

- *Azure AD*;
- *GitHub*;
- *Google OAuth2*;
- *LDAP*;

- *RADIUS*;
- *SAML*;
- *TACACS+*;
- *OIDC*.

Отключение встроенной аутентификации

В Astra Automation есть функция полного отключения встроенной (локальной) аутентификации. Она позволяет отключить все встроенные учетные записи пользователей для обеспечения входа в систему только через внешние механизмы аутентификации, например, LDAP, SAML, OAuth2 и другие.

Внимание

Перед отключением встроенной аутентификации выполните следующие действия:

1. Убедитесь, что как минимум один внешний провайдер (LDAP, SAML или другой) настроен и проверен.
2. Проверьте, что внешнему пользователю назначена роль «Системный администратор» (System Administrator) в Astra Automation.

В случае невыполнения этих действий вы можете потерять доступ к Astra Automation.

Чтобы отключить встроенную аутентификацию через графический интерфейс, выполните следующие действия:

1. Откройте графическую консоль Astra Automation с привилегиями системного администратора.
2. На панели навигации выберите **Настройки > Шлюз платформы** (Settings > Platform gateway).
3. Нажмите кнопку *Изменить настройки шлюза платформы* (Edit platform gateway settings).
4. В разделе **Безопасность** (**Security**) включите настройку **Базовая аутентификация шлюза включена** (Gateway basic auth enabled).
5. Нажмите кнопку *Сохранить настройки шлюза платформы* (Save platform gateway settings).

Azure AD

Для использования этого провайдера аутентификации необходимы ключ и секрет Azure AD. Пошаговые инструкции по их получению приведены в документации Azure[204].

Пошаговые инструкции по настройке провайдера через графический интерфейс Astra Automation приведены в секции *Методы аутентификации*.

Ассоциация с организациями

Настройки ассоциации пользователей Azure AD с организациями контроллера необходимо задавать в виде словаря, в котором ключи – названия организаций Astra Automation, а значения – вложенные словари с настройками.

Словарь с настройками может содержать следующие ключи:

- `admins` – правила назначения роли «Администратор»;
- `remove_admins` – управление отзывом роли «Администратор»;
- `users` – правила назначения роли «Участник»;

[204] https://learn.microsoft.com/en-us/entra/identity-platform/quickstart-register-app

- `remove_users` – управление отзывом роли «Участник».

Обработка значений параметров `admins` и `users` выполняется следующим образом:

- Если значение не задано – список администраторов и участников организации не меняется.
- `true` – всем пользователям Azure AD, выполнившим аутентификацию в Astra Automation, автоматически назначается роль «Администратор» или «Участник» соответственно.
- `false` – пользователям Azure AD автоматически назначается организационная роль «Администратор» или «Участник» соответственно. Аутентификация в Astra Automation не требуется.
- Если в значении параметра указана строка или массив строк, то на соответствие этим строкам проверяются названия учетных записей и адреса электронной почты пользователей Azure AD.

  Строки, в начале и конце которых стоит символ `/`, обрабатываются как регулярные выражения. Для регулярных выражений поддерживаются модификаторы, которые ставятся после закрывающей косой черты `/`:

  - `i` – игнорирование регистра символов;
  - `m` – многострочное выражение.

Обработка значений параметров `remove_admins` и `remove_users` выполняется следующим образом:

- `true` – организационная роль отзывается.
- `false` – организационная роль не отзывается.

Пример заполнения словаря:

```
{
  "Jupiter": {
    "admins": "admin@jupiter.example.com",
    "users": [
      "/.*?@users\.example\.com$/i",
      "/.*?@devops\.example\.com$/i"
    ],
    "remove_admins": true,
    "remove_users": true
  }
}
```

Если организация «Jupiter» не существует, она будет создана.

Пользователю с адресом электронной почты `admin@jupiter.example.com` назначается роль «Администратор». У прочих пользователей Azure AD эта роль автоматически отзывается.

Пользователям, адрес электронной почты которых заканчивается на `@users.example.com` или `@devops.example.com` (регистр символов в адресе электронной почты игнорируется), назначается организационная роль «Участник». У прочих пользователей Azure AD эта роль автоматически отзывается.

Ассоциация с командами

Настройки ассоциации пользователей Azure AD с командами контроллера необходимо задавать в виде словаря, где ключи – названия команды в контроллере, а значения – вложенные словари с настройками.

Словарь с настройками может содержать следующие ключи:

- `organization` – название организации, которой принадлежит команда.
  - Если организация с указанным названием не существует, она будет создана.

  - Если в организации не существует команда с указанным названием, она будет создана.

- `users` – параметр, определяющий членство пользователей в команде.
  - Если значение параметра не задано, список участников команды не меняется.
  - Если значение параметра равно `true`, пользователям назначается командная роль «Участник».
  - Если значение параметра равно `false`, у пользователей отзывается командная роль «Участник».
  - Если в значении параметра указана строка или массив строк, то на соответствие этим строкам проверяются названия учетных записей и адреса электронной почты пользователей Azure AD.

  Строки, в начале и конце которых стоит символ `/`, обрабатываются как регулярные выражения. Для регулярных выражений поддерживаются модификаторы, которые ставятся после закрывающей косой черты `/`:
  - `i` – игнорирование регистра символов;
  - `m` – многострочное выражение.
- `remove` – если значение этого параметра равно `true`, у не соответствующих критериям отбора пользователей Azure AD автоматически отзывается роль «Участник».

Пример заполнения словаря:

```
{
  "DevOps Team": {
    "organization": "Jupiter",
    "users": "/.*?@jupiter\.example\.com$/",
    "remove": true
  }
}
```

Если организация «Jupiter» не существует, она будет создана.

Если в организации «Jupiter» не существует команда «DevOps Team», она будет создана.

Пользователям, адрес электронной почты которых заканчивается на `@jupiter.example.com`, автоматически назначается роль «Участник».

GitHub

Важно

Эта часть документации находится в стадии разработки.

Google OAuth2

Важно

Эта часть документации находится в стадии разработки.

LDAP

Astra Automation поддерживает использование нескольких конфигураций провайдера аутентификации LDAP одновременно.

Для использования метода аутентификации LDAP необходимы следующие данные:

- URI серверов LDAP;

- пароли привязки к LDAP;
- запрос для поиска пользователей в LDAP;
- запрос для поиска групп в LDAP.

Для получения этих данных обратитесь к документации используемого провайдера аутентификации LDAP.

Автоматическая синхронизация пользователей LDAP с Astra Automation не выполняется.

При успешной аутентификации в LDAP для пользователя автоматически создается учетная запись в Astra Automation. У созданных таким образом учетных записей нельзя изменить название учетной записи, пароль, имя и фамилию пользователя.

По умолчанию Astra Automation использует для подключения к серверам LDAP настройки, обеспечивающие корректную работу с реализацией протокола в Microsoft Active Directory. Для прочих реализаций может быть использован один из обработчиков:

- `ActiveDirectoryGroupType`;
- `GroupOfNamesType`;
- `GroupOfUniqueNamesType`;
- `MemberDNGroupType`;
- `NestedActiveDirectoryGroupType`;
- `NestedGroupOfNamesType`;
- `NestedGroupOfUniqueNamesType`;
- `NestedMemberDNGroupType`;
- `NestedOrganizationalRoleGroupType`;
- `OrganizationalRoleGroupType`;
- `PosixGroupType`;
- `PosixUIDGroupType`.

Подробное описание каждого типа обработчика см. в документации[205] библиотеки `django-auth-ldap`.

Ассоциация данных пользователей

Настройки ассоциации данных пользователей LDAP с данными пользователей Astra Automation необходимо задавать в виде словаря, в котором ключи – названия полей данных пользователя Astra Automation, а значения – названия полей данных пользователя LDAP, например:

```
{
  "first_name": "givenName",
  "last_name": "sn",
  "email": "mail"
}
```

Назначение ролей «Системный администратор» и «Системный аудитор»

Возможно автоматическое назначение пользователям LDAP типов «Системный администратор» (System Administrator) и «Системный аудитор» (System Auditor). Настройки необходимо задавать в виде словаря, в котором ключи – `is_superuser` и `is_system_auditor` соответственно, а значения – запросы к LDAP для фильтрации пользователей.

Пример заполнения словаря:

[205] https://django-auth-ldap.readthedocs.io/en/stable/groups.html#types-of-groups

```
{
  "is_superuser": "CN=superuser,OU=groups,DC=example,DC=com",
  "is_system_auditor": "CN=auditors,OU=groups,DC=example,DC=com"
}
```

Ассоциация с организациями

Настройки ассоциации пользователей LDAP с организациями контроллера необходимо задавать в виде словаря, в котором ключи – названия организаций Astra Automation, а значения – вложенные словари с настройками.

Вложенный словарь может содержать следующие ключи:

- `admins` – правила назначения роли «Администратор»;
- `remove_admins` – управление отзывом роли «Администратор»;
- `users` – правила назначения роли «Участник»;
- `remove_users` – управление отзывом роли «Участник».

Обработка значений параметров `admins` и `users` выполняется следующим образом:

- Если значение не задано – список администраторов и участников организации не меняется.
- `true` – всем пользователям LDAP, выполнившим аутентификацию в Astra Automation, автоматически назначается роль «Администратор» или «Участник» соответственно.
- `false` – пользователям LDAP автоматически назначается роль «Администратор» или «Участник» соответственно.
- Если в значении параметра указана строка или массив строк, они интерпретируются как запрос к LDAP. Найденным пользователям автоматически назначается роль «Администратор» или «Участник» соответственно.

Пример заполнения словаря:

```
{
  "Jupiter": {
    "admins": "CN=admins,OU=groups,DC=jupiter,DC=example,DC=com",
    "users": [
      "CN=devops,OU=groups,DC=jupiter,DC=example,DC=com",
      "CN=testers,OU=groups,DC=jupiter,DC=example,DC=com"
    ],
    "remove_admins": true,
    "remove_users": true
  }
}
```

Если организация «Jupiter» не существует, она будет создана.

Пользователям, найденным при выполнении запроса `CN=admins,OU=groups,DC=jupiter,DC=example,DC=com`, назначается роль «Администратор». У прочих пользователей LDAP эта роль автоматически отзывается.

Пользователям, найденным при выполнении запросов `CN=devops,OU=groups,DC=jupiter,DC=example,DC=com` и `CN=terster,OU=groups,DC=jupiter,DC=example,DC=com`, назначается роль «Участник». У прочих пользователей LDAP эта роль автоматически отзывается.

Ассоциация с командами

Настройки ассоциации пользователей LDAP с командами контроллера необходимо задавать в виде словаря, где ключи – названия команд в контроллере, а значения – вложенные словари с настройками.

Словарь с настройками может содержать следующие ключи:

- `organization` – название организации, которой принадлежит команда.
  - Если организация с указанным названием не существует, она будет создана.
  - Если в организации не существует команда с указанным названием, она будет создана.
- `users` – параметр, определяющий назначение пользователю роли «Участник»:
  - Если значение параметра не задано, список участников команды не меняется.
  - Если значение параметра равно `true`, пользователю назначается роль «Участник».
  - Если значение параметра равно `false`, у пользователя отзывается роль «Участник».
  - Если в значении параметра указана строка или массив строк, то они интерпретируются как запросы к LDAP. Найденным пользователям назначается роль «Участник».
- `remove` – если значение этого параметра равно `true`, не соответствующие критериям отбора пользователи LDAP удаляются из списка участников команды.

Пример заполнения словаря:

```
{
  "DevOps Team": {
    "organization": "Jupiter",
    "users": "CN=devops,OU=groups,DC=jupiter,DC=example,DC=com",
    "remove": true
  }
}
```

Если организация «Jupiter» не существует, она будет создана.

Если в организации «Jupiter» не существует команда «DevOps Team», она будет создана.

Пользователям, найденным в LDAP при выполнении запроса `CN=devops,OU=groups,DC=jupiter,DC=example,DC=com`, автоматически назначается роль «Участник».

RADIUS

Предупреждение

Этот провайдер аутентификации считается устаревшим и в одной из следующих версий Astra Automation станет недоступным для применения.

SAML

В Astra Automation *поставщиком услуг* является кластер Astra Automation, а поставщиком идентификационных данных – группа серверов SAML, предоставляющих услугу *SSO*.

В качестве идентификатора объекта поставщика услуг SAML следует указать базовый URL Astra Automation. Если для доступа к кластеру используется балансировщик нагрузки, следует указать его URL, например:

```
https://ac.example.com/
```

Открытый сертификат и закрытый ключ должны быть указаны с маркерами начала и конца содержимого:

Открытый сертификат

```
-----BEGIN CERTIFICATE-----
AAAAA231jfdsalfkjkl........
-----END CERTIFICATE-----
```

Закрытый ключ

```
-----BEGIN PRIVATE KEY-----
PPPPP2183910das0921........
-----END PRIVATE KEY-----
```

Информацию об организации поставщика услуг SAML необходимо указывать в виде словаря, где ключи – названия локалей, а значения – вложенные словари с настройками.

Словарь с настройками может содержать следующие ключи:

- `url` – URL кластера Astra Automation.
- `displayname` – человекочитаемое название кластера Astra Automation.
- `name` – внутреннее название кластера Astra Automation. Используется в качестве идентификатора на стороне провайдера SAML, позволяя ему отличать один кластер от другого.

Пример заполнения словаря:

```
{
  "ru-RU": {
    "url": "https://ac.example.com",
    "displayname": "Кластер Automation Controller",
    "name": "ac-cluster"
  }
}
```

Атрибуты пользователей

По умолчанию Astra Automation ожидает от провайдера SAML данные пользователя в следующем формате:

```
Username(urn:oid:0.9.2342.19200300.100.1.1)
Email(urn:oid:0.9.2342.19200300.100.1.3)
FirstName(urn:oid:2.5.4.42)
LastName(urn:oid:2.5.4.4)
```

Если на стороне провайдера SAML данные о пользователях хранятся в полях с другими названиями, необходимо задать настройки ассоциации атрибутов пользователей SAML с атрибутами пользователей Astra Automation.

Настройки ассоциации атрибутов пользователей SAML с атрибутами пользователей Astra Automation необходимо задавать в виде словаря, где ключи – идентификаторы провайдеров идентификации, а значения – вложенные словари с настройками.

Словарь с настройками может содержать следующие ключи:

- `attr_user_permanent_id` – название атрибута пользователя SAML, в котором хранится уникальный идентификатор пользователя.
- `attr_username` – название атрибута SAML, в котором хранится название учетной записи пользователя.
- `attr_first_name` – название атрибута SAML, в котором хранится имя пользователя.
- `attr_last_name` – название атрибута SAML, в котором хранится фамилия пользователя.
- `attr_email` – название атрибута SAML, в котором хранится адрес электронной почты пользователя.

> **Предупреждение**
>
> Использование одного и того же адреса электронной почты, в том числе для пользователей, созданных средствами Astra Automation, запрещено.

- `entity_id` – идентификатор профиля, используемого для аутентификации Astra Automation. Предоставляется провайдером SAML.
- `url` – URL страницы аутентификации SAML. Astra Automation будет перенаправлять пользователей на эту страницу при аутентификации через SAML. Предоставляется провайдером SAML.
- `x509_cert` – сертификат провайдера идентификации SAML.

  При заполнении этого поля необходимо убрать из сертификата маркеры начала (`-----BEGIN CERTIFICATE`) и конца (`-----END CERTIFICATE----`), а оставшиеся строки объединить в одну.

Пример заполнения словаря:

```
{
  "keycloak": {
    "entity_id": "https://saml.example.com/keycloak/b2912kdsjkl1k",
    "url": "https://saml.example.com/keycloak/sso/",
    "x509cert": "MIID0DCCAj12309jklwqW...3123jklfdsajkl213",
    "attr_user_permanent_id": "user_id",
    "attr_username": "Email",
    "attr_email": "Email",
    "attr_first_name": "FirstName",
    "attr_last_name": "LastName"
  }
}
```

Ассоциация атрибутов организаций

Настройки ассоциации атрибутов SAML с организациями необходимо задавать в виде словаря, который может содержать следующие ключи:

- `saml_attr` – если в данных пользователя SAML есть атрибут с указанным названием, этому пользователю автоматически назначается роль «Участник» в *соответствующих организациях контроллера*.
- `saml_admin_attr` – если в данных пользователя SAML есть атрибут с указанным названием, этому пользователю автоматически назначается роль «Администратор» в *соответствующих организациях контроллера*.
- `remove` – если значение этого параметра равно `true`, у не соответствующих критериям отбора пользователей SAML автоматически отзывается роль «Участник».
- `remove_admins` – если значение этого параметра равно `true`, у не соответствующих критериям отбора пользователей SAML автоматически отзывается роль «Администратор».

Пусть имеется следующее описание атрибутов SAML:

```
<saml2:AttributeStatement>
  <saml2:Attribute FriendlyName="is-member"
                   Name="is-member"
                   NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">

    <saml2:AttributeValue>org.jptr</saml2:AttributeValue>
    <saml2:AttributeValue>org.strn</saml2:AttributeValue>
  </saml2:Attribute>

  <saml2:Attribute FriendlyName="is-admin"
                   Name="is-admin"
                   NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">

    <saml2:AttributeValue>org.jptr</saml2:AttributeValue>
  </saml2:Attribute>
</saml2:AttributeStatement>
```

Здесь:

- атрибут `is-member` указывает, что пользователь является участником организаций SAML `org.jptr` и `org.strn`;
- атрибут `is-admin` указывает, что пользователь является администратором организации SAML `org.jptr`.

Соответствующий словарь ассоциаций атрибутов организаций имеет следующий вид:

```
{
  "saml_attr": "is-member",
  "saml_admin_attr": "is-admin",
  "remove": true,
  "remove_admins": true
}
```

Пользователям SAML, обладающим атрибутом `is-member`, назначается роль «Участник» в соответствующих организациях контроллера. У прочих пользователей SAML эта роль автоматически отзывается.

Пользователям SAML, обладающим атрибутом `is-admin`, назначается роль «Администратор» в соответствующих организациях контроллера. У прочих пользователей SAML эта роль автоматически отзывается.

Ассоциация атрибутов команд

Настройки ассоциации атрибутов SAML с командами необходимо задавать в виде словаря, который может содержать следующие ключи:

- `saml_attr` – если в данных пользователя SAML есть атрибут с указанным названием, этому пользователю автоматически назначается роль «Участник» в *соответствующих командах контроллера*.
- `team_org_map` – массив словарей, содержащих настройки ассоциации пользователей SAML с командами контроллера.

  Каждый словарь в этом массиве может содержать следующие ключи:

  - `team` – название команды.
  - `team_alias` – псевдоним команды.
  - `organization` – название организации, которой принадлежит команда.
  - `organization_alias` – псевдоним организации, которой принадлежит команда.
    - Если организация контроллера с указанным названием не существует, она будет создана.
    - Если в организации контроллера не существует команда с указанным названием, она будет создана.

  Примечание

  Ключи `team_alias` и `organization_alias` не обязательны для заполнения, но они могут быть полезны, если на стороне поставщика услуг SAML используются сложные названия организаций и команд. Если значения ключей `team_alias` и `organization_alias` не заданы, организации и команды будут созданы в контроллере с названиями, полученными из атрибутов, указанных в поле `team` и `organization` соответственно.

- `remove` – если значение этого параметра равно `true`, у не соответствующих критериям отбора пользователей SAML автоматически отзывается роль «Участник».

Пусть имеется следующее описание атрибутов SAML:

```
<saml:AttributeStatement>
  <saml:Attribute xmlns:x500="urn:oasis:names:tc:SAML:2.0:profiles:attribute:X500"
                  x500:Encoding="LDAP"
                  NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
                  Name="urn:oid:1.3.6.1.4.1.5923.1.1.1.1"
                  FriendlyName="team">
    <saml:AttributeValue xsi:type="xs:string">data:team:devOps</saml:AttributeValue>
    <saml:AttributeValue xsi:type="xs:string">data:team:auditors</saml:AttributeValue>
  </saml:Attribute>
</saml:AttributeStatement>
```

Здесь атрибут `team` указывает, что пользователь является участником команд SAML `data:team:devOps` и `data:team:auditors`.

Соответствующий словарь ассоциаций атрибутов команды имеет следующий вид:

```
{
  "saml_attr": "team",
  "remove": true,
  "team_org_map": [
    {
      "team": "data:team:devOps",
      "team_alias": "DevOps Team"
      "organization": "org.jptr"
    }, {
      "team": "data:team:auditors",
      "team_alias": "Auditors Team",
      "organization": "org.jptr"
    }
  ]
}
```

Если в контроллере не существует организация `org.jptr`, она будет создана.

Если в организации контроллера `org.jptr` не существуют команды «DevOps Team» и «Auditors Team», они будут созданы. Этим командам контроллера соответствуют команды SAML `data:team:devOps` и `data:team:auditors` соответственно.

Пользователям SAML, обладающим атрибутом `team`, хотя бы одно из значений которого равно `data:team:devOps`, автоматически назначается роль «Участник» в команде «DevOps Team». У прочих пользователей эта роль автоматически отзывается.

Пользователям SAML, обладающим атрибутом `team` со значением, равным `data:team:auditors`, автоматически назначается роль «Участник» в команде «Auditors Team». У прочих пользователей эта роль автоматически отзывается.

Ассоциация с организациями

Настройки ассоциации пользователей SAML с организациями контроллера необходимо задавать в виде словаря, в котором ключи – названия организаций Astra Automation, а значения – вложенные словари с настройками.

Словарь с настройками может содержать следующие ключи:

- `admins` – правила назначения роли «Администратор»;
- `remove_admins` – управление отзывом роли «Администратор»;
- `users` – правила назначения роли «Участник»;
- `remove_users` – управление отзывом роли «Участник».

Обработка значений параметров `admins` и `users` выполняется следующим образом:

- Если значение не задано – список администраторов и участников организации не меняется.

- `true` – всем пользователям SAML, выполнившим аутентификацию в Astra Automation, автоматически назначается роль «Администратор» или «Участник» соответственно.
- `false` – пользователям SAML автоматически назначается организационная роль «Администратор» или «Участник» соответственно. Аутентификация в Astra Automation не требуется.
- Если в значении параметра указана строка или массив строк, то на соответствие этим строкам проверяются названия учетных записей пользователей SAML.

  Строки, в начале и конце которых стоит символ `/`, обрабатываются как регулярные выражения. Для регулярных выражений поддерживаются модификаторы, которые ставятся после закрывающей косой черты `/`:

  - `i` – игнорирование регистра символов;
  - `m` – многострочное выражение.

Обработка значений параметров `remove_admins` и `remove_users` выполняется следующим образом:

- `true` – организационная роль отзывается.
- `false` – организационная роль не отзывается.

Пример заполнения словаря:

```
{
  "Jupiter": {
    "admins": "admin@jupiter.example.com",
    "users": [
      "/.*?@users\.example\.com$/i",
      "/.*?@devops\.example\.com$/i"
    ],
    "remove_admins": true,
    "remove_users": true
  }
}
```

Если организация «Jupiter» не существует, она будет создана.

Пользователю с названием учетной записи `admin@jupiter.example.com` назначается роль «Администратор». У прочих пользователей SAML эта роль автоматически отзывается.

Пользователям, название учетной записи которых заканчивается на `@users.example.com` или `@devops.example.com` (без учета регистра), назначается организационная роль «Участник». Пользователи, не удовлетворяющие указанному условию, автоматически удаляются из списка участников организации «Jupiter».

Ассоциация с командами

Настройки ассоциации пользователей SAML с командами контроллера необходимо задавать в виде словаря, где ключи – названия команды в контроллере, а значения – вложенные словари с настройками.

Словарь с настройками может содержать следующие ключи:

- `organization` – название организации, которой принадлежит команда.
  - Если организация с указанным названием не существует, она будет создана.
  - Если в организации не существует команда с указанным названием, она будет создана.
- `users` – параметр, определяющий членство пользователей в команде.
  - Если значение параметра не задано, список участников команды не меняется.
  - Если значение параметра равно `true`, пользователям назначается командная роль «Участник».

  - Если значение параметра равно `false`, у пользователей отзывается командная роль «Участник».
  - Если в значении параметра указана строка или массив строк, то на соответствие этим строкам проверяются названия учетных записей SAML. В *настройках ассоциации атрибутов пользователей* указывается, какой из атрибутов SAML считать названием учетной записи.

  Строки, в начале и конце которых стоит символ `/`, обрабатываются как регулярные выражения. Для регулярных выражений поддерживаются модификаторы, которые ставятся после закрывающей косой черты `/`:

  - `i` – игнорирование регистра символов;
  - `m` – многострочное выражение.

- `remove` – если значение этого параметра равно `true`, у не соответствующих критериям отбора пользователей SAML автоматически отзывается роль «Участник».

Пример заполнения словаря:

```
{
  "DevOps Team": {
    "organization": "Jupiter",
    "users": "/.*?@jupiter\.example\.com$/",
    "remove": true
  }
}
```

Если организация «Jupiter» не существует, она будет создана.

Если в организации «Jupiter» не существует команда «DevOps Team», она будет создана.

Пользователям, название учетной записи которых заканчивается на `@jupiter.example.com`, автоматически назначается роль «Участник» на уровне команды «DevOps Team».

Ассоциация атрибутов и ролей SAML с ролями контроллера

Настройки ассоциации атрибутов и ролей SAML с типами пользователей Astra Automation необходимо задавать в виде словаря, который может содержать следующие ключи:

- `is_superuser_role` – название роли или ролей SAML, при наличии которых пользователю Astra Automation назначается роль «Системный администратор».
- `is_superuser_attr` – название атрибута SAML, используемого для принятия решения о назначении пользователю Astra Automation роли «Системный администратор».
- `is_superuser_value` – значение атрибута SAML, необходимое для назначения пользователю Astra Automation роли «Системный администратор».
- `remove_superusers` – управление автоматическим отзывом роли «Системный администратор» у пользователей, не соответствующим критериям:
  - `true` – роль отзывается;
  - `false` – роль не отзывается.
- `is_system_auditor_role` – название роли или ролей SAML, при наличии которых пользователю Astra Automation назначается роль «Системный аудитор».
- `is_system_auditor_attr` – название атрибута SAML, используемого для принятия решения о назначении пользователю Astra Automation роли «Системный аудитор».
- `is_system_auditor_value` – значение атрибута SAML, необходимое для назначения пользователю Astra Automation роли «Системный аудитор».
- `remove_system_auditors` – управление автоматическим отзывом роли «Системный аудитор» у пользователей, не соответствующим критериям:
  - `true` – роль отзывается;

- `false` – роль не отзывается.

Эти настройки имеют следующие особенности:

- Поля с `role` в названии и поля с `value` являются списками, и при обработке их значений используется логическое «ИЛИ». Это значит, что для назначения соответствующей роли пользователь SAML должен иметь *хотя бы одну* роль из списка, заданного в настройках ассоциаций.
- Если одновременно заданы значения параметров с `role` и `attr` в названии, то настройки с `attr` имеют более высокий приоритет перед `role`.
- Назначение ролей «Системный администратор» и «Системный аудитор» происходит при каждом входе пользователя в Astra Automation. Если назначить пользователю роль «Системный администратор» или «Системный аудитор» через графический интерфейс Astra Automation, а не через настройки SAML, то при следующем входе пользователя в систему назначенная роль будет отозвана.

  Чтобы предотвратить отзыв ролей «Системный администратор» или «Системный аудитор», укажите в ключах `remove_superusers` и `remove_system_auditors` соответственно значение `false`.

  Логика работы Astra Automation при отзыве и назначении ролей «Системный администратор» и «Системный аудитор» показана в таблице:

Наличие роли SAML	Наличие атрибута SAML	Наличие нужного значения атрибута SAML	Де
Нет	Нет	—	tr
Нет	Нет	—	fa
Нет	Нет	—	tr
Нет	Нет	—	fa
Да	Нет	—	tr
Да	Нет	—	fa
Да	Нет	—	tr
Да	Нет	—	fa
Нет	Да	Да	tr
Нет	Да	Да	fa
Нет	Да	Да	tr
Нет	Да	Да	fa
Нет	Да	Нет	tr
Нет	Да	Нет	fa
Нет	Да	Нет	tr
Нет	Да	Нет	fa
Нет	Да	Не задано	tr
Нет	Да	Не задано	fa
Нет	Да	Не задано	tr
Нет	Да	Не задано	fa
Да	Да	Да	tr
Да	Да	Да	fa
Да	Да	Да	tr
Да	Да	Да	fa
Да	Да	Нет	tr
Да	Да	Нет	fa
Да	Да	Нет	tr
Да	Да	Нет	fa
Да	Да	Не задано	tr
Да	Да	Не задано	fa
Да	Да	Не задано	tr
Да	Да	Не задано	fa

- Если значение ключа `remove_superusers` или `remove_system_auditors` равно `false`, соответствующая роль не будет отозвана у пользователя SAML никогда.

Пример заполнения словаря:

```
{
  "is_superuser_role": [
    "admin",
    "root"
  ],
  "is_superuser_attr": "groups",
  "is_superuser_value": [
    "admin",
    "root",
    "astra-admin"
  ],
  "is_system_auditor_role": "auditor",
  "is_system_auditor_attr": "groups",
  "is_system_auditor_value": ["auditor"]
}
```

Здесь:

- Роль «Системный администратор» назначается пользователям SAML, в отношении которых выполняется хотя бы одно из условий:
  - наличие роли `admin` или `root`;
  - наличие атрибута `groups`, хотя бы одним из значений которого является `admin`, `root` или `astra-admin`.
- Роль «Системный аудитор» назначается пользователям SAML, в отношении которых выполняется хотя бы одно из условий:
  - наличие роли `auditor`;
  - наличие атрибута `groups`, хотя бы одним из значений которого является `auditor`.
- Значения ключей `remove_superusers` и `remove_system_auditors` не заданы, поэтому используются значения по умолчанию – `true`. Это значит, что у всех пользователей SAML, не удовлетворяющих указанным требованиям, роли «Системный администратор» и «Системный аудитор» будут отозваны при первом входе в систему независимо от того, каким образом они были назначены – через настройки SAML, API или графический интерфейс Astra Automation.

TACACS+

Предупреждение

Этот провайдер аутентификации считается устаревшим и в одной из следующих версий Astra Automation станет недоступным для применения.

OIDC

Важно

Эта часть документации находится в стадии разработки.

Ролевая модель доступа

Ролевая модель доступа (*RBAC*) в Astra Automation построена как единая система управления привилегиями, действующая во всех основных компонентах платформы: Automation Controller, Private Automation Hub и Event-Driven Automation. Все они используют общие принципы проверки доступа и одну логическую модель ролей и привилегий.

Принципы управления

RBAC реализует централизованное управление и распределенное применение. Это означает, что информация о пользователях, группах и назначенных ролях хранится и управляется централизованно, но проверка привилегий выполняется там, где совершается действие – в соответствующем компоненте платформы.

Например, если пользователь:

- создает задание из шаблона в Automation Controller, то именно Automation Controller проверяет, есть ли у пользователя привилегии на запуск задания;
- публикует коллекцию в Private Automation Hub, то проверка выполняется в Private Automation Hub;
- создает правило в Event-Driven Automation, то решение принимает сам компонент Event-Driven Automation.

Такая архитектура обеспечивает надежность и масштабируемость: каждый сервис отвечает за свои проверки, но при этом все они опираются на единые данные о ролях и пользователях.

Принцип наименьших привилегий

По умолчанию доступ обычному пользователю к любому ресурсу закрыт. Пользователь получает возможность выполнять действия только после назначения соответствующей роли. Это обеспечивает предсказуемость и безопасность.

Проверка привилегий и принятие решений

Каждый запрос пользователя проходит несколько этапов:

1. Аутентификация – подтверждение личности (например, через локальную учетную запись или внешнюю систему аутентификации).
2. Определение контекста – система определяет, какие роли назначены пользователю напрямую и через команды пользователей.
3. Проверка привилегий – компонент (Automation Controller, Private Automation Hub или Event-Driven Automation) сверяет запрашиваемое действие с набором привилегий, предоставленных ролями.
4. Принятие решения – если привилегии достаточны, действие выполняется, если нет – отклоняется.
5. Аудит – результат фиксируется в потоке активности и системных журналах.

Централизация

Все компоненты используют единую базу ролей и привилегий. Это обеспечивает согласованность: изменения, внесенные администратором, автоматически распространяются на всю платформу.

Любое изменение привилегий, ролей или состава пользователей фиксируется. Администратор может просмотреть историю назначений и действий через *поток активности* в графической консоли или экспортировать данные в систему мониторинга безопасности.

Иерархия уровней доступа

RBAC в Astra Automation действует по принципу наложения уровней:

1. **Тип пользователя** задает исходный уровень доверия. Например, Astra Automation Administrator всегда имеет глобальные привилегии, а обычный пользователь – только те, которые назначены ему через роли.
2. **Роль** определяет набор привилегий для выполнения определенных действий.
3. **Привилегия** позволяет выполнять конкретное действие над конкретным ресурсом.

Система определяет эффективные привилегии пользователя динамически. Она объединяет все роли, назначенные пользователю напрямую и через команды пользователей, формируя итоговый набор привилегий. Это гарантирует, что пользователь сможет выполнять только те действия, которые ему явно разрешены.

Типы пользователей

Каждый пользователь имеет назначенный *тип*, который определяет его базовые возможности на уровне всей платформы. При создании учетной записи необходимо выбрать тип пользователя.

Доступны следующие типы:

- **Системный администратор** (Astra Automation Administrator) – может иметь полный доступ ко всем функциям и ресурсам платформы либо только к отдельным компонентам (Automation Controller, Private Automation Hub, Event-Driven Automation) в зависимости от параметров, выбранных при создании пользователя. Этот тип используется для первоначальной настройки и обслуживания системы.
- **Системный аудитор** (Astra Automation Auditor) – обладает привилегиями только на просмотр. Может просматривать любую информацию о платформе, но не может изменять ресурсы или выполнять действия.
- **Обычный пользователь** (Normal user) – не имеет привилегий по умолчанию. Его доступ ограничен назначенными ролями внутри конкретных организаций и компонентов.

Тип пользователя задает исходный уровень доверия и доступности функций. Чтобы обычный пользователь мог выполнять конкретные действия, ему требуется роль с соответствующими привилегиями.

Роли

Роли – основной механизм назначения привилегий. Каждая роль определяет, какие действия пользователь может выполнять и с какими типами ресурсов. Например, роль может разрешать изменение шаблонов заданий или публикацию коллекций.

Роли бывают двух типов:

- **Встроенные** (Built-in) – предопределенные платформой.

  Примечание

  Полный список встроенных ролей и их описание доступны в *справочном списке ролей*.

- **Собственные** (Custom) – создаются администраторами под конкретные задачи. Такие роли позволяют гибко комбинировать привилегии и точно контролировать доступ к отдельным ресурсам.

  Система позволяет создавать роль только в контексте конкретного компонента – Automation Controller, Event-Driven Automation или Private Automation Hub. Это значит, что одна роль не может содержать привилегии для управления, например, шаблонами заданий Automation Controller и средами принятия решений Event-Driven Automation.

  Внимание

  Роли этого типа не могут включать привилегии на управление командами (team) или организациями (organization), поскольку такие действия заблокированы на уровне системы RBAC. При попытке создания такой роли появится сообщение “Creating custom roles that include team permissions is disabled”.

В зависимости от области действия роль может быть двух типов:

- предоставляющая привилегии по отношению ко всем объектам определенного типа (привилегии на тип ресурсов, например на все проекты организации);
- предоставляющая привилегии по отношению к одному конкретному ресурсу, например привилегии на один проект или шаблон заданий.

Чтобы предоставить пользователю роли первого типа, необходимо в графическом интерфейсе открыть профиль организации и в списке пользователей нажать на кнопку настройки ролей конкретного пользователя. Более подробно см. в *описании окна Организации (Organizations)*.

Для назначения роли второго типа следует открыть профиль конкретного ресурса и во вкладке **Доступ** (Access) использовать встроенного помощника, следуя описанию графического интерфейса для соответствующего типа ресурса.

Привилегии

Привилегия – минимальная единица контроля доступа, позволяющая выполнять конкретное *действие* над конкретным *типом ресурса*. Например: просмотреть полномочие, создать инвентарь, удалить среду исполнения.

Изменить существующие привилегии или создать новые нельзя.

Привилегии группируются по типам ресурсов, таким как проекты, шаблоны заданий, инвентарные списки, среды исполнения и так далее.

Каждая роль объединяет одну или несколько привилегий, относящихся к одному типу содержимого. Например, чтобы пользователь мог управлять полномочиями и средами исполнения, требуется две разные роли – по одной для каждого типа ресурса.

Пользователи

Пользователю можно назначить различные роли, в том числе в разных организациях и командах.

Примечание

Чтобы пользователь считался участником *команды* или *организации*, его необходимо связать с ними через графический интерфейс.

Необходимые привилегии можно предоставить пользователю следующими способами:

- Установить флаг суперпользователя.

  Флаг суперпользователя предоставляет полный доступ ко всем ресурсам Astra Automation. Снять флаг суперпользователя с самого себя нельзя.
- Назначить пользователю роль на уровне организации.
- Сделать пользователя участником команды, обладающей необходимыми ролями.
- *Назначить* пользователю необходимую роль индивидуально.

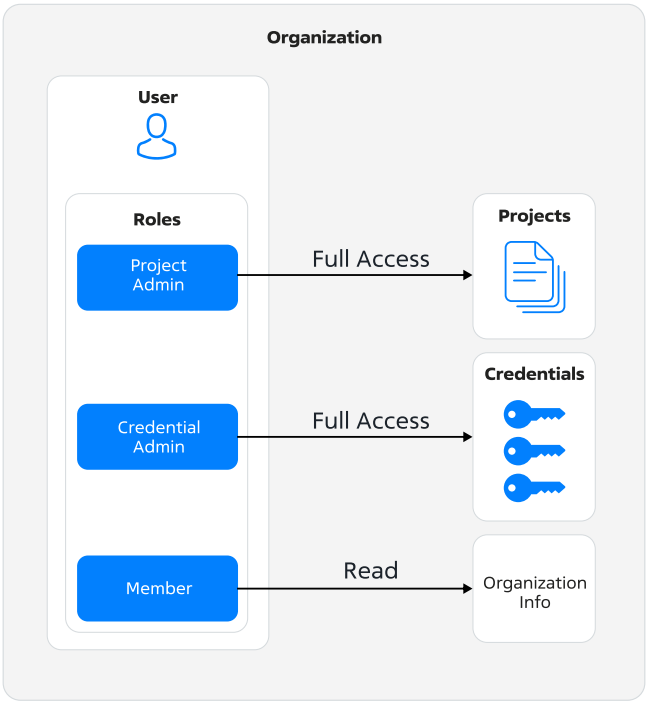
Организации

Организации используются для разграничения доступа к ресурсам платформы и настройки ограничений на количество управляемых узлов, предоставляемых действующей лицензией.

На уровне организации роли могут быть назначены отдельным пользователям и командам.

Если на уровне организации роль назначена команде, то привилегии распространяются на всех членов команды.

Пусть пользователю на уровне организации назначены роли «Администратор проекта» (Organization Project Admin), «Администратор учетных данных» (Organization Credential Admin) и «Участник» (Organization Member):



Назначенные роли предоставляют пользователю следующие привилегии на ресурсы организации:

- полный доступ ко всем проектам;
- полный доступ ко всем полномочиям;
- просмотр сведений об организации.

При развертывании платформы в ней создается организация с названием «Default». Если другие организации отсутствуют, то все создаваемые ресурсы, пользователи и команды будут принадлежать этой организации.

Связи с другими объектами

Для некоторых объектов связь с организацией обязательна. В таблице приведены все объекты, которые требуют связи с организацией.

Объект	Связь с организацией
Шаблон задания	Обязательна Принадлежит организации через проект
Проект	Обязательна
Инвентарный список	Обязательна
Управляемый узел	Обязательна Принадлежит организации через инвентарный список
Уведомления	Обязательна
Команда	Обязательна
Активация свода правил	Обязательна
Среда принятия решений	Обязательна
Поток событий	Обязательна
Полномочия для запуска сводов правил	Обязательна

Максимальное количество узлов

В настройках организации может быть задано ограничение на количество узлов. Такое ограничение позволяет решить следующие задачи:

- Контроль лимитов подписки.

  Этот параметр ограничивает количество узлов, которыми могут управлять участники организации.

- Разделение ресурсов между организациями.

  Если с контроллером работают несколько групп пользователей, управляющих разными узлами, для каждой группы создайте организацию с заданным ограничением на количество узлов. Это позволит избежать ситуации, когда одна группа пользователей не может выполнять свои задачи из-за того, что другая группа исчерпала ресурс действующей подписки.

- Планирование и масштабирование.

  Ограничение на число узлов в организации позволяет планировать ресурсы и масштабировать автоматизацию. Например, если известно, что одна организация будет расти быстрее других, можно настроить ограничение на количество узлов в ее пользу.

- Предотвращение ошибок.

  Если добавить в описание инвентаря слишком большое количество узлов, это может привести к выходу за ограничение, установленное подпиской. Ограничение на количество узлов служит защитой от таких ситуаций.

Особенности использования ограничения на количество узлов:

- Пользователи организации не смогут добавить в описание инвентаря больше узлов, чем задано этим параметром.
- Ограничение подписки действует на общее количество узлов в инвентаре и не зависит от ограничений на количество узлов в организациях. Можно задать любые желаемые ограничения для организаций или не задавать их вовсе, но в сумме количество узлов, к которым можно применять сценарии автоматизации, не может превышать ограничение, установленной в подписке.
- При использовании динамических инвентарных списков учитываются только узлы, которые были задействованы в сценариях.

- Значение настройки может быть в любое время изменено по усмотрению администратора.

Команды

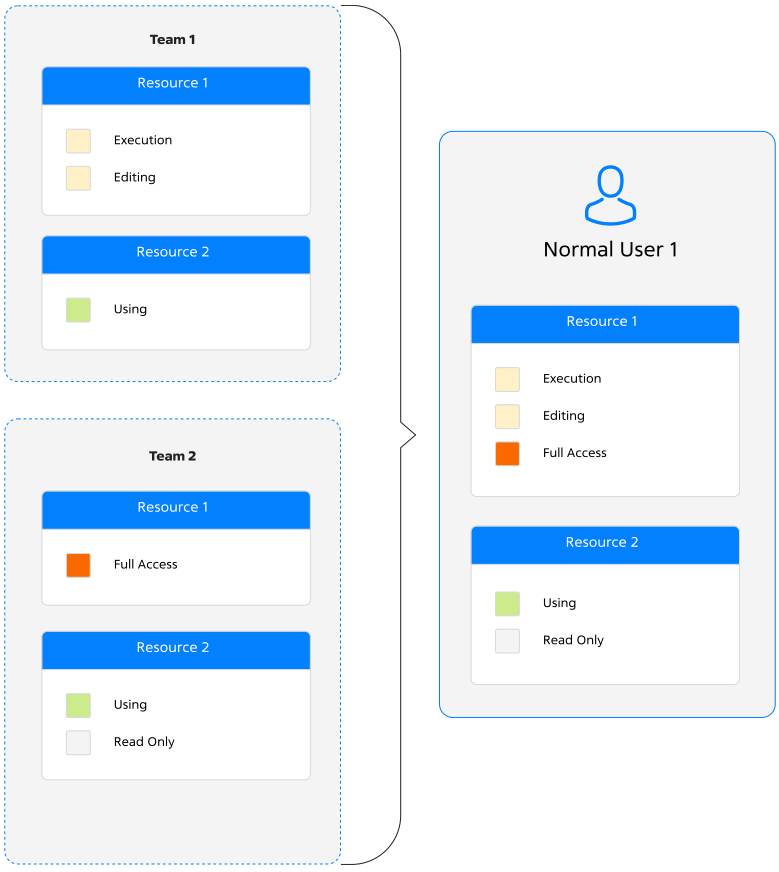
Команда пользователей существует в рамках одной организации и используется для группового назначения ролей.

Назначение роли на команду не ограничивает ее действие только на один экземпляр ресурса. Если роль действует на уровне организации, участники команды получают доступ ко всем ресурсам этого типа в рамках организации. Если же роль выдана на конкретный ресурс, доступ будет ограничен именно этим экземпляром.

Например, если команде назначена роль `ExecutionEnvironment Admin` в рамках конкретной организации, все ее участники смогут управлять всеми средами исполнения этой организации. Если та же роль назначена на конкретную среду исполнения, то участники команды смогут управлять только этим ресурсом: изменять, удалять и использовать его.

Участники команды могут принадлежать разным организациям, однако роли, назначенные команде, действуют только внутри той организации, к которой относится сама команда.

Привилегии доступа к конкретному экземпляру ресурса для обычного пользователя определяются объединением множества привилегий, предоставляемых ему как напрямую, так и через команды, в которые он входит. Принцип объединения привилегий, предоставляемых ролями, показан на схеме:



Роли команды «Team 1» предоставляют к ресурсу «Resource 1» привилегии на исполнение (execution) и изменение (editing), а к ресурсу «Resource 2» – на использование (using). Роли команды «Team 2» предоставляют к ресурсу «Resource 1» привилегии полного доступа (full access), а к ресурсу «Resource 2» – на использование и только чтение (read only).

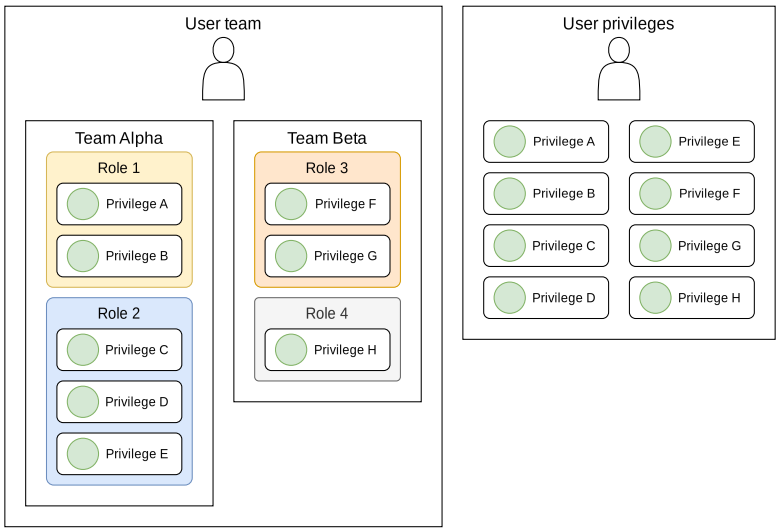
Все пользователи, входящие одновременно в команды «Team 1» и «Team 2», получают следующие привилегии на ресурсы:

- «Resource 1» – исполнение, изменение и полный доступ;
- «Resource 2» – использование и только чтение.

Пример

Рассмотрим связь пользователей, команд, ролей и привилегий на примере. Пусть имеются следующие ресурсы:

- один пользователь, не являющийся суперпользователем;
- две команды, `Alpha` и `Beta`;
- четыре роли, `Role 1` – `Role 4`;
- восемь привилегий, A – H.



Роли предоставляют следующие привилегии:

- `Role 1` – A, B.
- `Role 2` – C, D, E.
- `Role 3` – F, G;
- `Role 4` – H.

Команда `Alpha` связана с ролями `Role 1` и `Role 2`, а команда `Beta` – с ролями `Role 3` и `Role 4`.

Пользователь является участником обеих команд, а значит, он получает привилегии всех связанных с ними ролей, от A до H включительно.

Встроенные роли Astra Automation

Ниже приведены роли, определенные в компонентах платформы, и ресурсы, к которым они применяются.

Роли в Automation Controller

В компоненте Automation Controller определяются роли, управляющие шаблонами заданий, проектами, инвентарями, учетными данными и другими ресурсами. Роли контролируют возможность запуска, редактирования, делегирования привилегий и просмотра информации о ресурсах. Полный список ролей представлен в таблице ниже:

Ресурс	Роль	Описание
Credential	Credential Admin	Управление полномочиями
Credential	Credential Use	Использование полномочий
Execution Environment	ExecutionEnvironment Admin	Управление средой выполнения
Inventory	Inventory Admin	Управление инвентарным списком
Inventory	Inventory Update	Обновление инвентарного списка
Inventory	Inventory Use	Использование инвентарного списка при запуске заданий
Inventory	Inventory Adhoc	Выполнение ad-hoc команд
Instance Group	InstanceGroup Admin	Управление группой исполняющих узлов
Instance Group	InstanceGroup Use	Использование группы исполняющих узлов
Job Template	JobTemplate Admin	Управление шаблоном задания
Job Template	JobTemplate Execute	Запуск шаблона задания без возможности редактирования
Notification Template	NotificationTemplate Admin	Управление одним шаблоном уведомлений
Organization	Organization ExecutionEnvironment Admin	Управление средами выполнения внутри организации
Organization	Organization Project Admin	Управление проектами внутри организации
Organization	Organization NotificationTemplate Admin	Управление шаблонами уведомлений внутри организации
Organization	Organization JobTemplate Admin	Управление шаблонами заданий внутри организации
Organization	Organization Credential Admin	Управление полномочиями внутри организации
Organization	Organization WorkflowJobTemplate Admin	Управление шаблонами потока заданий внутри организации
Organization	Organization Inventory Admin	Управление инвентарными списками внутри организации
Organization	Organization Member	Использование назначенных ресурсов
Project	Project Admin	Управление проектом
Project	Project Update	Запуск обновления проекта (SCM Sync)
Project	Project Use	Использование проекта в шаблонах заданий
Team	Team Admin	Управление командой
Team	Team Member	Наследование привилегий, назначенных команде
Workflow Job Template	WorkflowJobTemplate Admin	Управление шаблоном потока заданий
Workflow Job Template	WorkflowJobTemplate Execute	Запуск шаблона потока заданий без возможности редактирования
Workflow Job Template	WorkflowJobTemplate Approve	Одобрение или отклонение запуска шаблона потока заданий, если включено подтверждение

Роли в Private Automation Hub

В компоненте Private Automation Hub ролевое управление регулирует доступ к коллекциям, контейнерам и пространствам имен. Роли определяют, кто может публиковать, изменять, модерировать и использовать контент. Полный список ролей представлен в таблице ниже:

Ресурс	Роль	Описание
Repository	galaxy.ansible_reposito	Управление репозиториями Ansible
Remote	galaxy.collection_remo	Управление внешними репозиториями
Execution Environment	galaxy.execution_envir	Публикация новых версий сред исполнения
Execution Environment	galaxy.execution_envir	Полный контроль над пространствами имен сред исполнения
Execution Environment	galaxy.execution_envir	Изменение существующих сред исполнения
Namespace	galaxy.collection_name	Управление пространством имен
Namespace	galaxy.collection_publis	Добавление коллекций в пространстве имен
Team	Galaxy Team Member	Наследование привилегий, назначенных команде
Team	Team Admin	Управление одной командой и наследует все назначения ролей этой команды
Team	Team Member	Наследование привилегий, назначенных команде
System	galaxy.collection_admi	Управление всеми коллекциями
System	galaxy.collection_curat	Согласование и синхронизация коллекций
System	galaxy.content_admin	Управление всеми типами контента
System	galaxy.execution_envir	Управление средами исполнения

Роли в Event-Driven Automation

В компоненте Event-Driven Automation роли управляют объектами автоматизации: активациями, проектами, окружениями принятия решений и потоками событий. Полный список ролей представлен в таблице ниже:

Ресурс	Роль	Описание
Activation	Activation Admin	Управление активации свода правил и связанными с ней ресурсами
Activation	Activation Use	Использование активации свода правил
Decision Environment	Decision Environment Admin	Управление средой принятия решений
Decision Environment	Decision Environment Use	Использование среды принятия решений
EDA Credential	EDA Credential Admin	Управление полномочиями Event-Driven Automation
EDA Credential	EDA Credential Use	Использование полномочиями Event-Driven Automation
Event Stream	Event Stream Admin	Управление потоком событий
Event Stream	Event Stream Use	Использование потока событий
Organization	Organization Admin	Полный контроль над организацией и всеми ее ресурсами
Organization	Organization Member	Участник организации с доступом к ее ресурсам
Organization	Organization Editor	Создание и изменение ресурсов внутри организации
Organization	Organization Contributor	Создание, изменение ресурсов организации, а также управление активациями rulebook
Organization	Organization Operator	Управление запуском активаций и просмотр ресурсов
Organization	Organization Viewer	Просмотр ресурсов организации
Organization	Organization Project Admin	Управление проектами внутри организации
Organization	Organization Decision Environment Admin	Управление средами принятия решений внутри организации
Organization	Organization Eda Credential Admin	Управление полномочиями Event-Driven Automation внутри организации
Organization	Organization Activation Admin	Управление активациями сводов правил внутри организации
Organization	Organization Event Stream Admin	Управление потоками событий внутри организации
Project	Project Admin	Управление проектом и связанными rulebook
Project	Project Use	Использование проекта при создании активаций сводов правил
Team	Team Admin	Управление командой
Team	Team Member	Наследование привилегий, назначенных команде

Примеры сценариев

Ниже приведены примеры типовых сценариев, демонстрирующих применение ролевой модели доступа в Astra Automation.

Назначение ролей пользователю

Например, когда новый DevOps-инженер должен запускать шаблоны, но не изменять их конфигурацию, администратор должен выполнить следующие действия:

1. *Добавить DevOps-инженера в список пользователей*.
2. *Создать команду* `DevOps`.
3. Если команда `DevOps` уже существует, *добавить туда инженера*.

4. Найти необходимый шаблон заданий и *назначить команде DevOps* роль `Job Template Execute`.
5. Проверить, что у пользователей команды `DevOps` нет привилегий на изменение шаблона.

Управление доступом к ресурсам

Этот пример демонстрирует разграничение доступа на уровне отдельных ресурсов. Например, когда QA-команде необходимо просматривать инвентарь `TestCluster` и запускать шаблон `RunTests` без привилегий на внесение изменений в шаблон, администратор должен выполнить следующие действия:

1. Создать команду `QA`.
2. *Назначить команде* `QA` роль `Inventory Read` для инвентаря `TestCluster`.
3. *Назначить команде* `QA` роль `Job Template Execute` для шаблона заданий `RunTests`.
4. Проверить, что у пользователей команды `QA` нет привилегий на внесение изменений.

Делегирование Team Admin

Следующий сценарий иллюстрирует делегирование привилегий внутри организации. Например, чтобы Team Lead мог самостоятельно управлять своей командой, например, `DevOps`, добавлять и удалять участников, не обращаясь к системному администратору, администратор должен *назначить пользователю* `Team Lead` роль `Team Admin`.

Управление журналами RBAC

Astra Automation реализует централизованную систему контроля доступа (RBAC) для всех ключевых компонентов (Automation Controller, Private Automation Hub, Event-Driven Automation) и поддерживает аудит всех операций, связанных с ролями, привилегиями и ресурсами.

Назначение

Цель регистрации сообщений в журналах – обеспечить контроль действий, связанных с безопасностью:

- соответствие требованиям информационной безопасности;
- отслеживание изменений привилегий и ролей;
- подотчетность действий администраторов;
- интеграцию с SIEM-системами[206].

Регистрация событий в журналах

Фиксируются следующие категории событий:

[206] https://ru.wikipedia.org/wiki/SIEM

Категория	Журналируемые события
Пользователи	Создание, удаление, изменение профиля, вход и выход из системы
Роли	Назначение и отзыв ролей, создание кастомных ролей
Команды	Добавление и удаление пользователей из команды
Организации	Назначение ролей и изменение параметров организации
Ресурсы	Выдача и отзыв привилегий на Job Template, Inventory, Credentials и другие ресурсы
Собственные (custom) роли	Создание, редактирование и удаление ролей с произвольным набором привилегий
Аутентификация	Входы, выходы и ошибки при попытках аутентификации
Журналы	Экспорт журналов и изменение каналов журналирования

Просмотр журналов

Система журналирования Astra Automation обеспечивает полное отслеживание всех операций, связанных с управлением доступом, назначением ролей и изменением конфигурации объектов. Журналы служат инструментом аудита и контроля безопасности, позволяя анализировать действия пользователей и состояние компонентов платформы. Все события записываются в поток активности (Activity Stream) и, при необходимости, могут пересылаться во внешние системы мониторинга.

Activity Streams

Activity Streams – это встроенный механизм фиксации всех действий, выполняемых пользователями через графический интерфейс или API. Он обеспечивает прозрачность всех изменений и позволяет анализировать историю операций на уровне платформы, организации, команды или конкретного ресурса.

Каждый компонент Astra Automation ведет собственный поток активности, доступный через REST API:

- **Platform Gateway**

  Фиксирует изменения, происходящие на уровне всей платформы: создание и модификацию организаций, пользователей, подключение компонентов (Automation Controller, Private Automation Hub, Event-Driven Automation). Используется для централизованного просмотра и анализа событий из всех подсистем.

  Точка API: `/api/gateway/v1/activitystream/`

- **Automation Controller**

  Отслеживает действия пользователей с объектами автоматизации: шаблонами заданий, инвентарями, проектами, учетными данными, командами и расписаниями. Поддерживает фильтры: `object_type` (тип объекта), `user` (пользователь), `action` (действие), `timestamp` (время выполнения). Эти параметры позволяют быстро находить нужные события при анализе журналов.

  Точка API: `/api/v2/activity_stream/`

API

Для анализа активности пользователей можно использовать REST API. Точки доступа API позволяют выполнять фильтрацию, экспорт и интеграцию журналов с внешними средствами анализа. Доступны следующие запросы:

```
GET /api/v2/activity_stream/?page_size=1000
GET /api/gateway/v1/activitystream/?page_size=1000
```

Доступ к журналам

Доступ к просмотру журналов зависит от уровня ролей пользователя. Администраторы и аудиторы имеют полный доступ ко всем записям, а обычные пользователи видят только свои действия. Такое разграничение исключает просмотр чужих событий и обеспечивает изоляцию данных между организациями.

Роль	Уровень доступа
System Administrator	Полный просмотр, экспорт и настройка параметров журналирования
System Auditor	Просмотр всех журналов системы
Organization Admin	Просмотр журналов организации
Team Admin	Просмотр событий, связанных с ресурсами команды
Пользователь	Просмотр только собственных действий

Защита и целостность журналов

Система ведения журналов должна гарантировать неизменность и достоверность записей. Нарушение целостности журналов может привести к сокрытию следов инцидентов, поэтому рекомендуется применять следующие меры защиты:

- Файлы журналов должны принадлежать пользователю и группе `awx:awx`, чтобы обеспечить корректный доступ сервисов. Это обеспечивает корректный доступ всех сервисов, выполняемых от имени пользователя awx, к операциям записи и чтения журналов.
- Задайте для файлов журналов права доступа `chmod 600`. В этом режиме только владелец имеет право читать и изменять файлы. Такая настройка предотвращает просмотр журналов посторонними и снижает риск утечки чувствительной информации.
- Ротацию журналов выполняйте с помощью `logrotate`, чтобы предотвратить переполнение хранилища и обеспечить регулярное обновление файлов.
- Для защиты от удаления или подмены включите атрибут неизменяемости (`chattr +a`).
- Для контроля целостности используйте инструменты AIDE[207], Tripwire[208] или мандатный контроль каталогов в Astra Linux[209].

Экспорт и интеграция с системами SIEM

Для централизованного анализа событий безопасности журналы платформы Astra Automation можно экспортировать во внешние системы класса **SIEM** (Security Information and Event Management). Такие системы собирают и коррелируют события из разных источников, чтобы выявлять инциденты, несанкционированные изменения привилегий и подозрительные действия пользователей.

Интеграция с системами SIEM позволяет выполнять следующие действия:

- получать единое представление о событиях безопасности во всех компонентах Astra Automation;
- отслеживать изменения ролей, назначений и попытки аутентификации в реальном времени;
- формировать отчеты и уведомления о подозрительных действиях администраторов;
- выполнять долгосрочный аудит, даже при удалении или ротации локальных журналов.

Интеграция осуществляется следующими способами:

- в *графическом интерфейсе*;

[207] https://en.wikipedia.org/wiki/Advanced_Intrusion_Detection_Environment
[208] https://ru.wikipedia.org/wiki/Tripwire
[209] https://wiki.astralinux.ru/pages/viewpage.action?pageId=153486002

- через API.

Для интеграции через API возможно прямое подключение к потоку активности (Activity Stream). Пример запроса:

```
GET /api/gateway/v1/activitystream/?page_size=1000&object_type=role
```

Этот способ позволяет собирать события напрямую, без использования Syslog-агентов, и обрабатывать их в формате JSON средствами системы SIEM.

При необходимости данные журналов можно фильтровать по типу событий, пользователю, действию и ресурсу. Рекомендуется пересылать следующие категории событий:

- аутентификация пользователей;
- назначение и отзыв ролей;
- создание или изменение собственных (custom) ролей;
- операции с организациями и командами;
- попытки доступа к неразрешенным ресурсам.

Особенности собственных ролей

Собственные (custom) роли позволяют администратору создавать наборы привилегий для нестандартных сценариев использования платформы. Поскольку такие роли напрямую влияют на модель безопасности, система фиксирует все действия с ними в журнале активности. Каждое создание, изменение или удаление роли фиксируется в потоке активности (Activity Stream) и доступно для анализа администраторам и аудиторам.

Отслеживание изменений позволяет:

- контролировать, кто создает или изменяет роли, и в какой момент это происходит;
- выявлять несанкционированные модификации структуры RBAC;
- проводить аудит изменений при настройке делегирования привилегий в организациях.

Для поиска соответствующих событий в Activity Stream можно использовать API-запрос с фильтрацией по типу ресурса и операции:

```
GET /api/v2/activity_stream/?object_type=role&operation=update&page_size=1000
```

В ответе API будут возвращены записи обо всех действиях, связанных с ролями. Чтобы выделить именно пользовательские роли, анализируйте элементы, где объект имеет атрибут `custom`.

Пример типичных событий:

- создание новой роли с ограниченными привилегиями выполнения шаблонов заданий;
- изменение привилегий в существующей собственной роли;
- удаление роли, ранее назначенной команде или пользователю.

Рекомендуется регулярно проверять Activity Stream на наличие подобных операций, особенно в системах с делегированным управлением ролями. Такой контроль помогает своевременно выявлять несогласованные изменения в модели доступа.

Общие рекомендации

Для обеспечения надежного аудита и централизации данных журналирования придерживайтесь следующих рекомендаций:

- используйте контроль целостности журналов с помощью AIDE или аналогичных инструментов;
- применяйте фильтры по собственным ролям и пользователям в интерфейсе потока активности (Activity Stream) для быстрой аналитики изменений RBAC;

- регулярно экспортируйте поток активности через API и сохраняйте результаты в централизованное хранилище событий;
- храните архивные копии журналов в изолированном каталоге с ограниченным доступом и контролем привилегий.

Выполнение этих рекомендаций повышает уровень прозрачности системы, снижает риск несанкционированных изменений и облегчает проведение аудита безопасности.

Графический интерфейс

Раздел **Управление доступом** (Access Management) предназначен для централизованного управления пользователями, организациями и их правами в Astra Automation. Он объединяет инструменты аутентификации, контроля доступа и разграничения прав, обеспечивая гибкую и безопасную модель взаимодействия с платформой.

В этом разделе администраторы могут выполнять следующие действия:

- выбирать и настраивать методы аутентификации пользователей (локальные учётные записи, LDAP, SSO и др.);
- создавать организации и управлять их настройками;
- формировать команды внутри организаций для удобного назначения ролей и прав;
- добавлять, изменять и удалять пользователей;
- назначать роли и управлять уровнями доступа;
- регистрировать и контролировать использование внешних приложений, интегрируемых по OAuth 2.0.

Ниже представлены основные элементы интерфейса раздела **Управление доступом**.

Методы аутентификации

Раздел **Методы аутентификации** (Authentication Methods) используется для настройки способов входа в систему. Поддерживаются следующие механизмы: локальные учетные записи, внешние провайдеры (например, LDAP), а также интеграция с системами единого входа (SSO). Это обеспечивает гибкость и адаптацию под корпоративные стандарты безопасности.

Организации

В разделе **Организации** (Organizations) создаются изолированные рабочие области для пользователей и ресурсов. *Организации* упрощают управление инфраструктурой, в которой присутствует множество подразделений компании, и позволяют распределять доступ к заданиям, проектам и инфраструктуре между этими подразделениями.

Команды

Раздел **Команды** (Teams) позволяет объединять пользователей в группы внутри организаций. *Команды* применяются для удобного управления назначением ролей и делегирования прав доступа.

Пользователи

Раздел **Пользователи** (Users) используется для управления учетными записями. Здесь можно создавать *учетные записи*, изменять их данные, управлять активностью и правами, а также удалять.

Роли

В разделе **Роли** (Roles) администраторы назначают и настраивают права пользователей и команд. *Роли* определяют доступ к различным объектам и операциям в системе, обеспечивая гибкую модель разграничения полномочий.

OAuth-приложения

Раздел **OAuth-приложения** (OAuth Applications) используется для регистрации сторонних приложений, которым требуется доступ к API платформы. OAuth 2.0 позволяет безопасно делегировать полномочия и ограничивать их заданным набором разрешений.

Методы аутентификации

Окно **Методы аутентификации** (Authentication Methods) используется для конфигурации каналов идентификации пользователей. С его помощью администратор может подключать и настраивать внешние провайдеры.

Для перехода к окну **Методы аутентификации** выберите на панели навигации *Управление доступом* ▸ *Методы аутентификации* (*Access Management* ▸ *Authentication Methods*).

Таблица методов аутентификации

Внешний вид окна **Методы аутентификации** (Authentication Methods) представлен на схеме:

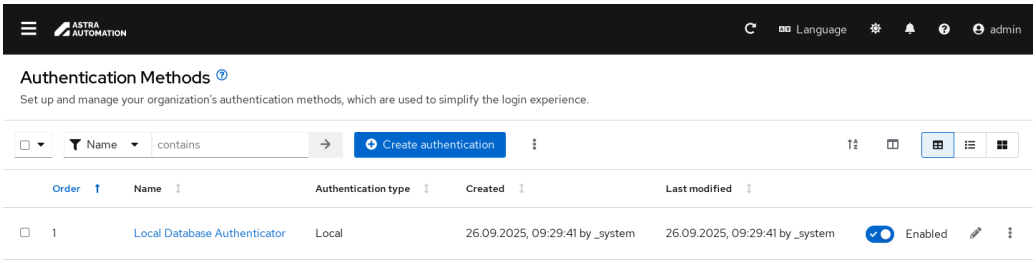


Таблица методов аутентификации состоит из столбцов согласно настройке с помощью кнопки *Настроить столбцы* (Manage columns). Полный список столбцов:

- Флаги для выбора нескольких записей.
- **Порядок** (Order) – числовое значение, определяющее приоритет метода аутентификации. Методы с меньшим значением выполняются раньше при проверке учетных данных пользователя.
- **Название** (Name) – название метода аутентификации.
- **Тип аутентификации** (Authentication type) – тип механизма проверки подлинности, например, Local, LDAP или SAML.
- **Дата создания** (Created) – дата и время создания метода аутентификации.
- **Последнее изменение** (Last modified) – дата и время последнего изменения метода аутентификации.
- Флаг переключения активности метода аутентификации.
- Кнопки вызова часто выполняемых действий:
  - редактирование метода аутентификации;
  - удаление метода аутентификации.

Создание метода аутентификации

Для создания метода аутентификации выполните следующие действия:

1. В окне **Методы аутентификации** (Authentication Methods) нажмите кнопку *Создать метод аутентификации* (Create authentication).
2. В открывшемся окне выберите необходимый тип метода аутентификации в поле **Тип аутентификации** (Authentication type) и нажмите кнопку *Далее* (Next).

   Примечание

   Список доступных методов и их описание см. в документе *Методы аутентификации*.

3. Заполните поля во вкладке **Данные аутентификации** (Authentication details), соответствующие выбранному методу аутентификации.

Controller admin

- **Название** (Name) – название метода аутентификации. Отображается в списке методов и при выборе источника.
- **Автоматически мигрировать пользователей из** (Auto migrate users from) – позволяет выбрать существующий метод аутентификации, из которого следует автоматически перенести пользователей.
- **Additional Authenticator Fields** – используется для задания дополнительных параметров в формате YAML или JSON.
- **Опции** (Options):
  - **Включенный** (Enabled) – если опция включена, метод будет активирован после его создания;
  - **Создавать объекты** (Create objects) – если опция включена, аутентификатору разрешено создавать объекты (пользователей, команды, организации);
  - **Удалить пользователей** (Remove users) – если опция включена, пользователь проходящий аутентификацию с помощью этого источника, будет удален из всех других групп, в которые он был добавлен ранее.

Azuread

- **Название** (Name) – название метода аутентификации.
- **Автоматически мигрировать пользователей из** (Auto migrate users from) – позволяет выбрать существующий метод аутентификации, из которого следует автоматически перенести пользователей.
- **OIDC Key** – **Идентификатор клиента** (Client ID), выданный при регистрации приложения в Azure AD (Microsoft Entra ID).
- **OIDC Secret** – **Секрет клиента** (Client Secret), соответствующий ключу OIDC.
- **Groups Claim** – ключ JSON, содержащий список групп пользователя, получаемый от провайдера аутентификации. Используется для сопоставления групп и ролей платформы с группами пользователя.
- **Additional Authenticator Fields** – используется для задания дополнительных параметров в формате YAML или JSON.
- **Опции** (Options):
  - **Включенный** (Enabled) – если опция включена, метод будет активирован после его создания;
  - **Создавать объекты** (Create objects) – если опция включена, аутентификатору разрешено создавать объекты (пользователей, команды, организации);

  - **Удалить пользователей** (Remove users) – если опция включена, пользователь проходящий аутентификацию с помощью этого источника, будет удален из всех других групп, в которые он был добавлен ранее.

GitHub

- **Название** (Name) – название метода аутентификации.
- **Автоматически мигрировать пользователей из** (Auto migrate users from) – позволяет выбрать существующий метод аутентификации, из которого следует автоматически перенести пользователей.
- **GitHub OAuth2 Key** – **Идентификатор клиента** (Client ID), выданный при регистрации OAuth-приложения в GitHub.
- **GitHub OAuth2 Secret** – **Секрет клиента** (Client Secret), соответствующий указанному ключу.
- **Additional Authenticator Fields** – используется для задания дополнительных параметров в формате YAML или JSON.
- **Опции** (Options):
  - **Включенный** (Enabled) – если опция включена, метод будет активирован после его создания;
  - **Создавать объекты** (Create objects) – если опция включена, аутентификатору разрешено создавать объекты (пользователей, команды, организации);
  - **Удалить пользователей** (Remove users) – если опция включена, пользователь проходящий аутентификацию с помощью этого источника, будет удален из всех других групп, в которые он был добавлен ранее.

GitHub enterprise

- **Название** (Name) – название метода аутентификации.
- **Автоматически мигрировать пользователей из** (Auto migrate users from) – позволяет выбрать существующий метод аутентификации, из которого следует автоматически перенести пользователей.
- **GitHub OAuth2 Key** – **Идентификатор клиента** (Client ID), выданный при регистрации OAuth-приложения в GitHub.
- **GitHub OAuth2 Secret** – **Секрет клиента** (Client Secret), соответствующий указанному ключу.
- **Base URL** – базовый URL экземпляра GitHub, например, `https://github.example.com`.
- **GitHub OAuth2 Enterprise API URL** – URL API OAuth2 корпоративного экземпляра GitHub, например, `https://github.example.com/api/v3`.
- **Additional Authenticator Fields** – используется для задания дополнительных параметров в формате YAML или JSON.
- **Опции** (Options):
  - **Включенный** (Enabled) – если опция включена, метод будет активирован после его создания;
  - **Создавать объекты** (Create objects) – если опция включена, аутентификатору разрешено создавать объекты (пользователей, команды, организации);
  - **Удалить пользователей** (Remove users) – если опция включена, пользователь проходящий аутентификацию с помощью этого источника, будет удален из всех других групп, в которые он был добавлен ранее.

GitHub enterprise organization

- **Название** (Name) – название метода аутентификации.
- **Автоматически мигрировать пользователей из** (Auto migrate users from) – позволяет выбрать существующий метод аутентификации, из которого следует автоматически перенести пользователей.
- **GitHub OAuth2 Key** – **Идентификатор клиента** (Client ID), выданный при регистрации OAuth-приложения в GitHub.
- **GitHub OAuth2 Secret** – **Секрет клиента** (Client Secret), соответствующий указанному ключу.
- **Base URL** – базовый URL экземпляра GitHub, например, `https://github.example.com`.
- **GitHub OAuth2 Enterprise API URL** – URL API OAuth2 корпоративного экземпляра GitHub, например, `https://github.example.com/api/v3`.
- **GitHub OAuth2 Enterprise Org Name** – название организации в экземпляре GitHub Enterprise, пользователи которой могут проходить аутентификацию через данный метод.
- **Additional Authenticator Fields** – используется для задания дополнительных параметров в формате YAML или JSON.
- **Опции** (Options):
  - **Включенный** (Enabled) – если опция включена, метод будет активирован после его создания;
  - **Создавать объекты** (Create objects) – если опция включена, аутентификатору разрешено создавать объекты (пользователей, команды, организации);
  - **Удалить пользователей** (Remove users) – если опция включена, пользователь проходящий аутентификацию с помощью этого источника, будет удален из всех других групп, в которые он был добавлен ранее.

GitHub enterprise team

- **Название** (Name) – название метода аутентификации.
- **Автоматически мигрировать пользователей из** (Auto migrate users from) – позволяет выбрать существующий метод аутентификации, из которого следует автоматически перенести пользователей.
- **GitHub OAuth2 Key** – **Идентификатор клиента** (Client ID), выданный при регистрации OAuth-приложения в GitHub.
- **GitHub OAuth2 Secret** – **Секрет клиента** (Client Secret), соответствующий указанному ключу.
- **Base URL** – базовый URL экземпляра GitHub, например, `https://github.example.com`.
- **GitHub OAuth2 Enterprise API URL** – URL API OAuth2 корпоративного экземпляра GitHub, например, `https://github.example.com/api/v3`.
- **GitHub OAuth2 team ID** – **Идентификатор команды** (Team ID) в GitHub Enterprise, пользователи которой могут проходить аутентификацию.
- **Additional Authenticator Fields** – используется для задания дополнительных параметров в формате YAML или JSON.
- **Опции** (Options):
  - **Включенный** (Enabled) – если опция включена, метод будет активирован после его создания;
  - **Создавать объекты** (Create objects) – если опция включена, аутентификатору разрешено создавать объекты (пользователей, команды, организации);

  - **Удалить пользователей** (Remove users) – если опция включена, пользователь проходящий аутентификацию с помощью этого источника, будет удален из всех других групп, в которые он был добавлен ранее.

GitHub organization

- **Название** (Name) – название метода аутентификации.
- **Автоматически мигрировать пользователей из** (Auto migrate users from) – позволяет выбрать существующий метод аутентификации, из которого следует автоматически перенести пользователей.
- **GitHub OAuth2 Key** – **Идентификатор клиента** (Client ID), выданный при регистрации OAuth-приложения в GitHub.
- **GitHub OAuth2 Secret** – **Секрет клиента** (Client Secret), соответствующий указанному ключу.
- **GitHub OAuth2 Organization Name** – название организации на GitHub.com, пользователи которой могут входить в систему через данный метод.
- **Additional Authenticator Fields** – используется для задания дополнительных параметров в формате YAML или JSON.
- **GitHub OAuth2 Scope** – список разрешений (scope), запрашиваемых у GitHub при аутентификации пользователя. Значение по умолчанию: `read:org`.
- **Опции** (Options):
  - **Включенный** (Enabled) – если опция включена, метод будет активирован после его создания;
  - **Создавать объекты** (Create objects) – если опция включена, аутентификатору разрешено создавать объекты (пользователей, команды, организации);
  - **Удалить пользователей** (Remove users) – если опция включена, пользователь проходящий аутентификацию с помощью этого источника, будет удален из всех других групп, в которые он был добавлен ранее.

GitHub team

- **Название** (Name) – название метода аутентификации.
- **Автоматически мигрировать пользователей из** (Auto migrate users from) – позволяет выбрать существующий метод аутентификации, из которого следует автоматически перенести пользователей.
- **GitHub OAuth2 Key** – **Идентификатор клиента** (Client ID), выданный при регистрации OAuth-приложения в GitHub.
- **GitHub OAuth2 Secret** – **Секрет клиента** (Client Secret), соответствующий указанному ключу.
- **GitHub OAuth2 Team ID** – идентификатор команды (**Team ID**) на GitHub.com, пользователи которой смогут входить в систему.
- **Additional Authenticator Fields** – используется для задания дополнительных параметров в формате YAML или JSON.
- **GitHub OAuth2 Scope** – список разрешений (scope), запрашиваемых у GitHub при аутентификации пользователя. Значение по умолчанию: `read:org`.
- **Опции** (Options):
  - **Включенный** (Enabled) – если опция включена, метод будет активирован после его создания;
  - **Создавать объекты** (Create objects) – если опция включена, аутентификатору разрешено создавать объекты (пользователей, команды, организации);

  - **Удалить пользователей** (Remove users) – если опция включена, пользователь проходящий аутентификацию с помощью этого источника, будет удален из всех других групп, в которые он был добавлен ранее.

Google OAuth

- **Название** (Name) – название метода аутентификации.
- **Автоматически мигрировать пользователей из** (Auto migrate users from) – позволяет выбрать существующий метод аутентификации, из которого следует автоматически перенести пользователей.
- **Google OAuth2 Key** – **Идентификатор клиента** (Client ID) приложения из Google Cloud Console.
- **Google OAuth2 Secret** – **Секрет клиента** (Client Secret), соответствующий указанному ключу.
- **Authorization URL** – адрес авторизации Google, например, `https://accounts.google.com/o/oauth2/auth`.
- **Access Token URL** – адрес для получения токена доступа, например, `https://oauth2.googleapis.com/token`.
- **Access Token Method** – метод HTTP для получения токена (обычно P0ST).
- **Revoke Token URL** – адрес для аннулирования токена, например, `https://oauth2.googleapis.com/revoke`.
- **Revoke Token Method** – метод HTTP для аннулирования токена (обычно P0ST).
- **Redirect State** – если параметр включен, платформа сохраняет и проверяет состояние при редиректе (рекомендуется для защиты от атак CSRF).
- **Additional Authenticator Fields** – используется для задания дополнительных параметров в формате YAML или JSON.
- **List of OAuth2 Scope(s)** – список разрешений (scope), запрашиваемых у провайдера при аутентификации пользователя. Значение по умолчанию: `["openid", "email", "profile"]`.
- **Опции** (Options):
  - **Включенный** (Enabled) – если опция включена, метод будет активирован после его создания;
  - **Создавать объекты** (Create objects) – если опция включена, аутентификатору разрешено создавать объекты (пользователей, команды, организации);
  - **Удалить пользователей** (Remove users) – если опция включена, пользователь проходящий аутентификацию с помощью этого источника, будет удален из всех других групп, в которые он был добавлен ранее.

keycloak

- **Название** (Name) – название метода аутентификации.
- **Автоматически мигрировать пользователей из** (Auto migrate users from) – позволяет выбрать существующий метод аутентификации, из которого следует автоматически перенести пользователей.
- **Keycloak Access Token URL** – URL получения токена доступа.
- **Keycloak Provider URL** – базовый URL Keycloak, например `https://keycloak.example.com/realms/myrealm`.
- **Keycloak OIDC Key** – **Идентификатор клиента** (Client ID) в Keycloak.
- **Keycloak Public Key** – открытый ключ для проверки подписи токенов.
- **Keycloak OIDC Secret** – **Секрет клиента** (Client Secret), соответствующий указанному ключу.

- **Groups Claim** – ключ JSON, содержащий список групп пользователя, получаемый от провайдера аутентификации. Используется для сопоставления групп и ролей платформы с группами пользователя.
- **Additional Authenticator Fields** – используется для задания дополнительных параметров в формате YAML или JSON.
- **Опции** (Options):
  - **Включенный** (Enabled) – если опция включена, метод будет активирован после его создания;
  - **Создавать объекты** (Create objects) – если опция включена, аутентификатору разрешено создавать объекты (пользователей, команды, организации);
  - **Удалить пользователей** (Remove users) – если опция включена, пользователь проходящий аутентификацию с помощью этого источника, будет удален из всех других групп, в которые он был добавлен ранее.

LDAP

- **Название** (Name) – название метода аутентификации.
- **Автоматически мигрировать пользователей из** (Auto migrate users from) – позволяет выбрать существующий метод аутентификации, из которого следует автоматически перенести пользователей.
- **LDAP Server URI** – адрес сервера LDAP, например, `ldap://ldap.example.com` или `ldaps://ldap.example.com:636`.
- **LDAP Bind DN** – отличительное наименование пользователя (Distinguished Name), используемого для авторизации в каталоге LDAP, например, `CN=AutomationControllerAdmin,CN=users,DC=example,DC=com`.

  Важно

  Значение в этом поле должно точно совпадать со значением атрибута пользователя «Distinguished Name bind».

- **LDAP Bind Password** – пароль для учетной записи, указанной в поле **LDAP Bind DN**.
- **LDAP Group Type** – тип групп в каталоге, например, `GroupOfNamesType`, `PosixGroupType`, `ActiveDirectoryGroupType` и так далее.
- **LDAP User DN Template** – шаблон DN пользователя, применяемый при формировании пути к учетной записи, например, `uid={0},ou=people,dc=example,dc=com`.
- **LDAP Start TLS** – использование защищенное соединение TLS, если LDAP не использует SSL.
- **Additional Authenticator Fields** – используется для задания дополнительных параметров в формате YAML или JSON.
- **LDAP Connection Options** – параметры подключения к серверу LDAP.

  Пример

  ```
  OPT_REFERRALS: 0
  OPT_NETWORK_TIMEOUT: 30
  ```

- **LDAP Group Type Parameters** – параметры, определяющие способ поиска членов групп.

Пример

```
name_attr: cn
member_attr: member
```

- **LDAP Group Search** – параметры поиска групп в LDAP.

Пример

```
base_dn: "ou=groups,dc=example,dc=com"
filter: "(objectClass=groupOfNames)"
```

- **LDAP User Attribute Map** – сопоставление атрибутов LDAP полям пользователя в платформе.

Пример

```
first_name: givenName
last_name: sn
email: mail
```

- **LDAP User Search** – параметры поиска пользователей.

Пример

```
[
  "DC=example,DC=com",
  "SCOPE_SUBTREE",
  "(cn=%(user)s)"
]
```

Примечание

При работе с Microsoft Active Directory используйте атрибут sAMAccountName:

```
[
  "DC=example,DC=com",
  "SCOPE_SUBTREE",
  "(sAMAccountName=$(user)s)"
]
```

- **Опции** (Options):
  - **Включенный** (Enabled) – если опция включена, метод будет активирован после его создания;
  - **Создавать объекты** (Create objects) – если опция включена, аутентификатору разрешено создавать объекты (пользователей, команды, организации);
  - **Удалить пользователей** (Remove users) – если опция включена, пользователь проходящий аутентификацию с помощью этого источника, будет удален из всех других групп, в которые он был добавлен ранее.

Local

- **Название** (Name) – название метода аутентификации.
- **Автоматически мигрировать пользователей из** (Auto migrate users from) – позволяет выбрать существующий метод аутентификации, из которого следует автоматически перенести пользователей.
- **Additional Authenticator Fields** – используется для задания дополнительных параметров в формате YAML или JSON.

- **Опции** (Options):
  - **Включенный** (Enabled) – если опция включена, метод будет активирован после его создания;
  - **Создавать объекты** (Create objects) – если опция включена, аутентификатору разрешено создавать объекты (пользователей, команды, организации);
  - **Удалить пользователей** (Remove users) – если опция включена, пользователь проходящий аутентификацию с помощью этого источника, будет удален из всех других групп, в которые он был добавлен ранее.

Generic OIDC

- **Название** (Name) – название метода аутентификации.
- **Автоматически мигрировать пользователей из** (Auto migrate users from) – позволяет выбрать существующий метод аутентификации, из которого следует автоматически перенести пользователей.
- **OIDC Provider URL** – адрес провайдера OpenID Connect, например `https://idp.example.com`.
- **OIDC Key** – **Идентификатор клиента** (Client ID) OpenID Connect.
- **OIDC Secret** – **Секрет клиента** (Client Secret) OpenID Connect.
- **Access Token URL** – URL для получения токена доступа (access token).
- **Access Token Method** – HTTP-метод получения токена (P0ST или GET).
- **Authorization URL** – URL авторизации пользователя.
- **ID Key** – параметр, определяющий идентификатор пользователя (часто sub).
- **ID Token Issuer** – URL издателя (issuer) ID-токена.
- **JWKS URL** – ссылка на JSON Web Key Set (JWKS) с открытыми ключами для проверки подписи токенов.
- **OIDC Public Key** – открытый ключ для проверки подписи токенов (используется, если JWKS URL недоступен).
- **Revoke Token URL** – URL для аннулирования токена.
- **Revoke Token Method** – метод HTTP, используемый при отзыве токена (P0ST или GET).
- **Response Type** – тип ответа при аутентификации (обычно code).
- **Token Endpoint Auth Method** – метод аутентификации клиента при обращении к token-endpoint, например, `client_secret_basic` или `client_secret_post`.
- **Userinfo URL** – URL для получения информации о пользователе.
- **Username Key** – ключ JSON, используемый для извлечения названия учетной записи пользователя из ID-токена или ответа userinfo.
- **Groups Claim** – ключ JSON, содержащий список групп пользователя, получаемый от провайдера аутентификации. Используется для сопоставления групп и ролей платформы с группами пользователя.
- **Verify OIDC Provider Certificate** – активация проверки сертификата провайдера.
- **Redirect State** – если параметр включен, платформа сохраняет и проверяет состояние при редиректе (рекомендуется для защиты от атак CSRF).
- **Additional Authenticator Fields** – используется для задания дополнительных параметров в формате YAML или JSON.
- **OIDC JWT Algorithm(s)** – список допустимых алгоритмов подписи токенов JWT.

Пример

```
- RS256
- HS256
```

- **OIDC JWT Decode Options** – параметры проверки и декодирования токенов JWT.

Пример

```
verify_signature: true
verify_aud: true
```

- **List of OAuth2 Scope(s)** – список разрешений (scope), запрашиваемых у провайдера при аутентификации пользователя. Значение по умолчанию: `["openid", "email", "profile"]`.
- **Опции** (Options):
  - **Включенный** (Enabled) – если опция включена, метод будет активирован после его создания;
  - **Создавать объекты** (Create objects) – если опция включена, аутентификатору разрешено создавать объекты (пользователей, команды, организации);
  - **Удалить пользователей** (Remove users) – если опция включена, пользователь проходящий аутентификацию с помощью этого источника, будет удален из всех других групп, в которые он был добавлен ранее.

Radius

- **Название** (Name) – название метода аутентификации.
- **Автоматически мигрировать пользователей из** (Auto migrate users from) – позволяет выбрать существующий метод аутентификации, из которого следует автоматически перенести пользователей.
- **Hostname of RADIUS Server** – адрес узла или IP-адрес сервера RADIUS, к которому выполняется обращение для проверки учетных данных пользователей.
- **Shared secret for authenticating to RADIUS server** – общий секрет, используемый для аутентификации на сервере RADIUS.
- **Additional Authenticator Fields** – используется для задания дополнительных параметров в формате YAML или JSON.
- **Опции** (Options):
  - **Включенный** (Enabled) – если опция включена, метод будет активирован после его создания;
  - **Создавать объекты** (Create objects) – если опция включена, аутентификатору разрешено создавать объекты (пользователей, команды, организации);
  - **Удалить пользователей** (Remove users) – если опция включена, пользователь проходящий аутентификацию с помощью этого источника, будет удален из всех других групп, в которые он был добавлен ранее.

SAML

- **Название** (Name) – название метода аутентификации.
- **Автоматически мигрировать пользователей из** (Auto migrate users from) – позволяет выбрать существующий метод аутентификации, из которого следует автоматически перенести пользователей.
- **SAML Service Provider Entity ID** – **Уникальный идентификатор** (SP Entity ID), под которым платформа зарегистрирована у поставщика удостоверений.

- **SAML Service Provider Public Certificate** – публичный сертификат, применяемый для подписи запросов к поставщику услуг SAML (SP).
- **IdP Login URL** – URL для входа на стороне IdP.
- **IdP Public Cert** – публичный сертификат IdP, используемый для проверки подписанных ответов.
- **Entity ID** – идентификатор объекта IdP.
- **Groups** – атрибут утверждения SAML, содержащий список групп пользователя.
- **User Email** – атрибут утверждения SAML, содержащий адрес электронной почты пользователя.
- **Username** – атрибут утверждения SAML, содержащий названия пользователя.
- **User Last Name** – атрибут утверждения SAML, содержащий фамилию пользователя.
- **User First Name** – атрибут утверждения SAML, содержащий имя пользователя.
- **User Permanent ID** – атрибут утверждения SAML, содержащий постоянный идентификатор пользователя.
- **SAML Assertion Consumer Service (ACS) URL** – адрес точки приема утверждений SAML, на который поставщик удостоверений (IdP) отправляет ответ после успешной аутентификации.
- **SAML Service Provider Private Key** – приватный ключ для расшифровки зашифрованных утверждений и подписи запросов.
- **Additional Authenticator Fields** – используется для задания дополнительных параметров в формате YAML или JSON.
- **SAML Service Provider Organization Info** – сведения о провайдере услуг, включая URL-адрес, отображаемое и внутреннее названия приложения.
- **SAML Service Provider Technical Contact** – контактные данные технического представителя провайдера услуг.
- **SAML Service Provider Support Contact** – контактные данные службы поддержки провайдера услуг.
- **SAML Service Provider extra configuration data** – дополнительные параметры конфигурации провайдера услуг, передаваемые поставщику удостоверений.
- **SAML Security Config** – параметры безопасности соединения SAML, включая алгоритмы подписи и требования к шифрованию.
- **SAML IDP to extra_data attribute mapping** – сопоставление атрибутов поставщика удостоверений (IdP) дополнительным данным пользователя (extra_data). Каждый атрибут будет представлять собой список, даже если в нем будет только одно значение.
- **Опции** (Options):
  - **Включенный** (Enabled) – если опция включена, метод будет активирован после его создания;
  - **Создавать объекты** (Create objects) – если опция включена, аутентификатору разрешено создавать объекты (пользователей, команды, организации);
  - **Удалить пользователей** (Remove users) – если опция включена, пользователь проходящий аутентификацию с помощью этого источника, будет удален из всех других групп, в которые он был добавлен ранее.

TACACS+

- **Название** (Name) – название метода аутентификации.
- **Автоматически мигрировать пользователей из** (Auto migrate users from) – позволяет выбрать существующий метод аутентификации, из которого следует автоматически перенести пользователей.
- **Hostname of TACACS+ Server** – адрес узла или IP-адрес сервера TACACS+, используемого для проверки учетных данных пользователей.
- **TACACS+ Authentication Protocol** – протокол аутентификации, применяемый сервером TACACS+.
- **Shared secret for authenticating to TACACS+ server** – общий секрет для аутентификации на сервере TACACS+.
- **TACACS+ client address sending enabled** – включение передачи IP-адреса клиента при выполнении аутентификации.
- **Additional Authenticator Fields** – используется для задания дополнительных параметров в формате YAML или JSON.
- **Опции** (Options):
  - **Включенный** (Enabled) – если опция включена, метод будет активирован после его создания;
  - **Создавать объекты** (Create objects) – если опция включена, аутентификатору разрешено создавать объекты (пользователей, команды, организации);
  - **Удалить пользователей** (Remove users) – если опция включена, пользователь проходящий аутентификацию с помощью этого источника, будет удален из всех других групп, в которые он был добавлен ранее.

4. Нажмите кнопку *Далее* (Next).
5. (Опционально) При создании метода аутентификации можно задать правила назначения ролей и прав доступа пользователям, прошедшим внешнюю аутентификацию. Каждое правило определяется типом сопоставления, условием его срабатывания (триггером) и действием при отзыве соответствия. Для этого выполните следующие действия:
   1. Нажмите кнопку *Добавить сопоставление аутентификации* (Add authentication mapping) и выберите тип сопоставления. Доступны следующие типы сопоставления:
      - **Разрешить** (Allow) – разрешает вход определенным пользователям или группам, прошедшим внешнюю аутентификацию.
      - **Организация** (Organization) – назначает пользователям роль в конкретной организации.
      - **Команда** (Team) – добавляет пользователей в определенную команду.
      - **Роли** (Role) – выдает системную роль.
      - **Суперпользователь** (Superuser) – назначает пользователю права суперпользователя (полный доступ ко всем объектам платформы).
   2. Заполните поля, соответствующие выбранному типу сопоставления.

      ##### Allow

      Тип сопоставления **Разрешить** (Allow) используется для предоставления пользователям доступа к платформе после успешной аутентификации.

      Для заполнения доступны следующие поля:

      - **Название** (Name) – уникальное название правила сопоставления.
      - **Триггер** (Trigger) – условие, при котором применяется правило:

  - **Всегда** (Always) – разрешение доступа применяется ко всем пользователям;
  - **Никогда** (Never) – правило не срабатывает (удобно для временного отключения);
  - **Группы** (Groups) – правило срабатывает, если пользователь входит в указанную группу, возвращаемую провайдером аутентификации;
  - **Атрибуты** (Attributes) – правило срабатывает при совпадении указанных атрибутов, полученных от провайдера аутентификации.
- **Отозвать** (Revoke) – если включено, доступ будет отозван при утрате соответствия условию.

Organization

Тип сопоставления **Организация** (Organization) автоматически добавляет пользователей во внутреннюю организацию платформы и назначает им роль после успешной аутентификации.

Для заполнения доступны следующие поля:

- **Название** (Name) – уникальное название правила сопоставления.
- **Триггер** (Trigger) – условие, при котором применяется правило:
  - **Всегда** (Always) – разрешение доступа применяется ко всем пользователям;
  - **Никогда** (Never) – правило не срабатывает (удобно для временного отключения);
  - **Группы** (Groups) – правило срабатывает, если пользователь входит в указанную группу, возвращаемую провайдером аутентификации;
  - **Атрибуты** (Attributes) – правило срабатывает при совпадении указанных атрибутов, полученных от провайдера аутентификации.
- **Отозвать** (Revoke) – если включено, при утрате соответствия условию пользователи будут исключены из организации.
- **Организация** (Organization) – организация, в которую будут добавлены пользователи.
- **Роль** (Role) – роль, назначаемая пользователям в рамках выбранной организации.

Team

Тип сопоставления **Team** (Команда) позволяет автоматически добавлять пользователей в определенную команду и назначать им роль внутри выбранной команды.

Для заполнения доступны следующие поля:

- **Название** (Name) – уникальное название правила сопоставления.
- **Триггер** (Trigger) – условие, при котором применяется правило:
  - **Всегда** (Always) – разрешение доступа применяется ко всем пользователям;
  - **Никогда** (Never) – правило не срабатывает (удобно для временного отключения);
  - **Группы** (Groups) – правило срабатывает, если пользователь входит в указанную группу, возвращаемую провайдером аутентификации;
  - **Атрибуты** (Attributes) – правило срабатывает при совпадении указанных атрибутов, полученных от провайдера аутентификации.

- **Отозвать** (Revoke) – если включено, пользователь будет исключен из команды при утрате соответствия условию.
- **Команда** (Team) – команда, в которую будут добавлены пользователи.
- **Организация** (Organization) – организация, в которую будут добавлены пользователи.
- **Роль** (Role) – роль, назначаемая пользователям в рамках выбранной команды.

Role

Тип сопоставления **Role** (Роль) используется для назначения пользователям определенной роли.

Для заполнения доступны следующие поля:

- **Название** (Name) – уникальное название правила сопоставления.
- **Триггер** (Trigger) – условие, при котором применяется правило:
  - **Всегда** (Always) – разрешение доступа применяется ко всем пользователям;
  - **Никогда** (Never) – правило не срабатывает (удобно для временного отключения);
  - **Группы** (Groups) – правило срабатывает, если пользователь входит в указанную группу, возвращаемую провайдером аутентификации;
  - **Атрибуты** (Attributes) – правило срабатывает при совпадении указанных атрибутов, полученных от провайдера аутентификации.
- **Отозвать** (Revoke) – если включено, роль будет отозвана при утрате соответствия условию.
- **Команда** (Team) – команда, в которую будут добавлены пользователи.
- **Организация** (Organization) – организация, в которую будут добавлены пользователи.
- **Роль** (Role) – роль, назначаемая пользователям в рамках выбранной команды.

Superuser

Тип сопоставления **Superuser** (Суперпользователь) назначает пользователю административные права на уровне всей платформы.

Для заполнения доступны следующие поля:

- **Название** (Name) – уникальное название правила сопоставления.
- **Триггер** (Trigger) – условие, при котором применяется правило:
  - **Всегда** (Always) – разрешение доступа применяется ко всем пользователям;
  - **Никогда** (Never) – правило не срабатывает (удобно для временного отключения);
  - **Группы** (Groups) – правило срабатывает, если пользователь входит в указанную группу, возвращаемую провайдером аутентификации;
  - **Атрибуты** (Attributes) – правило срабатывает при совпадении указанных атрибутов, полученных от провайдера аутентификации.
- **Отозвать** (Revoke) – если включено, роль суперпользователя будет отозвана при утрате соответствия условию.

6. Нажмите кнопку *Далее* (Next).
7. (Опционально) Ознакомьтесь с порядком сопоставлений аутентификации и скорректируйте его при необходимости.

8. Нажмите кнопку *Далее* (Next).
9. Проверьте корректность настроек и нажмите кнопку *Завершить* (Finish).

Редактирование

Чтобы изменить метод аутентификации, выполните следующие действия:

1. В таблице нажмите на название метода аутентификации, который необходимо изменить.
2. Нажмите кнопку *Изменить аутентификацию* (Edit authentication).
3. Внесите необходимые изменения.
4. Нажмите кнопку *Завершить* (Finish).

Удаление

Чтобы удалить методы аутентификации, выполните следующие действия:

1. В таблице установите флаги в строках с методами аутентификации, которые необходимо удалить.
2. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите пункт *Удалить аутентификации* (Delete authentications).
3. Подтвердите удаление.

Организации

Окно **Организации** (Organizations) используется для структурного разделения ресурсов и управления доступом с помощью следующих операций:

- создание и редактирование *организаций*;
- объединение пользователей и команд в рамках организационной модели;
- назначение организациям ресурсов, включая проекты, инвентари, своды правил и полномочия;
- контроль членства и ролей участников внутри организации.

Для перехода к окну **Организации** выберите на панели навигации *Управление доступом* ▸ *Организации* (*Access Management* ▸ *Organizations*).

Таблица организаций

Внешний вид окна **Организации** (Organizations) представлен на схеме:

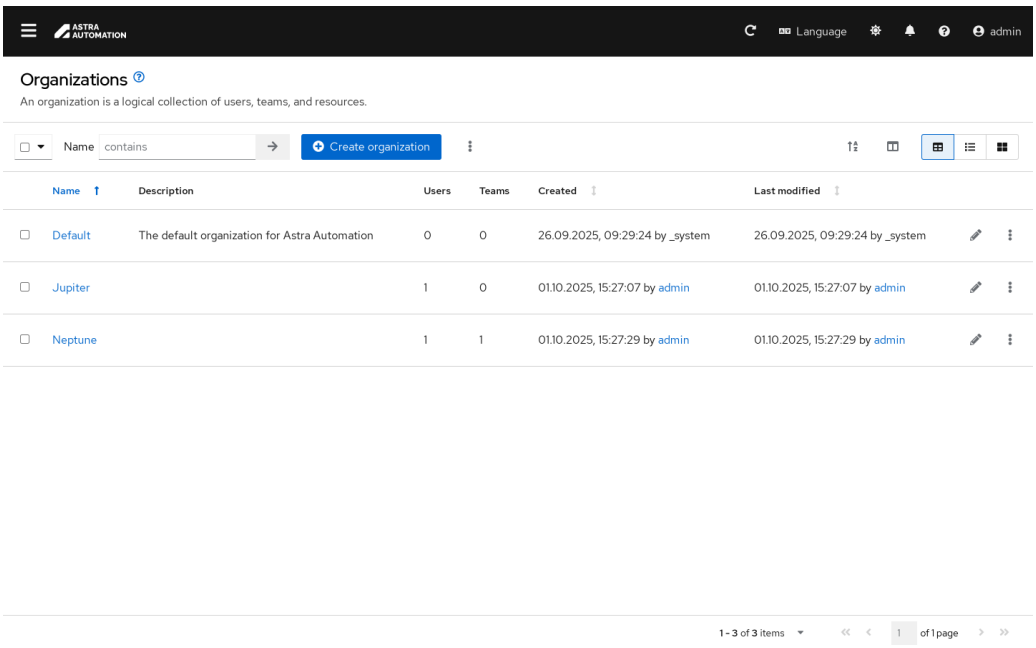


Таблица организаций состоит из столбцов согласно настройке с помощью кнопки *Настроить столбцы* (Manage columns). Полный список столбцов:

- флаги для выбора нескольких записей;
- **Название** (Name) – ссылка на окно просмотра подробных сведений об организации;
- **Описание** (Description) – дополнительная информация об организации;
- **Пользователи** (Users) – количество пользователей, связанных с организацией;
- **Команды** (Teams) – количество команд пользователей, связанных с организацией;
- **Дата создания** (Created) – дата и время создания организации;
- **Последнее изменение** (Last modified) – дата и время последнего изменения организации;
- кнопки вызова часто выполняемых действий:
  - редактирование свойств организации;
  - удаление организации.

Просмотр

Для получения подробных сведений об организации нажмите на ее название в таблице организаций.

Окно сведений об организации состоит из следующих вкладок:

- **Подробности** (Details) – общие сведения об организации;
- **Пользователи** (Users) – список пользователей, связанных с организацией;
- **Администраторы** (Administrator) – список администраторов организации;
- **Команды** (Teams) – список команд организации;
- **Среды исполнения** (Execution Environments) – список сред исполнения, принадлежащих организации;
- **Уведомления** (Notifications) – список уведомлений, связанных с организацией.

Создание

Для создания организации выполните следующие действия:

1. В окне **Организации** (Organizations) нажмите кнопку *Создать организацию* (Create organization).
2. Заполните форму **Сведения об организации** (Organization details):
   - **Название** (Name) – название организации.

     Особенности заполнения поля:

     - Не допускается создание двух организаций с одним и тем же названием.
     - Название организации не может состоять из одних пробельных символов (пробелы, табуляции и тому подобное).
     - Допускается использование символов кириллицы и специальных символов.
   - **Описание** (Description).

     В описании можно указать дополнительные данные об организации, используемые при администрировании контроллера. Например, контакты ответственных лиц или назначение организации.
   - **Среда исполнения** (Execution environment) – *среда исполнения*, которую по умолчанию следует использовать на уровне организации. Эта настройка имеет более высокий приоритет чем та, которая задана *на уровне контроллера*.
   - **Группы исполняющих узлов** (Instance Groups) – группы узлов контроллера, которые следует использовать для запуска заданий, созданных пользователями организации.

     Подробности о порядке выбора узлов для запуска заданий см. в секции *Выбор группы узлов для запуска заданий*.
   - **Полномочия Galaxy** (Galaxy Credentials) – список полномочий типа *API-токен Ansible Galaxy/Automation Hub*.

     По умолчанию в этом поле указано только полномочие для доступа к реестру коллекций Ansible Galaxy[210]. Для доступа к другим реестрам коллекций Ansible, например, Automation Hub и Private Automation Hub, создайте полномочия указанного типа и выберите их в этом поле.
   - **Максимум узлов** (Max Hosts)

     Ограничение на количество узлов, к которым пользователи организации могут применять сценарии автоматизации.

     Подробности см. в секции *Максимальное количество узлов*.
3. Нажмите кнопку *Далее* (Next).
4. Проверьте корректность введенных данных и нажмите кнопку *Завершить* (Finish).

Изменение

Для изменения свойств организации выполните следующие действия:

1. В окне **Организации** (Organizations) нажмите на название организации, свойства которой необходимо изменить.
2. Нажмите кнопку *Редактировать организацию* (Edit organization).
3. Внесите необходимые изменения в форме **Сведения об организации** (Organization details).
4. Нажмите кнопку *Далее* (Next).
5. Проверьте корректность введенных данных и нажмите кнопку *Завершить* (Finish).

[210] https://galaxy.ansible.com/

Удаление

Предупреждение

Вместе с организацией удаляются связанные с ней ресурсы. Принадлежащие организации пользователи и команды при этом не удаляются, а открепляются от удаляемой организации и могут быть в дальнейшем привязаны к другой.

Чтобы удалить организацию, выполните следующие действия:

1. В таблице установите флаги в строках с организациями, которые необходимо удалить.
2. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите пункт *Удалить организации* (Delete organizations).
3. Подтвердите удаление.

Настройка доступа

Во вкладках **Пользователи** (Users) и **Команды** (Teams) выводится список связанных с организацией пользователей и команд пользователей. Нажатие на имя пользователя или название команды приводит к переходу в окно просмотра сведений о пользователе или команде соответственно.

Назначение ролей пользователям

Чтобы назначить пользователям роли на уровне организации, выполните следующие действия:

1. Во вкладке **Пользователи** (Users) нажмите кнопку *Добавить пользователей* (Add users).
2. Выберите пользователей, которым хотите назначить роли, и нажмите кнопку *Далее* (Next).
3. Выберите роли, которые хотите назначить выбранным на предыдущем шаге пользователям.

   Примечание

   При *связывании пользователя и команды* выбранные командные роли будут назначены пользователю автоматически.

4. Проверьте корректность введенных данных и нажмите кнопку *Завершить* (Finish).

Назначение ролей командам пользователей

Чтобы назначить командам роли на уровне организации, выполните следующие действия:

1. Во вкладке **Команды** (Teams) нажмите кнопку *Добавить роль* (Add roles).
2. Выберите команды, которым хотите назначить роли, и нажмите кнопку *Далее* (Next).
3. Выберите роли, которые хотите назначить выбранным на предыдущем шаге командам.
4. Проверьте корректность введенных данных и нажмите кнопку *Завершить* (Finish).

Отзыв ролей у пользователей

Чтобы отозвать роль у пользователя, выполните следующие действия:

1. Во вкладке **Пользователи** (Users) нажмите на имя пользователя, у которого необходимо отозвать роль.
2. Перейдите во вкладку **Роли** (Roles).
3. Нажмите кнопку **-** рядом с названием отзываемой роли.
4. Подтвердите отзыв.

Примечание

Роли «Системный администратор» (System Administrator) и «Системный аудитор» (System Auditor) нельзя отозвать через настройку доступа на уровне организации. Используйте для этого изменение свойств пользователя.

Отзыв ролей у команд

Чтобы отозвать роль у команды, выполните следующие действия:

1. Во вкладке **Команды** (Teams) нажмите на название команды, у которой необходимо отозвать роль.
2. Перейдите во вкладку **Роли** (Roles).
3. Нажмите кнопку **-** рядом с названием отзываемой роли.
4. Подтвердите отзыв.

Уведомления

Во вкладке **Уведомления** (Notifications) выводится список уведомлений, связанных с организацией. Для каждого уведомления доступны переключатели, управляющие типом событий, при наступлении которых должно быть отправлено уведомление:

- **Согласование** (Approval) – согласован переход к очередному заданию в потоке заданий;
- **Пуск** (Start) – запущено задание или поток заданий;
- **Успех** (Success) – успешно выполнено задание или поток заданий;
- **Сбой** (Failure) – при выполнении задания произошла ошибка, для которой не предусмотрен обработчик.

Примечание

Если уведомления для организации не настроены, отображается пустой список. Для создания уведомлений и управления ими выберите на панели навигации *Автоматизация процессов* ▸ *Администрирование* ▸ *Источники уведомлений* (*Automation Execution* ▸ *Administration* ▸ *Notifiers*).

Примеры

Изучите управление организациями на примерах.

Назначение роли отдельному пользователю

Пусть пользователю `alex` необходимо назначить роль, позволяющую управлять всеми проектами организации. Для этого в окне просмотра сведений об организации выполните следующие действия:

1. Во вкладке **Пользователи** (Users) нажмите кнопку *Добавить пользователей* (Add users).
2. Из списка пользователей выберите `alex` и нажмите кнопку *Далее* (Next).
3. Во вкладке **Автоматизация процессов** (Automation Execution) из списка ролей выберите **Organization Project Admin** и нажмите кнопку *Далее* (Next).
4. Во вкладке **Обработка событий** (Automation Decisions) из списка ролей выберите **Organization Project Admin** и нажмите кнопку *Далее* (Next).
5. Проверьте корректность введенных данных и нажмите кнопку *Завершить* (Finish).

Назначение ролей команде

Пусть для управления инфраструктурой используется команда с названием `DevOps Team`. Всем ее участникам необходимо предоставить следующие привилегии:

- управление средами исполнения;
- управление средами принятия решений;
- управление проектами.

Для этого в окне организации выполните следующие действия:

1. Во вкладке **Команды** (Teams) нажмите кнопку *Добавить роль* (Add roles).
2. Из списка команд выберите `DevOps Team` и нажмите кнопку *Далее* (Next).
3. Из списка ролей выберите **Администратор проекта** (Project Admin) и **Администратор среды исполнения** (Execution Environment Admin).
4. Во вкладке **Автоматизация процессов** (Automation Execution) из списка ролей выберите **Organization Project Admin** и **Organization ExecutionEnvironment Admin** и нажмите кнопку *Далее* (Next).
5. Во вкладке **Обработка событий** (Automation Decisions) из списка ролей выберите **Organization Project Admin** и **Organization Decision Environment Admin** и нажмите кнопку *Далее* (Next).
6. Проверьте корректность введенных данных и нажмите кнопку *Завершить* (Finish).

Команды

Окно **Команды** (Teams) позволяет выполнять следующие действия с командами пользователей:

- *просматривать подробные сведения о команде*;
- *создавать новые команды*;
- *удалять команды*;
- *назначать и отзывать командные роли*.

Для перехода к окну **Команды** (Teams) выберите на панели навигации *Управление доступом* ▸ *Команды* (*Access Management* ▸ *Teams*).

Таблица команд

Внешний вид окна **Команды** (Teams) представлен на схеме:

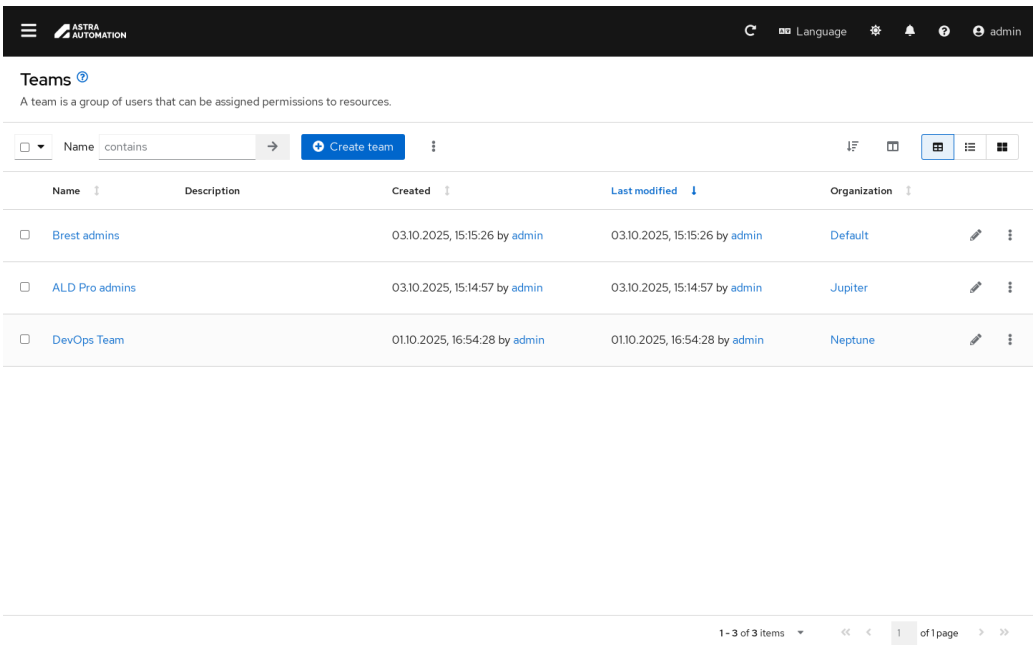


Таблица команд состоит из столбцов согласно настройке с помощью кнопки *Настроить столбцы* (Manage columns). Полный список столбцов:

- флаги для выбора нескольких записей;
- **Название** (Name) – ссылка для перехода в окно просмотра подробных сведений о команде;
- **Описание** (Description) – дополнительная информация о команде;
- **Дата создания** (Created) – дата и время создания команды, а также пользователь, который создал команду;
- **Последнее изменение** (Last modified) – дата и время последнего изменения команды, а также пользователь, который произвел изменения;
- **Организация** (Organization) – название организации, которой принадлежит команда;
- кнопки для вызова часто выполняемых действий:
  - редактирование команды;
  - удаление команды.

Просмотр

Для получения подробных сведений о команде нажмите на ее название в таблице.

Окно сведений о команде состоит из следующих вкладок:

- **Подробности** (Details) – общие сведения о команде;
- **Роли** (Roles) – список командных ролей. Указанные здесь роли автоматически назначаются всем участникам команды.
- **Пользователи** (Users) – список пользователей, добавленных в команду.
- **Администраторы** (Administrators) – список администраторов команды.

Создание

Для создания команды выполните следующие действия:

1. В окне **Команды** (Teams) нажмите кнопку *Создать команду* (Create team).
2. Заполните форму **Создать команду** (Create team):

- **Название** (Name) – укажите название команды.

  Особенности заполнения поля:

  - Название команды должно быть уникальным на уровне организации.
  - Название команды не может состоять из одних пробельных символов (пробелы, табуляции и так далее).
  - Допускается использование символов кириллицы и специальных символов.
- **Описание** (Description) – укажите дополнительные сведения о команде, например, контакты ответственных лиц или назначение команды.
- **Организация** (Organization) – выберите организацию, которой принадлежит команда.

3. Нажмите кнопку *Создать команду* (Create team).

Удаление

Предупреждение

При удалении команды ее участники теряют предоставленные командой привилегии.

Чтобы удалить команды, выполните следующие действия:

1. В таблице установите флаги в строках с командами, которые необходимо удалить.
2. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите пункт *Удалить команды* (Delete teams).
3. Подтвердите удаление.

Настройка доступа

Во вкладке **Роли** (Roles) выводится список назначенных команде *ролей*.

Назначение командных ролей

Чтобы предоставить участникам команды доступ к существующим ресурсам, выполните следующие действия:

1. Во вкладке **Роли** (Roles) нажмите кнопку *Добавить роль* (Roles).
2. Выберите тип ресурса и нажмите кнопку *Далее* (Next).
3. Выберите экземпляры ресурсов и нажмите кнопку *Далее* (Next).
4. Выберите роли и нажмите кнопку *Далее* (Next).
5. Проверьте корректность введенных данных и нажмите кнопку *Завершить* (Finish).

Отзыв ролей

Чтобы отозвать командую роль, выполните следующие действия:

1. Во вкладке **Роли** (Roles) нажмите кнопку **-** в строке отзываемой роли.
2. Подтвердите отзыв.

Пользователи

Окно **Пользователи** (Users) предназначено для полной административной работы с учетными записями с помощью следующих операций:

- создание, активация и блокировка *учетных записей пользователей* платформы;

- назначение ролей и полномочий с определением уровня доступа;
- отслеживание статуса учетных записей и управление их жизненным циклом.

Для перехода к окну **Пользователи** (Users) выберите на панели навигации *Управление доступом* ▸ *Пользователи* (*Access Management* ▸ *Users*).

Таблица пользователей

Внешний вид окна **Пользователи** (Users) представлен на схеме:

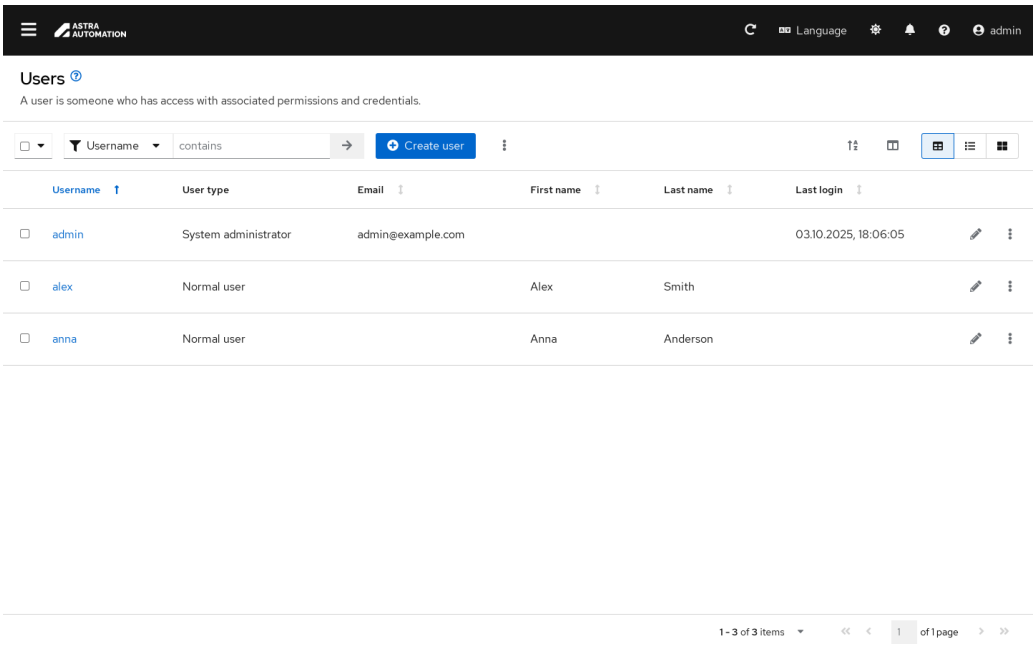


Таблица пользователей состоит из столбцов согласно настройке с помощью кнопки *Настроить столбцы* (Manage columns). Полный список столбцов:

- флаги для выбора нескольких записей;
- **Название учетной записи** (Username) – название учетной записи пользователя, используемой для работы с контроллером;
- **Тип пользователя** (User type) – тип пользователя;
- **Электронная почта** (Email) – электронная почта пользователя;
- **Имя** (First Name) – имя пользователя;
- **Фамилия** (Last Name) – фамилия пользователя;
- **Последний вход в систему** (Last login) – дата и время последнего входа пользователя в систему;
- **Дата создания** (Created) – дата и время создания пользователя;
- **Последнее изменение** (Last modified) – дата и время последнего изменения пользователя;
- кнопки вызова часто выполняемых действий:
  - редактирование пользователя;
  - удаление пользователя.

Просмотр

Для получения подробных сведений о пользователе нажмите на название его учетной записи в таблице пользователей.

Окно сведений о пользователе состоит из следующих вкладок:

- **Подробности** (Details) – общие сведения о пользователе.
- **Команды** (Teams) – список команд, участником которых является пользователь.

  Чтобы сделать пользователя участником команды, *свяжите его с необходимой командой*.
- **Роли** (Roles) – список ролей, назначенных пользователю.
- **Токены** (Tokens) – список токенов доступа, созданных пользователем.

Примечание

Эта вкладка доступна только при просмотре собственной учетной записи.

Создание

Для создания пользователя выполните следующие действия:

1. В окне **Пользователи** (Users) нажмите кнопку *Создать пользователя* (Create user).
2. Заполните форму **Создать пользователя** (Create user):
   - **Название учетной записи** (Username) – название учетной записи пользователя.

     Требования к названию учетной записи:
     - Максимально допустимая длина – 150 символов.
     - Разрешено использование букв, цифр и символов `_`, `@`, `+`, `.`, `-`.
     - Уникальность на уровне контроллера.
   - **Пароль** (Password) и **Подтвердить пароль** (Confirm Password) – пароль и его подтверждение.

     Длина пароля должна быть не менее 8 символов.
   - **Имя** (First Name).
   - **Фамилия** (Last Name).
   - **Электронная почта** (Email).
   - **Тип пользователя** (User Type).

     Подробности о типах пользователей см. в секции *Типы пользователей*.
   - **Организация** (Organization) – организация, которой принадлежит пользователь.
3. Нажмите кнопку *Создать пользователя* (Create user).

Удаление

Чтобы удалить пользователей, выполните следующие действия:

1. В таблице установите флаги в строках с пользователями, которые необходимо удалить.
2. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите пункт *Удалить пользователей* (Delete users).
3. Подтвердите удаление.

Настройка доступа

Окно **Пользователи** (Users) предоставляет функциональность, позволяющую выполнять с пользователями следующие действия:

- связывать с командами;
- назначение индивидуальные роли на доступ к экземплярам ресурсов.

Связывание с командой

Чтобы связать пользователя с командой, выполните следующие действия:

1. Во вкладке **Команды** (Teams) нажмите кнопку *Добавить команды* (Add teams).
2. Выберите команды, с которыми хотите связать пользователя, и нажмите кнопку *Добавить команды* (Add teams).

Назначение индивидуальных ролей

Чтобы назначить пользователю роль, выполните следующие действия:

1. Во вкладке **Роли** (Roles) нажмите кнопку *Добавить роль* (Add roles).
2. Выберите тип объектов.

   Поддерживаются следующие значения:

   - *шаблоны заданий (job templates)*;
   - *шаблоны потока заданий (workflow job templates)*;
   - полномочия (credentials);
   - *инвентарь (inventories)*;
   - *проекты (projects)*;
   - группы узлов контроллера (instance groups);
   - среды исполнения (execution environment);
   - шаблоны уведомлений (notification template).
3. Нажмите кнопку *Далее* (Next).
4. Выберите экземпляры объектов и нажмите кнопку *Далее* (Next).
5. Выберите роли. Список доступных ролей зависит от выбранного ранее типа объектов.
6. Нажмите кнопку *Далее* (Next).
7. Проверьте корректность введенных данных и нажмите кнопку *Завершить* (Finish).

Отзыв ролей

Чтобы отозвать у пользователя роль, выполните следующие действия:

1. Во вкладке **Роли** (Roles) нажмите кнопку **-** в строке отзываемой роли.
2. Подтвердите отзыв.

Смена пароля

Чтобы изменить пароль пользователя, выполните следующие действия:

1. В таблице пользователей нажмите на название учетной записи пользователя, пароль которого необходимо изменить.
2. Во вкладке **Подробности** (Details) нажмите кнопку *Редактировать пользователя* (Edit user).
3. Укажите новый пароль в полях **Пароль** (Password) и **Подтвердить пароль** (Confirm Password).
4. Нажмите кнопку *Сохранить пользователя* (Save user).

Смена типа пользователя

Чтобы изменить *тип пользователя*, выполните следующие действия:

1. В таблице пользователей нажмите на название учетной записи пользователя, тип которого необходимо изменить.
2. Во вкладке **Подробности** (Details) нажмите кнопку *Редактировать пользователя* (Edit user).
3. В поле **Тип пользователя** (User Type) выберите необходимое значение.
4. Нажмите кнопку *Сохранить пользователя* (Save user).

Управление токенами

Токены позволяют сторонним приложениям получить доступ к контроллеру от имени выбранного пользователя. Каждый пользователь создает свои токены самостоятельно.

Для управления токенами выполните следующие действия:

1. Войдите в веб-интерфейс от имени необходимого пользователя.
2. На панели навигации выберите *Управление доступом* ▸ *Пользователи* (*Access Management* ▸ *Users*).
3. Нажмите на название учетной записи активного пользователя.

> **Совет**
>
> Для быстрого перехода в необходимое окно также можно выбрать в пользовательском меню пункт **Сведения о пользователе** (User details).

4. Выберите вкладку **Токены** (Tokens).

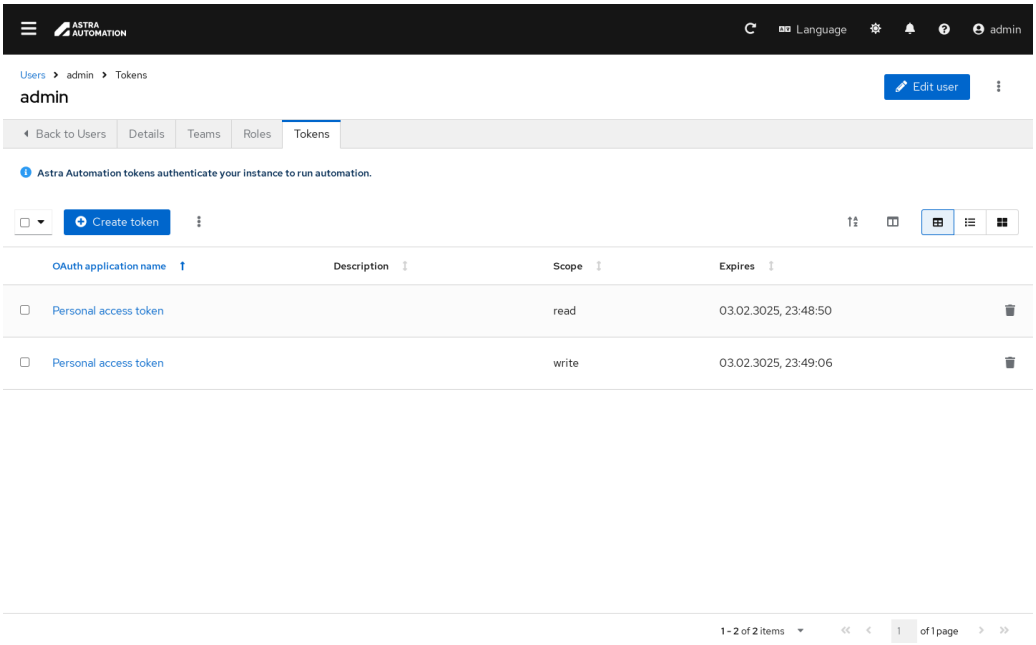


Таблица токенов состоит из столбцов согласно настройке с помощью кнопки *Настроить столбцы* (Manage columns). Полный список столбцов:

- Флаги для выбора нескольких записей.
- **Название OAuth-приложения** (OAuth application name) – название приложения, связанного с токеном. Если с токеном не связано ни одно приложение, выводится название **Личный токен доступа** (Personal access token).
- **Описание** (Description) – описание токена.

- **Область** (Scope) – режим доступа к данным контроллера, предоставляемый токеном.
- **Истекает** (Expires) – дата и время истечения срока действия токена.
- **Дата создания** (Created) – дата и время создания токена.
- **Последнее изменение** (Last modified) – дата и время последнего изменения токена.
- Кнопка удаления токена.

Создание токена

Для создания токена выполните следующие действия:

1. Во вкладке **Токены** (Tokens) нажмите кнопку *Создать токен* (Create token).
2. Заполните форму **Создать токен** (Create token):
   - **OAuth-приложение** (OAuth application) – выберите приложение, для которого создается токен.

     Примечание

     Приложение должно быть предварительно добавлено в список приложений.

   - **Описание** (Description) – добавьте краткое описание токена, например, название приложения, для которого он создается.
   - **Область** (Scope) – выберите режим доступа к данным контроллера, предоставляемый токеном:
     - **Чтение** (Read);
     - **Запись** (Write).
3. Нажмите кнопку **Создать токен** (Create token).
4. В открывшемся диалоговом окне скопируйте значение из поля **Токен** (Token) в буфер обмена и сохраните в надежное место

   Важно

   После закрытия диалогового окна получить доступ к значению токена невозможно.

5. Закройте окно **Информация о токене** (Token information).

Удаление токена

Предупреждение

При удалении токена все приложения, которые его используют, теряют доступ к контроллеру.

Для удаления пользовательского токена выполните следующие действия:

1. Во вкладке **Токены** (Tokens) установите флаги в строках с токенами, которые необходимо удалить.
2. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите пункт *Удалить токены* (Delete tokens).
3. Подтвердите удаление.

Роли

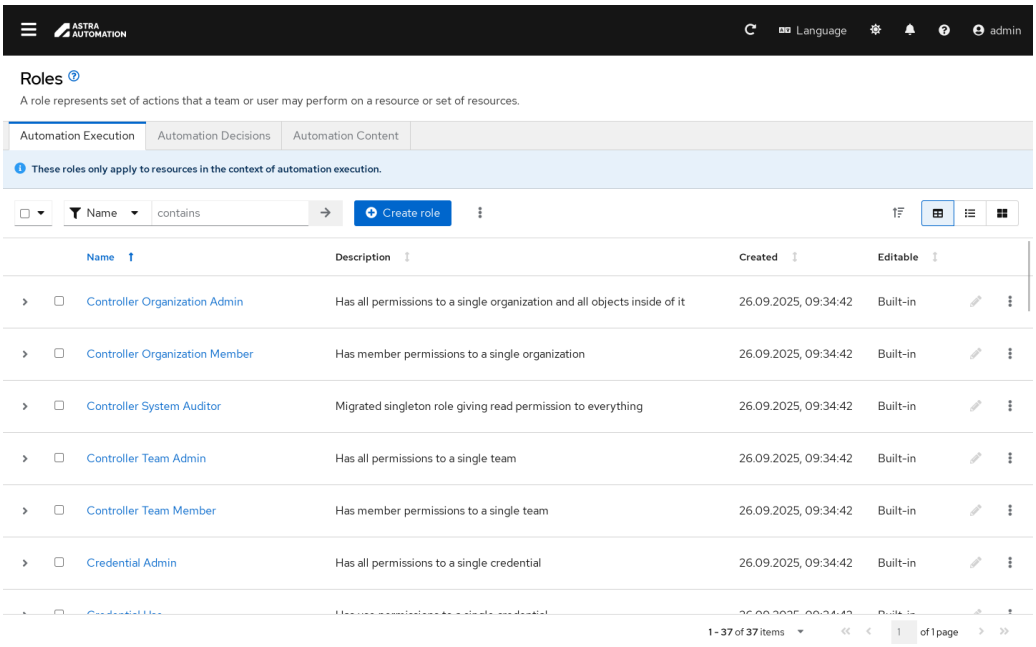
Окно **Роли** (Roles) обеспечивает управление ролевой моделью доступа с помощью следующих операций:

- просмотр встроенных системных ролей и их разрешений;
- создание и настройка пользовательских ролей;
- назначение ролей пользователям и командам для контроля привилегий;
- управление делегированием доступа к ресурсам платформы.

Для перехода к окну **Роли** (Roles) выберите на панели навигации *Управление доступом* ▸ *Роли* (*Access Management* ▸ *Roles*).

Таблица ролей

Внешний вид окна **Роли** (Roles) представлен на схеме:



Окно **Роли** (Roles) состоит из трех вкладок:

- **Автоматизация процессов** (Automation Execution) – роли применяющиеся только к ресурсам в контексте автоматизации процессов;
- **Обработка событий** (Automation Decisions) – роли применяющиеся только к ресурсам в контексте обработки событий;
- **Контент автоматизации** (Automation Content) – роли применяющиеся только к ресурсам в контексте контента автоматизации.

Во всех вкладках выводится таблица, содержащая следующие столбцы:

- флаги для выбора нескольких записей;
- **Название** (Name) – название роли;
- **Описание** (Description) – краткое описание роли;
- **Дата создания** (Created) – дата и время создания роли;
- **Редактируемый** (Editable) – возможность изменения свойств роли;
- кнопки для быстрого вызова часто выполняемых действий:
  - редактирование роли;
  - удаление роли.

Примечание

Встроенные (built-in) роли нельзя изменить или удалить.

Создание пользовательской роли

Для создания роли выполните следующие действия:

1. В окне **Роли** (Roles) нажмите кнопку *Создать роль* (Create role).
2. Заполните форму **Создать роль** (Create role):
    1. **Название** (Name) – название роли. Не допускается создание двух ролей с одним и тем же названием.

       Внимание

       К названию создаваемой роли для компонента Private Automation Hub предъявляются следующие дополнительные требования:

       - начинается с префикса `galaxy.`;
       - может содержать только символы латинского алфавита, цифры, точки и символы подчеркивания.

    2. **Описание** (Description) – краткое описание роли.
    3. **Тип контента** (Content Type) – тип контента, которым сможет управлять пользователь с создаваемой ролью.

       Важно

       Будьте внимательны при выборе типа контента – его нельзя изменить.

    4. **Привилегии** (Permissions) – действия, которые позволяет выполнять роль.
3. Нажмите кнопку *Создать роль* (Create role).

Редактирование пользовательской роли

Для редактирования роли выполните следующие действия:

1. В окне **Роли** (Roles) нажмите кнопку *Редактировать роль* (Edit role) в строке роли, которую необходимо редактировать.
2. Измените необходимые данные в форме **Изменить роль** (Edit role).
3. Нажмите кнопку *Сохранить роль* (Save role).

Удаление пользовательской роли

Предупреждение

При удалении роли у связанных с ней команд и пользователей отзываются предоставляемые ролью разрешения.

Чтобы удалить пользовательские роли, выполните следующие действия:

1. В таблице установите флаги в строках с ролями, которые необходимо удалить.
2. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите пункт *Удалить роли* (Delete roles).

3. Подтвердите удаление.

OAuth-приложения

Окно **OAuth-приложения** (OAuth Applications) предназначено для управления интеграциями внешних сервисов с платформой по протоколу OAuth 2.0 с помощью следующих операций:

- регистрация новых OAuth-клиентов и назначение им параметров доступа;
- генерация и отзыв токенов авторизации для API;
- контроль активных интеграций и применение политик безопасности для внешних систем.

Для перехода к окну **OAuth-приложения** (OAuth Applications) выберите на панели навигации *Управление доступом* ▸ *OAuth-приложения* (*Access Management* ▸ *OAuth Applications*).

Таблица приложений

Внешний вид окна **OAuth-приложения** (OAuth Applications) представлен на схеме:

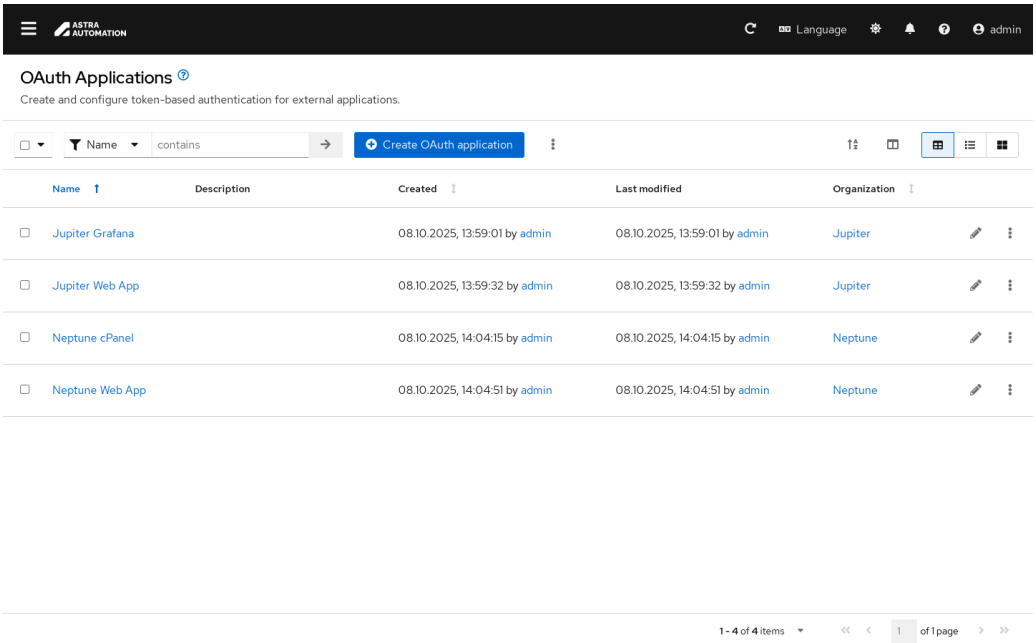


Таблица приложений состоит из столбцов согласно настройке с помощью кнопки *Настроить столбцы* (Manage columns). Полный список столбцов:

- флаги для выбора нескольких записей;
- **Название** (Name) – ссылка для перехода в окно просмотра сведений о приложении;
- **Описание** (Description) – краткое описание приложения;
- **Дата создания** (Created) – дата и время создания приложения;
- **Последнее изменение** (Last Modified) – дата и время последнего изменения приложения;
- **Организация** (Organization) – ссылка для перехода в окно просмотра сведений об организации-владельце;
- кнопки для быстрого вызова часто выполняемых действий:
  - редактирование приложения;
  - удаление приложения.

Просмотр сведений о приложении

Для получения подробных сведений о приложении нажмите на ссылку с его названием в таблице приложений.

Окно сведений о приложении состоит из следующих вкладок:

- **Подробности** (Details) – общие сведения о приложении.
- **Токены** (Tokens) – таблица связанных с приложением токенов.

  Таблица токенов состоит из следующих столбцов:

  - флаги для выбора нескольких записей;
  - **Название** (Name) – ссылка для перехода в окно просмотра сведений о пользователе-владельце токена;
  - **Область** (Scope) – режим доступа к данным платформы, предоставляемый токеном;
  - **Истекает** (Expires) – дата и время истечения срока действия токена.

Создание приложения

Для создания приложения выполните следующие действия:

1. В окне **OAuth-приложения** (OAuth Applications) нажмите кнопку *Создать OAuth-приложение* (Create OAuth application).
2. Заполните форму **Создать OAuth-приложение** (Create OAuth application):
   - **Название** (Name) – укажите название приложения. Название должно быть уникальным в рамках организации.
   - **Описание** (Description) – укажите описание приложения.
   - **Организация** (Organization) – выберите организацию-владельца.
   - **Тип разрешения** (Authorization grant type) – выберите тип доступа к контроллеру.

     Поддерживаются следующие значения:

     - **Код авторизации** (Authorization code);
     - **Пароль** (Password).
   - **Тип клиента** (Client type) – укажите уровень защищенности приложения, которому предоставляете доступ.

     Поддерживаются следующие значения:

     - **Конфиденциальный** (Confidential) – приложение имеет механизмы защиты доступа к своим данным;
     - **Публичный** (Public) – данные приложения публично доступны.
   - **URI перенаправления** (Redirect URIs) – укажите список разрешенных URI. В качестве разделителя используйте символ пробела.

     Примечание

     Это поле обязательно для заполнения, если в поле **Тип разрешения** (Authorization grant type) выбрано значение **Код авторизации** (Authorization code).
3. Нажмите кнопку *Создать OAuth-приложение* (Create OAuth application).
4. В открывшемся окне **Application information** выводятся следующие данные, необходимые для использования приложения:
   - **ID клиента** (Client ID) – идентификатор клиента.

- **Client secret** – секретная фраза клиента. Выводится только если при создании приложения выбран тип клиента «Конфиденциальный».

> **Важно**
>
> Сохраните показанные значения в надежном месте – после закрытия диалогового окна получить к ним доступ повторно невозможно.

5. Создайте токены согласно инструкций, приведенных в секции *Создание токена*.

Удаление приложения

> **Опасно**
>
> Вместе с приложением удаляются все связанные с ним токены. Для восстановления доступа приложений к контроллеру все токены нужно будет создать заново.

Чтобы удалить приложения, выполните следующие действия:

1. В таблице установите флаги в строках с приложениями, которые необходимо удалить.
2. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите пункт *Удалить OAuth-приложения* (Delete OAuth Applications).
3. Подтвердите удаление.

Удаление токенов приложения

> **Предупреждение**
>
> При удалении токена использующие его приложения теряют доступ к контроллеру.

Чтобы удалить токен приложения, выполните следующие действия:

1. В таблице нажмите на название приложения, токен которого необходимо удалить.
2. Во вкладке **Токены** (Tokens) включите флаги в строках с удаляемыми токенами.
3. На панели инструментов нажмите кнопку ⋮ и в открывшемся меню выберите пункт *Удалить токены* (Delete tokens).
4. Подтвердите удаление.

ГЛАВА 14

Настройка платформы

В API и графическом интерфейсе платформы предусмотрена настройка различных компонентов Astra Automation, включая параметры подписки, системы безопасности и системы управления журналами. Можно настроить формат визуального представления данных и включить систему помощи поиска неисправностей.

14.1 Категории системных параметров

Системные параметры организованы в несколько основных категорий. Каждая категория содержит взаимосвязанные параметры, которые можно редактировать одновременно.

Подробная информация о каждой категории доступна в *описании графического интерфейса*.

14.1.1 Подписка и лицензия

Настройка подписки обеспечивает доступ к интерфейсу платформы. Подписка, в соответствии с приобретенной лицензией, определяет предельное количество управляемых узлов (автоматизируемых узлов). Для эффективного использования этого лимита предусмотрено управление лицензией:

- оценка количество автоматизируемых узлов и неиспользуемых возможностей лицензии;
- своевременное удаление неиспользуемых узлов из списка автоматизируемых узлов для подключения других узлов.

Порядок расчета количества узлов

Подписка предоставляет право на использование Automation Controller для автоматизации определенного количества управляемых узлов.

Узел называется *автоматизируемым*, если он связан хотя бы с одним заданием.

Подсчет количества автоматизируемых узлов выполняется по их уникальным идентификаторам. Использование доменных имен или названий вместо IP-адреса в качестве идентификаторов имеет следующие особенности:

- Если IP-адрес узла изменился, но доменное имя или название осталось прежним и корректно конвертируется в измененный IP-адрес, счетчик автоматизируемых узлов не увеличивается.

- Несколько узлов с разными доменными именами или названиями, указывающими на один и тот же IP-адрес, считаются одним узлом.

Удаление неиспользуемых узлов

Функция мягкого удаления исключает неиспользуемые узлы из подсчета количества автоматизируемых узлов, но оставляет запись о них в системе.

Функцию мягкого удаления следует использовать в следующих случаях:

- для временных узлов;
- для узлов, которые больше не используются и никогда не будут автоматизироваться;
- для одноразовых или тестовых узлов, которые не являются постоянными рабочими серверами.

Функция не позволяет обойти ограничение на количество автоматизируемых узлов. Если повторно использовать удаленный таким способом узел, то он будет считаться автоматизируемым.

Чтобы произвести мягкое удаление выполните следующие шаги в графической панели платформы:

1. В панели навигации выбурите **Автоматизация процессов > Администрирование > Метрики узлов** (Automation Execution > Administration > Host Metrics).
2. В окне **Метрики узлов** (Host Metrics) отметьте узлы, подлежащие мягкому удалению.
3. В выпадающем списке действий выберите **Удалить названия узлов** (Delete hostnames).

Ограничение количества управляемых узлов на организацию

Чтобы использовать подписку наиболее эффективно, создайте организации и настройте ограничения на количество узлов, которые может автоматизировать каждая из них. Это позволит эффективно распределить автоматизируемые узлы между пользователями исходя из их потребностей.

14.1.2 Сетевые параметры

Сетевые взаимодействие платформы настраивается с помощью параметров шлюза, которые в основном касаются безопасного подключения к платформе:

- базовый URL платформы, используемый для формирования ссылок в уведомлениях;
- применение сертификатов TLS (да или нет);
- ограничения на пароли пользователей;
- разрешение использования базового варианта авторизации через API (basic auth);
- ограничения на анонимный доступ по API;
- ограничения на токены доступа по API;
- временные ограничения на доступ к платформе;
- использование определенного имени пользователя при взаимодействии с социальными сетями.

14.1.3 Настройки визуального представления данных

Настройка визуального представления позволяет организовать удобную раскладку окон, формат представления и интервал обновления данных. Можно выбрать удобную цветовую схему.

14.1.4 Параметры безопасности

Среди системных параметров платформы есть параметры безопасности.

Список доверенных прокси-серверов (Proxy IP Allowed List)

Если платформа расположена за reverse proxy или load balancer, добавьте IP адреса этих сервисов. Это необходимо для учета специфики подключения через proxy:

- правильной идентификация IP-адреса клиента;
- корректное ограничение по каналу связи – rate limiting;
- регистрация в журналах настоящего IP-адреса пользователя (не proxy).

Список доверенных CORS источников (CSRF Trusted Origins List)

Для защиты от CSRF (Cross-Site Request Forgery) атак добавьте все домены, с которых могут быть отправлены запросы к API платформы.

Ограничение администраторов организаций

Если всеми пользователями должны управлять только системные администраторы, то такую возможность можно снять с администраторов организаций.

14.1.5 Параметры выполнения заданий

Задания автоматизации содержат множество параметров, некоторые из которых можно настроить в рамках всей системы:

- разрешенные модули Ansible;
- временный каталог для выполнения заданий;
- предельно допустимое количество одновременных заданий;
- время ожидания выполнения задания;
- предельно допустимое количество ответвленных процессов (forks) для одного задания;
- разрешение загрузки коллекций и ролей Ansible;
- локальные настройки проектов для заданий (каталог, интервал обновления);
- и другие.

14.1.6 Управление журналами

Настройку ведения журналов обеспечивают несколько параметров:

- уровень критичности регистрируемых данных (Log Level);
- настройка локального хранилища журналов;
- настройка на внешние агрегаторы журналов.

14.1.7 Поиск неисправностей

Несколько параметров могут быть полезными для поиска неисправностей:

- запрет очистки временного каталога;
- настройка запросов к веб-сервису;
- специальная настройка рецепторов.

14.2 Графический интерфейс

Раздел **Настройки** (Settings) – это центр управления параметрами платформы Astra Automation. Здесь можно следить за состоянием подписки, менять личные предпочтения, управлять безопасностью и интеграциями, а также решать задачи диагностики.

Раздел позволяет выполнять следующие действия:

- проверять статус лицензии и использование ресурсов;
- настраивать доступ к платформе и подключение внешних сервисов;
- менять тему интерфейса и формат отображения данных;
- определять правила запуска заданий;
- подключать журналы и системы мониторинга;
- собирать информацию для поиска неисправностей.

Ниже описаны основные разделы меню **Настройки**.

14.2.1 Подписка

В разделе **Подписка** (Subscription) проверяется текущий статус лицензии: ее тип, количество доступных узлов и срок действия. Эти данные помогают контролировать использование ресурсов и вовремя продлевать подписку.

14.2.2 Шлюз платформы

Раздел **Шлюз платформы** (Platform gateway) отвечает за подключение к инфраструктуре и внешним сервисам. Здесь настраиваются параметры безопасности, правила входа и доступ к системе. Этот раздел позволяет настроить защищенную работу платформы и удобную интеграцию с другими инструментами.

14.2.3 Пользовательские настройки

В разделе **Пользовательские настройки** (User Preferences) настраивается графический интерфейс всей платформы. Доступен выбор темы оформления, формата отображения данных и настройки обновления страниц. Эти параметры помогают сделать работу комфортнее.

14.2.4 Система

Раздел **Система** (System) задает общие правила работы платформы: как обрабатываются данные, кто имеет доступ и каким образом собирается аналитика. Обычно эти параметры изменяют администраторы при интеграции с другими системами.

14.2.5 Задания

Раздел **Задания** (Job) позволяет управлять выполнением автоматизации. Здесь определяются правила запуска, ограничения и дополнительные настройки для задач. Они помогают находить компромисс между безопасностью и производительностью.

14.2.6 Управление журналами

В разделе **Управление журналами** (Logging) платформа подключается к внешним системам ведения журналов. Это позволяет централизованно хранить сообщения, контролировать ошибки и упрощает диагностику.

14.2.7 Поиск неисправностей

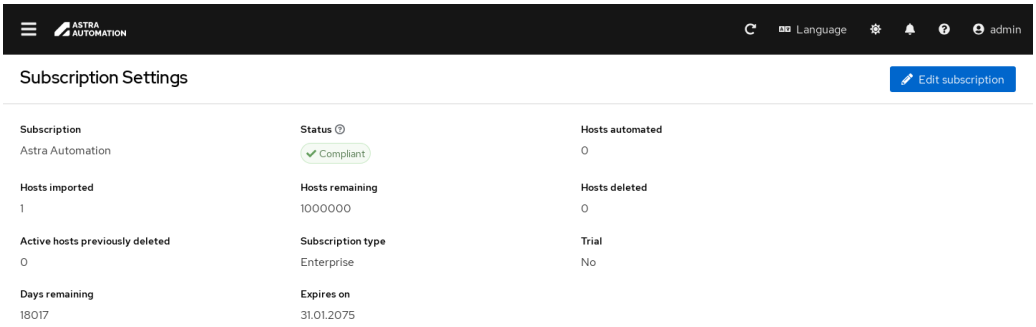
Раздел **Поиск неисправностей** (Troubleshooting) помогает находить и устранять проблемы. Здесь доступны инструменты очистки временных файлов, отладки запросов и контроля служебных процессов.

Подписка

Окно **Подписка** (Subscription) предоставляет сведения о лицензировании платформы и ее использовании:

- просмотр статуса и параметров текущей подписки;
- отслеживание количества потребляемых ресурсов согласно модели подписки.

Для перехода к окну **Подписка** (Subscription) выберите на панели навигации *Настройки* ▸ *Подписка* (*Settings* ▸ *Subscription*).



Получение информации о подписке

В окне просмотра информации о подписке выводится следующая информация:

- **Подписка** (Subscription) – название подписки.
- **Статус** (Status) – действующий статус подписки.

  После окончания срока действия подписки или при превышении количества автоматизированных узлов в этом поле будет сообщение «Не соответствует требованиям».
- **Узлы, к которым применялась автоматизация** (Hosts automated) – количество *автоматизированных* узлов, связанных хотя бы с одним заданием.

  Подробности о работе механизма подсчета значения в этом поле см. в секции *Порядок расчета количества узлов*.
- **Импортированные узлы** (Hosts imported) – количество узлов, импортированных из внешних источников.

  До момента связывания этих узлов с заданиями автоматизации они не учитываются при подсчете количества использованных узлов.
- **Осталось узлов** (Hosts remaining) – количество узлов, которые могут быть дополнительно автоматизированы в рамках действующей подписки.

  Совет

  Для исключения более неиспользуемых узлов из подсчета воспользуйтесь *функцией мягкого удаления*.
- **Удаленные узлы** (Hosts deleted) – количество узлов, которые были автоматизированы, но затем удалены.
- **Активные узлы, бывшие ранее удаленными** (Active hosts previously deleted) – количество узлов, которые были удалены, но теперь снова активны (связаны с заданиями).

- **Тип подписки** (Subscription type) – уровень технической поддержки действующей подписки:
  - `Standard` – «Стандартный»;
  - `Enterprise` – «Привилегированный».

  Подробности об уровнях технической поддержки см. в разделе Положение о технической поддержке Astra Automation[211] на сайте ПАО Группа Астра.
- **Пробная** (Trial) – используется пробный период подписки:
  - **Да** (Yes);
  - **Нет** (No).
- **Осталось дней** (Days remaining) – оставшийся срок действия подписки в днях.
- **Истекает** (Expires on) – локальные дата и время истечения срока действия подписки.

Редактирование

Для изменения действующей подписки в окне **Подписка** (Subscription) выполните следующие действия:

1. Нажмите кнопку *Изменить подписку* (Edit subscription).
2. Активируйте подписку, следуя инструкциям для используемого окружения:
   - *Активация в окружении с доступом к интернету*.
   - *Активация в окружении без доступа к интернету*.

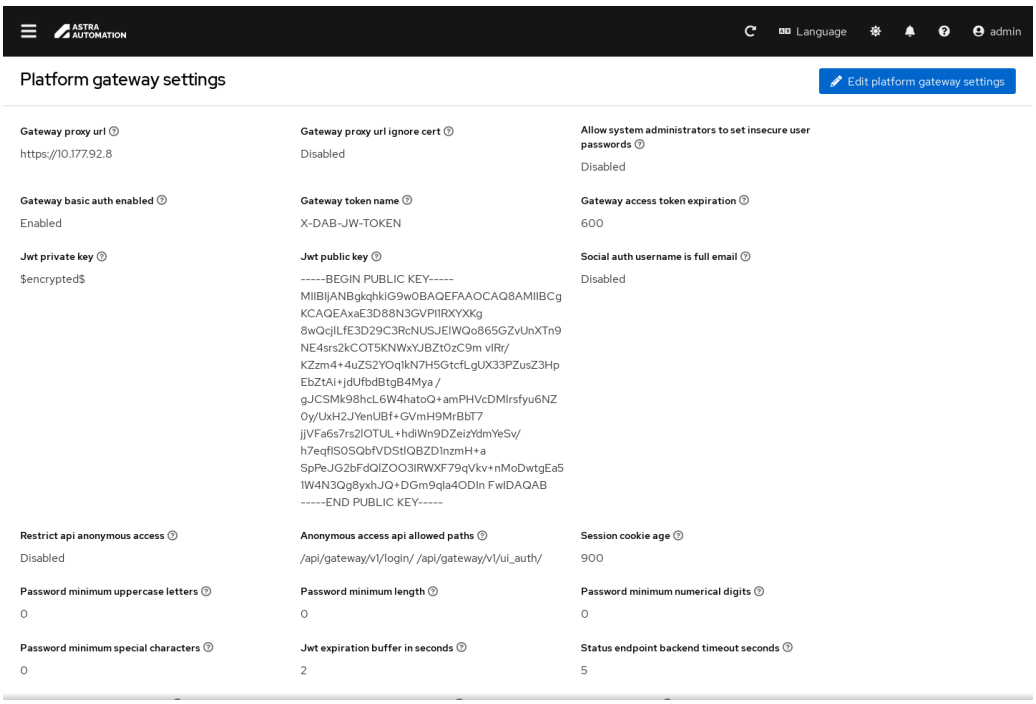
Шлюз платформы

Окно **Шлюз платформы** (Platform Gateway) предназначено для централизованного управления параметрами сетевого взаимодействия и аутентификации, используемыми платформой автоматизации при интеграции внутренних и внешних сервисов:

- управление безопасностью и политиками доступа;
- конфигурация параметров аутентификации;
- контроль параметров SSO и перенаправления входа;
- управление политиками сессий и паролей.

Для перехода к окну **Шлюз платформы** (Platform gateway) выберите на панели навигации *Настройки* ▸ *Шлюз платформы* (*Settings* ▸ *Platform gateway*).

[211] https://astra.ru/support/support-docs/polozhenie-o-tekhnicheskoy-podderzhke-astra-automation/



Получение информации о шлюзе

В окне просмотра информации отображается информация о настройках шлюза. Отображаются только заданные настройки. Для их редактирования воспользуйтесь *инструкцией*.

Полный список параметров:

- **Шлюз платформы** (Platform gateway):
  - **URL прокси шлюза** (Gateway proxy url) – URL шлюза, через который осуществляется взаимодействие с платформой.
  - **Игнорировать сертификат URL прокси шлюза** (Gateway proxy url ignore cert) – проверка сертификата HTTPS для шлюза.

    Возможные значения:

    * Включенный (Enable) – не проверять сертификат;
    * Неактивный (Disable) – проверять сертификат.
- **Безопасность** (Security):
  - **Разрешить системным администраторам устанавливать ненадежные пароли пользователей** (Allow system administrators to set insecure user passwords) – игнорировать политики сложности паролей для администраторов.

    Возможные значения:

    * Включенный (Enable) – администраторы могут сохранять ненадежные пароли;
    * Неактивный (Disable) – ненадежные пароли запрещены.
  - **Базовая аутентификация шлюза включена** (Gateway basic auth enabled) – базовая аутентификация для API шлюза.

    Возможные значения:

    * Включенный (Enable) – разрешить простую небезопасную базовую аутентификацию для HTTP;
    * Неактивный (Disable) – запретить базовую аутентификацию.
  - **Имя токена шлюза** (Gateway token name) – название HTTP-заголовка, который прокси передает во внутренний сервис (backend).

Предупреждение

При изменении значения этого поля необходимо обновить все внутренние сервисы (backends) для корректной работы.

- **Срок действия токена доступа шлюза** (Gateway access token expiration) – время жизни токена доступа (в секундах).
- **Приватный ключ JWT (Jwt private key) – закрытый ключ, используемый для подписи токенов JWT.**
  Отображается в зашифрованном виде.
- **Публичный ключ JWT** (Jwt public key) – открытый ключ, используемый для проверки подписи токенов JWT.

Примечание

Недоступно для редактирования.

- **Login redirect override** – URL, на который будут перенаправляться неавторизованные пользователи для входа. Если оставить поле пустым, пользователи будут направлены на стандартную страницу входа.
- **Подтвердить переопределение перенаправления входа** (Confirm login redirect override) – значение в этом поле должно совпадать со значением параметра **Login redirect override**. Используется для защиты от ошибок при вводе URL.
- **Имя пользователя социальной аутентификации – это полный email** (Social auth username is full email) – использование адреса электронной почты вместо названия учетной записи пользователя при аутентификации через внешние сервисы.

  Возможные значения:

  * Включенный (Enable) – использование адреса электронной почты;
  * Неактивный (Disable) – использование названия учетной записи.
- **Restrict api anonymous access** (Restrict api anonymous access) – все конечные точки API, кроме указанных в списке **Anonymous access api allowed paths**, будут требовать аутентификации.

  Возможные значения:

  * Включенный (Enable) – требовать аутентификацию;
  * Неактивный (Disable) – не требовать аутентификацию.
- **Anonymous access api allowed paths** (Anonymous access api allowed paths) – список конечных точек доступа API, к которым разрешен анонимный доступ.

- **Сеанс** (Session):
  - **Время жизни cookie сессии** (Session cookie age) – срок действия cookie сеанса связи в секундах.
- **Безопасность паролей** (Password Security):
  - **Минимальное количество заглавных букв в пароле** (Password minimum uppercase letters) – минимальное количество заглавных букв, которое должно присутствовать в пароле учетной записи;
  - **Минимальная длина пароля** (Password minimum length) – минимальное количество символов, которое должно присутствовать в пароле учетной записи;

  - **Минимальное количество цифр в пароле** (Password minimum numerical digits) – минимальное количество цифр, которое должно присутствовать в пароле учетной записи;
  - **Минимальное количество специальных символов в пароле** (Password minimum special characters) – минимальное количество специальных символов, которое должно присутствовать в пароле учетной записи.

- **Пользовательский вход** (Custom Login):
  - **Custom login info** – произвольный текст (например, юридическое уведомление или дисклеймер), отображаемый в модальном окне входа.
  - **Custom logo** – изображение, используемое в качестве логотипа. Допустимые форматы: GIF, PNG, JPEG.
- **Другие настройки** (Other settings):
  - **Jwt expiration buffer in seconds** – время до истечения срока действия токена JWT, после которого он удаляется из кеша (в секундах).
  - **Таймаут бэкенда конечной точки статуса в секундах** (Status endpoint backend timeout seconds) – время ожидания (в секундах) при попытке соединения с внутренним сервисом (backend) через конечную точку статуса.
  - **Проверка бэкенда конечной точки статуса** (Status endpoint backend verify) – проверка SSL-сертификатов сервисов при вызове отдельных узлов для получения статусов.

    Возможные значения:

    * Включенный (Enable) – SSL-сертификаты будут проверяться;
    * Неактивный (Disable) – SSL-сертификаты проверяться не будут.
  - **Таймаут запроса** (Request timeout) – время ожидания (в секундах), по истечении которого прокси возвращает ошибку HTTP 504 (Gateway Timeout).
  - **Разрешить внешним пользователям создавать OAuth2-токены** (Allow external users to create OAuth2 tokens) – разрешение генерации внешними пользователями собственных токенов OAuth2.

    Возможные значения:

    * Включенный (Enable) – разрешено;
    * Неактивный (Disable) – запрещено.

    Примечание

    Существующие токены сохраняются при отключении параметра.

- **Controller sso url** – адрес, используемый для запуска аутентификации Controller SSO, настроенной в предыдущих версиях Astra Automation.
- **Automation hub sso url** – адрес, используемый для запуска аутентификации Automation Hub SSO, настроенной в предыдущих версиях Astra Automation.

Редактирование

Для изменения действующих настроек шлюза в окне **Шлюз платформы** (Platform gateway) выполните следующие действия:

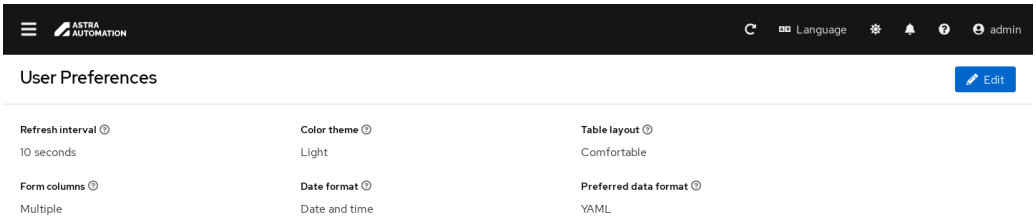
1. Нажмите кнопку *Изменить настройки шлюза платформы* (Edit platform gateway settings).
2. Внесите необходимые изменения.
3. Нажмите кнопку *Сохранить настройки шлюза платформы* (Save platform gateway settings).

Пользовательские настройки

Окно **Пользовательские настройки** (User Preferences) позволяет настраивать интерфейс и рабочую среду:

- управление форматами отображения даты и времени;
- выбор темы оформления и параметров локализации;
- управление персональными параметрами интерфейса и уведомлений.

Для перехода к окну **Пользовательские настройки** (User Preferences) выберите на панели навигации *Настройки* ▸ *Пользовательские настройки* (*Settings* ▸ *User Preferences*).



Получение информации

В окне просмотра информации о пользовательских настройках выводится следующая информация:

- **Период обновления** (Refresh interval) – время между автоматическим обновлением данных на странице (в секундах).

  Возможные значения:

  - 5 секунд (5 seconds);
  - 10 секунд (10 seconds);
  - 30 секунд (30 seconds);
  - 1 минута (1 minute);
  - 5 минут (5 minutes);
  - Никогда (Never).

- **Цветовая схема** (Color theme) – цветовая тема оформления графического интерфейса.

  Возможные значения:

  - Светлая (Light) – светлая тема;
  - Темная (Dark) – темная тема;
  - Система (System) – тема, используемая ОС по умолчанию.

- **Макет таблицы** (Table layout) – режим представления таблиц.

  Возможные значения:

  - Компактное (Compact) – уменьшение интервалов между элементами таблицы для более компактного вида;
  - Удобное (Comfortable) – увеличение расстояния между элементами таблицы для более удобного восприятия.

- **Столбцы формы** (Form columns) – отображение информации в одном или нескольких столбцах.

  Возможные значения:

  - Несколько (Multiple) – отображение формы и деталей в нескольких столбцах;
  - Одиночный (Single) – отображение формы и деталей в одном столбце.

- **Формат даты** (Date format) – предпочтительный формат даты для отображения в таблицах и деталях.

  Возможные значения:

  - Дата и время (Date and time) – точные дата и время;
  - Относительно (Relative) – отображение даты и времени относительно текущего момента (например, 10 минут назад).

- **Предпочтительный формат данных** (Preferred data format) – предпочтительный формат данных для редактирования и отображения данных.

  Возможные значения:

  - YAML;
  - JSON.

Редактирование

Для изменения действующих пользовательских настроек в окне **Пользовательские настройки** (User Preferences) выполните следующие действия:

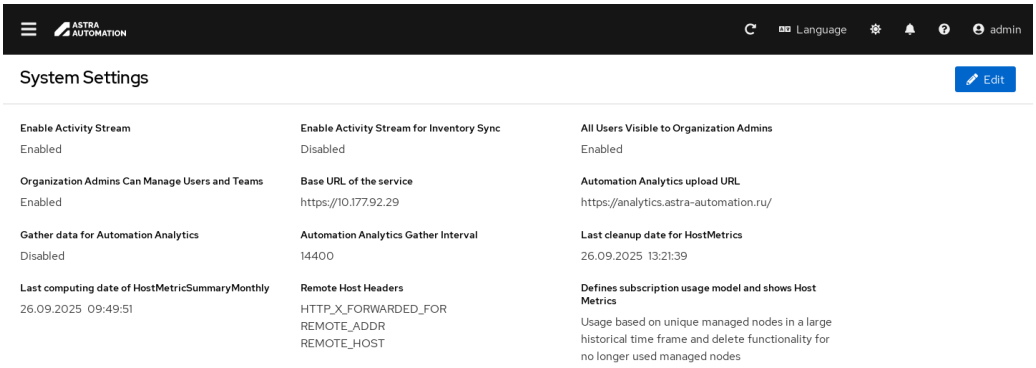
1. Нажмите кнопку *Редактировать* (Edit).
2. Внесите необходимые изменения.
3. Нажмите кнопку *Сохранить настройки* (Save user preference).

Система

Окно **Система** (System) предоставляет средства управления базовыми системными параметрами платформы:

- настройка параметров идентификации и сетевых атрибутов системы;
- управление глобальными переменными и общесистемными опциями.

Для перехода к окну **Система** (System) выберите на панели навигации *Настройки* ▸ *Система* (*Settings* ▸ *System*).



Получение информации

В окне просмотра информации о системе выводится информация об общих параметрах системы. Отображаются только заданные настройки. Для их редактирования воспользуйтесь *инструкцией*.

Полный список параметров:

- **Base URL of the service** – URL, используемый при создании ссылок на страницу платформы, в том числе при настройке аутентификации SAML. Укажите в этом поле URL, используя доменное имя, назначенное шлюзу платформы, например: https://tower.example.com.

  При использовании балансировщика нагрузки укажите его URL.

- **Proxy IP Allowed List** – список IP-адресов разрешенных прокси-серверов или балансировщика нагрузки. При значении `[]` запрещено использование прокси-серверов или балансировщика нагрузки.

  Значение по умолчанию: `[]`.

- **CSRF Trusted Origins List** – настройка адресов в формате `schema://addresses`, которым сервис должен доверять при проверке значений заголовка Origin, если сервис работает через обратный прокси-сервер или балансировщик нагрузки.

- **Astra Automation customer username** – название учетной записи, используемой для подключения к серверу Automation Analytics.

- **Astra Automation customer password** – пароль учетной записи, используемый для подключения к серверу Automation Analytics.

- **Astra Automation or Satellite username** – название учетной записи, используемой для получения информации о подписке и контенте.

- **Astra Automation or Satellite password** – пароль учетной записи, используемой для получения информации о подписке и контенте.

- **Automation Analytics upload URL** – URL для передачи информации, собранной службой Automation Analytics.

- **Global default execution environment** – среда исполнения, используемая по умолчанию.

- **Custom virtual environment path** – дополнительные пути для поиска сред исполнения (кроме `/var/lib/awx/venv/`).

- **Last gather date for Automation Analytics** – дата и время последнего сбора данных для Automation Analytics.

- **Last gathered entries from the data collection service of Automation Analytics** – последние записи из сервиса сбора данных Automation Analytics.

- **Automation Analytics Gather Inverval** – период (в секундах) отправки информации, собранной службой Automation Analytics.

  Значение по умолчанию: `14400` (4 часа).

- **Last cleanup date for HostMetrics** – дата и время последней очистки журнала метрик узлов.

- **Last computing date of HostMetricSummaryMonthly** – дата и время последнего расчета месячного отчета метрик узлов.

- **Remote Host Headers** – заголовки HTTP и мета-ключи, используемые для определения названия или IP-адреса внешнего узла. Если доступ к платформе осуществляется через обратный прокси-сервер, добавьте в список дополнительные заголовки, например, `HTTP_X_FORWARDED_FOR`.

  Значение по умолчанию:

```
- REMOTE_ADDR
- REMOTE_HOST
```

- **Defines subscription usage model and shows Host Metrics** – модель использования подписки и отображения метрики узлов.

  Возможные значения:

  - `No subscription. Deletion of host_metrics will not be considered for purposes of managed host counting` – без подписки; удаление метрик узлов не влияет на подсчет управляемых узлов.
  - `Usage based on unique managed nodes in a large historical time frame and delete functionality for no longer used managed nodes` – расчет по уникальным узлам за период с удалением неиспользуемых.

- **Опции** (Options):
  - **Enable Activity Stream** – если эта настройка включена, ведется поток активности.

    Значение по умолчанию: включено.
  - **Enable Activity Stream for Inventory Sync** – если эта настройка включена, в ленте активности выводятся сведения о событиях синхронизации инвентаря.

    Значение по умолчанию: выключено.
  - **All Users Visible to Organization Admins** – если эта настройка включена, администраторы организаций имеют возможность управлять пользователями других организаций.

    Значение по умолчанию: включено.
  - **Organization Admins Can Manage Users and Teams** – если эта настройка включена, администраторы организаций могут управлять своими пользователями и командами.

    Значение по умолчанию: выключено.
  - **Gather data for Automation Analytics** – сбор данных для сервиса Automation Analytics.

    Значение по умолчанию: выключено.

Редактирование

Для изменения действующих настроек системы в окне **Система** (System) выполните следующие действия:

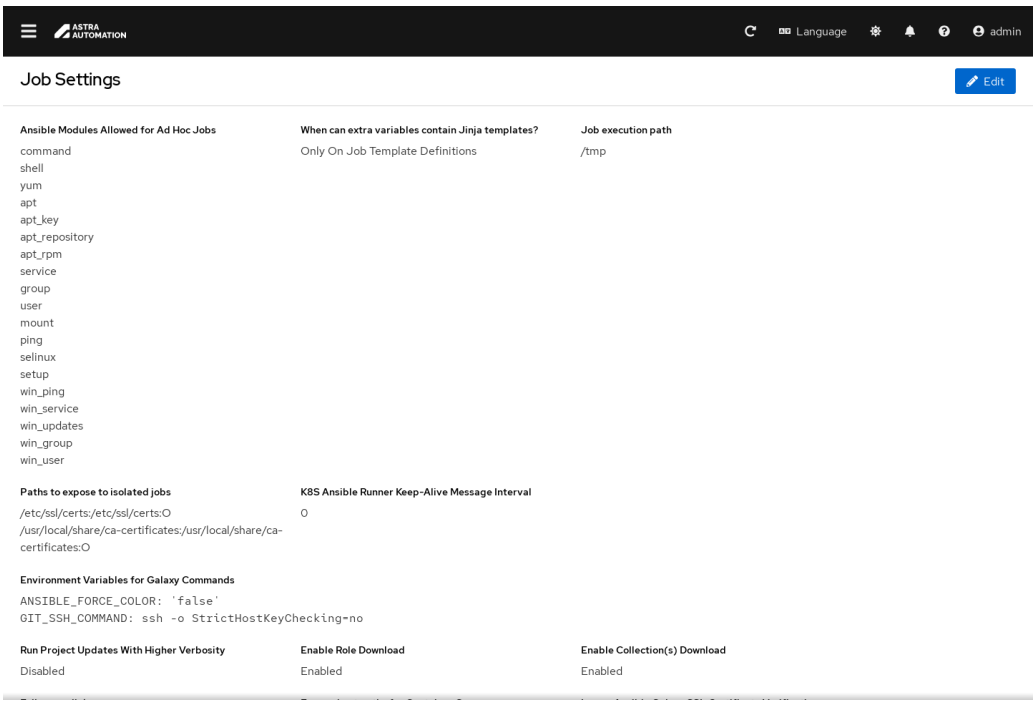
1. Нажмите кнопку *Редактировать* (Edit).
2. Внесите необходимые изменения.
3. Нажмите кнопку *Сохранить* (Save).

Настройка заданий

Окно **Настройка заданий** (Job Settings) определяет глобальные параметры выполнения задач автоматизации:

- настройка стратегии запуска и параллельности выполнения;
- управление временем ожидания, политиками повторного запуска и поведением при сбоях;
- задание параметров журналирования и сохранения артефактов выполнения.

Для перехода к окну **Настройка заданий** (Job Settings) выберите на панели навигации *Настройки* ▸ *Задание* (*Settings* ▸ *Job*).



Получение информации

В окне просмотра информации отображается информация о настройках, которые управляют запуском и исполнением всех заданий. Отображаются только заданные настройки. Для их редактирования воспользуйтесь *инструкцией*.

Полный список параметров:

- **Ansible Modules Allowed for Ad Hoc Jobs** – список модулей Ansible, которые разрешено использовать при выполнении отдельных (ad-hoc) команд.

  Значение по умолчанию:

```
- command
- shell
- yum
- apt
- apt_key
- apt_repository
- apt_rpm
- service
- group
- user
- mount
- ping
- selinux
- setup
- win_ping
- win_service
- win_updates
- win_group
- win_user
```

- **When can extra variables contain Jinja templates?** – разрешение на использование шаблонов Jinja2 в дополнительных переменных.

  Возможные значения:

  - `Always` – всегда;
  - `Never` – никогда;
  - `Only On Job Template Definition` – только в шаблоне задания;

Предупреждение

Для этой настройки рекомендуется использовать значение `Never` или `Only On Job Template Definition`. Это связано с тем, что шаблоны Jinja2 потенциально могут быть использованы для запуска произвольного кода на языке Python, что создает угрозу безопасности.

Значение по умолчанию: `Only On Job Template Definition`.

- **Job execution path** – путь к каталогу, в котором контроллер должен создать временный подкаталог для хранения данных, связанных с заданием, например, файлы полномочий.

  Значение по умолчанию: `/tmp`.

- **Paths to expose to isolated jobs** – список путей, монтируемых на исполняющих узлах в контейнеры как отдельные тома, по одному на строку. Пути, не указанные в этом списке, недоступны в контейнерах и не могут быть использованы при выполнении заданий.

  Формат записей:

```
<HOST_DIR>[:<CONTAINER_DIR>[:<OPTIONS>]]
```

  Здесь:

  - <HOST_DIR> – путь к каталогу в файловой системе исполняющего узла;
  - <CONTAINER_DIR> – точка монтирования каталога в файловой системе контейнера;
  - <OPTIONS> – опциональные параметры монтирования.

  Значение по умолчанию:

```
- /etc/ssl/certs:/etc/ssl/certs:O
- /usr/local/share/ca-certificates:/usr/local/share/ca-certificates:O
```

- **Extra Environment Variables** – дополнительные переменные среды, заданные для запуска сценариев, обновления инвентаря, обновления проектов и отправки уведомлений.
- **K8S Ansible Runner Keep-Alive Message Interval** – интервал в секундах, через который узлам из группы контейнеров отправляется сообщение для поддержания соединения. При значении `0` сообщения не отправляются.

  Значение по умолчанию: `0`.

- **Environment Variables for Galaxy Commands** – дополнительные переменные окружения, используемые при вызове команды `ansible-galaxy` для обновления проекта. Использование этих переменных может быть полезно, если команда `ansible-galaxy` работает через прокси-сервер, но не Git.

  Значение по умолчанию:

```
{
  "ANSIBLE_FORCE_COLOR": "false",
  "GIT_SSH_COMMAND": "ssh -o StrictHostKeyChecking=no"
}
```

- **Standard Output Maximum Display Size** – максимальный размер (в байтах) вывода в стандартный поток (stdout), отображаемого в графического интерфейсе.

  Значение по умолчанию: `1048576` (1 МБ).

- **Job Event Standard Output Maximum Display Size** – максимальный размер (в байтах) вывода одного события в стандартный поток (stdout), отображаемого в графического интерфейсе.

Значение по умолчанию: 1024.

- **Job Event Maximum Websocket Messages Per Second** – максимальное количество сообщений в секунду о событиях, отправляемых через WebSocket.

  Значение по умолчанию: 30.

- **Maximum Scheduled Jobs** – количество заданий на основе одного и того же шаблона, которые можно поставить в очередь выполнения. Если указанное значение будет достигнуто, создавать новые задания будет нельзя.

  Значение по умолчанию: 10.

- **Ansible Callback Plugins** – список путей для поиска расширений обратного вызова Ansible, по одному на строку. При значении [] расширения обратного вызова не используются.

  Значение по умолчанию: [].

- **Default Job Timeout** – время в секундах, отведенное на исполнение каждого задания. Значение этой настройки может быть переопределено в настройках шаблона задания. При значении 0 задания выполняются без ограничений по времени.

  Значение по умолчанию: 0.

- **Default Job Idle Timeout** – если за указанный период времени в секундах не происходит вывода Ansible, исполнение задания прерывается. При значении 0 задания могут выполняться без вывода неограниченно долго.

  Значение по умолчанию: 0.

- **Default Inventory Update Timeout** – время в секундах, отведенное на выполнение задачи обновления инвентаря из внешнего источника. Значение этой настройки может быть переопределено в настройках инвентаря. При значении 0 задания обновления инвентаря выполняются без ограничений по времени.

  Значение по умолчанию: 0.

- **Default Project Update Timeout** – время в секундах, отведенное на выполнение задачи обновления проекта из внешнего источника. Значение этой настройки может быть переопределено в настройках проекта. При значении 0 задания обновления проекта выполняются без ограничений по времени.

  Значение по умолчанию: 0.

- **Per-Host Ansible Fact Cache Timeout** – период времени в секундах, в течение которого факты Ansible считаются действительными с момента их последнего изменения. В наборе сценариев будут доступны только действительные, не устаревшие факты. При значении 0 факты не устаревают.

  Важно

  Эта настройка не влияет на удаление фактов в базе данных Automation Controller.

  Значение по умолчанию: 0.

- **Maximum number of forks per job** – максимально допустимое количество ответвленных процессов, которое можно задать в настройках шаблона задания. Попытка сохранить шаблон задания со значением больше указанного приведет к ошибке.

  Значение по умолчанию: 200.

- **Container Run Options** – список параметров для передачи Podman при запуске контейнера.

  Значение по умолчанию:

```
- '--network'
- slirp4netns:enable_ipv6=true
```

- **Run Project Updates With Higher Verbosity** – если эта настройка включена, используется более детализированный вывод при выполнении заданий обновления проектов. Использование этой настройки эквивалентно выполнению команды `ansible-playbook project_update.yml` с параметром `-vvv`.

  Значение по умолчанию: выключено.

- **Enable Role Download** – если эта настройка включена и источником кода проекта является система контроля версий, разрешена загрузка ролей, перечисленных в файле проекта `requirements.yml`.

  Значение по умолчанию: включено.

- **Enable Collection(s) Download** – если эта настройка включена и источником кода проекта является система контроля версий, разрешена загрузка коллекций, перечисленных в файле проекта `requirements.yml`.

  Значение по умолчанию: включено.

- **Follow symlinks** – если эта настройка включена, при обработке сценариев разрешен переход по символическим ссылкам.

  Предупреждение

  Переход по ссылкам может привести к бесконечной рекурсии, если ссылка указывает на свой родительский каталог.

  Значение по умолчанию: выключено.

- **Expose host paths for Container Groups** – если эта настройка включена, разрешено использование в подах переменных `hostPath`, указывающих на каталоги исполняющего узла, для монтирования этих каталогов в качестве томов к контейнерам среды исполнения.

  Предупреждение

  Использование томов `hostPath` содержит потенциальные риски безопасности. Рекомендуется избегать использования `hostPath`, где это возможно.

  Значение по умолчанию: выключено.

- **Ingore Ansible Galaxy SSL Certificate Verification** – если эта настройка включена, при подключении к реестрам коллекций Ansible не выполняется проверка сертификата TLS.

  Значение по умолчанию: выключено (проверка выполняется).

Редактирование

Для изменения действующих настроек заданий в окне **Задания** (Job) выполните следующие действия:

1. Нажмите кнопку *Редактировать* (Edit).
2. Внесите необходимые изменения.
3. Нажмите кнопку *Сохранить* (Save).

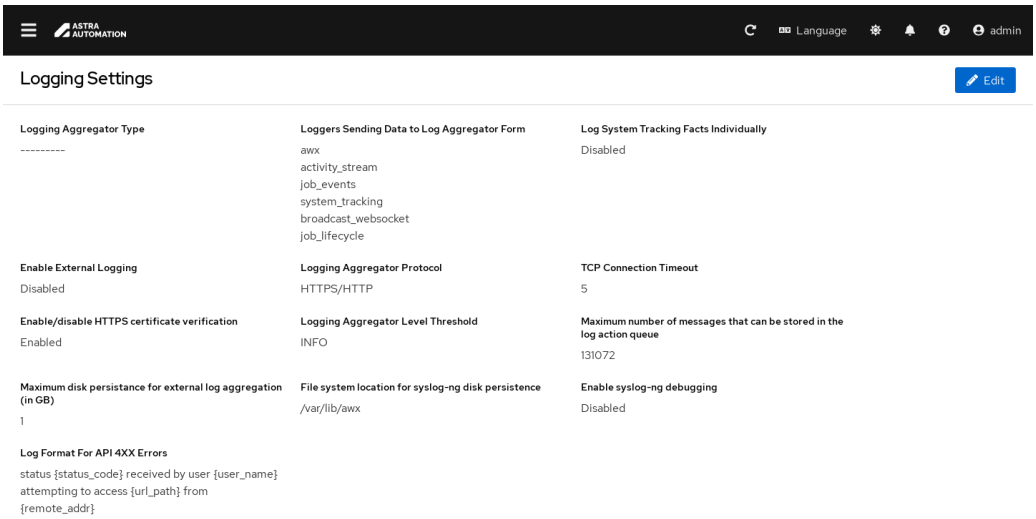
Управление журналами

Окно **Настройка ведения журнала** (Logging Settings) позволяет управлять параметрами журналирования платформы:

- настройка источников и уровня детализации журналов;
- настройка интеграции с внешними системами учета событий;

- определение формата журнала и параметров отправки в удаленные хранилища.

Для перехода к окну **Настройка ведения журнала** (Logging Settings) выберите на панели навигации *Настройки* ▸ *Регистрация* (*Settings* ▸ *Logging*).



Получение информации

В окне просмотра информации отображается информация о настройках, управляющих ведением журнала. Отображаются только заданные настройки. Для их редактирования воспользуйтесь *инструкцией*.

Полный список параметров:

- **Logging Aggregator** – FQDN или IP-адрес сервера, на котором размещена внешняя система журналирования.
- **Logging Aggregator Port** – номер порта, через который осуществляется подключение к внешней системе журналирования.
- **Logging Aggregator Type** – тип внешней системы журналирования.

  Возможные значения:

  - `---------` – внутренняя система журналирования;
  - `logstash` – Elastic Logstash;
  - `splunk` – Splunk;
  - `loggly` – Solarwinds Loggly;
  - `sumologic` – Sumo Logic;
  - `other` – внешняя система журналирования с настройками, заданными вручную.
- **Logging Aggregator Username** – название учетной записи, используемой для доступа к внешней системе журналирования.
- **Logging Aggregator Password/Token** – пароль или токен для доступа к внешней системе журналирования.
- **Loggers Sending Data to Log Aggregator Form** – список типов регистраторов, отправляющих данные в агрегатор журналов.

  Возможные значения:

  - `awx` – внутренние события контроллера.
  - `activity_stream` – изменение ресурсов.
  - `job_events` – события, связанные с заданиями.
  - `system_tracking` – факты, собранные при выполнении заданий.

Примечание

Для сбора фактов необходимо включить настройку **Включить хранилище фактов** (Enable Fact Storage) в свойствах шаблона задания.

  - `broadcast_websocket` – ошибки широковещательных рассылок веб-сокетов.
  - `job_lifecycle` – ошибки, связанные с обработкой заданий.

Значение по умолчанию:

```
- awx
- activity_stream
- job_events
- system_tracking
- broadcast_websocket
- job_lifecycle
```

- **Cluster-wide unique identifier** – уникальный идентификатор кластера.
- **Logging Aggregator Protocol** – протокол соединения с внешней системой журналирования.

  Возможные значения:

  - `HTTPS/HTTP`.

Примечание

При выборе этого значения используется протокол HTTPS, если только протокол HTTP не указан явно в значении настройки **Logging Aggregator**.

  - `TCP`.
  - `UDP`.

  Значение по умолчанию: `HTTPS/HTTP`.
- **TCP Connection Timeout** – время в секундах, по истечении которого соединение с внешней системой журналирования разрывается. Значение этой настройки используется только, если для настройки **Logging Aggregator Protocol** задано значение `HTTPS/HTTP` или `TCP`.

  Значение по умолчанию: 5.
- **Logging Aggregator Level Threshold** – минимальный уровень сообщений, которые записываются в журнал. Уровни сообщений, от наименьшего к наибольшему:

  1. `DEBUG`;
  2. `INFO`;
  3. `WARNING`;
  4. `ERROR`;
  5. `CRITICAL`.

  Сообщения с уровнем ниже указанного будут пропущены обработчиком журнала.

  Значение по умолчанию: `INFO`.

Примечание

Настройка **Logging Aggregator Level Threshold** применяется только к регистраторам, поддерживающим динамическое изменение уровня журналирования.

Настройку уровня поддерживают следующие регистраторы:

- awx – любой уровень;
- broadcast_websocket – любой уровень.

Для следующих регистраторов используется фиксированный уровень, заданный системой, независимо от значения настройки:

- activity_stream – INFO;
- job_events – INFO;
- system_tracking – INFO;
- job_lifecycle – DEBUG.

- **Maximum number of messages that can be stored in the log action queue** – максимальное количество сообщений, которые могут храниться в очереди действий журнала.

  Увеличение значения этого параметра может повлиять на использование памяти. Когда очередь достигнет 75% от этого числа, она начнет записываться на диск. Когда очередь достигнет 90%, сообщения уровней NOTICE, INFO и DEBUG начнут отбрасываться.

- **Maximum disk persistance for external log aggregation (in GB)** – максимальный размер дискового пространства для внешней агрегации журналов (в ГБ).

  Представляет объем данных для хранения на случай сбоя внешнего агрегатора журналов.

  Значение по умолчанию: 1.

- **File system location for syslog-ng disk persistence** – расположение, где временно сохраняются журналы, если не могут быть переданы во внешний агрегатор. Когда соединение восстанавливается, передача автоматически продолжается (включая журналы, временно сохраненные в указанном каталоге).

  Значение по умолчанию: /var/lib/awx.

- **Log Format For API 4XX Errors** – шаблон записей о событиях, связанных с кодами HTTP 4XX при обращении к точкам доступа API.

  В шаблоне поддерживается использование следующих переменных:

  - status_code – код статуса HTTP;
  - user_name – название учетной записи пользователя, выполнившего запрос к точке доступа API;
  - url_path – URL точки доступа API, обращение к которой привело к ошибке;
  - remote_addr – адрес внешнего узла, выполнявшего запрос;
  - error – сообщение об ошибке, возвращенное точкой доступа API.

  Если точка доступа API не возвращает собственный текст сообщения об ошибке, в журнал записывается стандартное описание статуса HTTP, соответствующее коду ответа. Например, при обращении к несуществующему ресурсу (код 404) в журнал будет записано:

```
status_code=404 user_name=jdoe url_path=/api/v2/nonexistent/ remote_addr=192.168.1.
↪10 error="Not Found"
```

  А при ошибке аутентификации (код 401):

```
status_code=401 user_name=anonymous url_path=/api/v2/jobs/ remote_addr=203.0.113.5␣
↪error="Unauthorized"
```

- **Log System Tracking Facts Individually** – если этот параметр включен, факты, отслеживаемые системой, будут регистрироваться в журнале для каждого пакета, сервиса или другой сущности, найденной при сканировании, что обеспечивает глубокую детализацию поискового запроса. Если этот параметр выключен, факты будут отправляться в виде единого словаря, что обеспечивает большую эффективность обработки фактов.

  Значение по умолчанию: выключено.

- **Enable External Logging** – если эта настройка включена, для хранения журналов используется внешняя система журналирования.

  Примечание

  Для включения этой настройки необходимо задать значения настроек **Logging Aggregator** и **Logging Aggregator Type**.

- **Enable/disable HTTPS certificate verification** – если эта настройка включена, при соединении с внешней системой журналирования по протоколу HTTPS выполняется проверка сертификата.

  Значение по умолчанию: включено.

- **Enable syslog-ng debugging** – если этот параметр включен, детализация журналирования работы `syslog-ng` повышается, чтобы диагностировать проблемы с подключением и доставкой журналов во внешние системы.

Редактирование

Для изменения действующих настроек ведения журнала в окне **Настройка ведения журнала** (Logging Settings) выполните следующие действия:

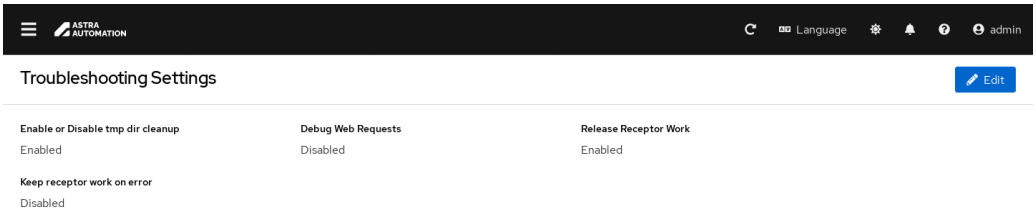
1. Нажмите кнопку *Редактировать* (Edit).
2. Внесите необходимые изменения.
3. Нажмите кнопку *Сохранить* (Save).

Поиск неисправностей

Окно **Параметры устранения неполадок** (Troubleshooting Settings) позволяет активировать дополнительные режимы диагностики для выявления и расследования проблем в работе платформы автоматизации:

- управление очисткой временного каталога;
- включение трассировки веб-запросов;
- контроль освобождения рабочих каталогов рецептора;
- сохранение рабочих данных рецептора при ошибках.

Для перехода к окну **Параметры устранения неполадок** (Troubleshooting Settings) выберите на панели навигации *Настройки* ▸ *Поиск неисправностей* (*Settings* ▸ *Troubleshooting*).



Получение информации

В окне просмотра информации о настройках отладки выводится следующая информация:

- **Enable or Disable tmp dir cleanup** – если эта настройка включена, происходит автоматическая очистка временных каталогов, используемых платформой. Это предотвращает накопление временных файлов, которые могут занимать значительное дисковое пространство. Отключение настройки может быть полезно для отладки и анализа временных файлов, но требует ручного управления их удалением.

  Значение по умолчанию: включено.

- **Debug Web Requests** – если эта настройка включена, происходит запись в журнал сообщений о всех веб-запросах. Это полезно при диагностике и анализе проблем, связанных с веб-запросами. Отключение настройки снижает нагрузку на систему.

  Значение по умолчанию: выключено.

- **Release Receptor Work** – если эта настройка включена, служба рецепторов освобождает ресурсы по мере завершения заданий.

  Значение по умолчанию: включено.

- **Keep receptor work on error** – если эта настройка включена, временные данные, созданные службой рецепторов при выполнении задания, не удаляются при возникновении ошибки. Это позволяет сохранить информацию для последующего анализа и отладки. При отключенной настройке данные автоматически очищаются после завершения задания.

  Значение по умолчанию: выключено.

Редактирование

Для изменения действующих настроек в окне **Параметры устранения неполадок** (Troubleshooting Settings) выполните следующие действия:

1. Нажмите кнопку *Редактировать* (Edit).
2. Внесите необходимые изменения.
3. Нажмите кнопку *Сохранить* (Save).

ГЛАВА 15

Средства аналитики

Важно

Эта часть документации находится в стадии разработки.

ГЛАВА 16

Обеспечение безопасности

Для повышения уровня защищенности платформы следует учитывать и применять ряд рекомендуемых мер защиты при развертывании, настройке и технической поддержке.

16.1 Критерии безопасности

Безопасность Astra Automation основана на фундаментальной модели информационной безопасности, известной как триада CIA (Confidentiality, Integrity, Availability). Эта модель определяет три взаимосвязанные принципа, которые должны быть реализованы для защиты автоматизированной инфраструктуры и данных, обрабатываемых платформой.

Применение требований CIA триады в контексте Astra Automation обеспечивает построение защищенной автоматизированной среды, предотвращению несанкционированного доступа, защите от изменения данных и обеспечению непрерывной доступности критических сервисов.

16.2 Конфиденциальность (Confidentiality)

Конфиденциальность гарантирует, что чувствительная информация доступна только авторизованным пользователям и процессам. Это означает защиту от несанкционированного доступа к данным, включая учетные данные, ключи доступа, персональные данные и конфиденциальную бизнес-информацию.

В контексте Astra Automation конфиденциальность охватывает следующие компоненты и действия:

- Учетные данные:
  - Все учетные данные (пароли SSH, облачные ключи, токены API) хранятся в зашифрованном виде в базе данных.
  - Используется шифрование AES-256 в режиме CBC с добавлением HMAC SHA256 для аутентификации передаваемых данных.
  - Секреты автоматизации никогда не экспортируются через REST API платформы, что предотвращает их случайное разглашение.
- Переменные и чувствительные данные:
  - Использование Ansible Vault для шифрования чувствительных переменных в файлах инвентаря установки.

  - Применение атрибута `no_log: True` в сценариях автоматизации для предотвращения записи в журналах конфиденциальной информации.
  - Обозначение чувствительных переменных как «безопасных» для исключения их из вывода JSON callback в сценариях автоматизации.
- Управление доступом:
  - Реализация Role-Based Access Control (RBAC) для ограничения доступа к объектам автоматизации на основе ролей пользователей.
  - Использование различных учетных данных для разных уровней автоматизации, что минимизирует риск от компрометации одного ключа.
  - Принцип наименьших привилегий (least privilege) при назначении прав доступа.
- Аутентификация и авторизация:
  - Поддержка множественных механизмов аутентификации: LDAP, SAML, TACACS+, Radius, OAuth2, OIDC, Keycloak и GitHub.
  - Двухфакторная аутентификация (2FA/MFA) для защиты учетных записей пользователей.
  - Использование ключей SSH вместо аутентификации по паролям для подключений между компонентами.
- Защита межкомпонентной коммуникации:
  - Применение TLS/SSL для всех веб-сервисов и взаимодействия между компонентами платформы.
  - Поддержка взаимной аутентификации TLS (mTLS) для коммуникации между сервисами и базой данных PostgreSQL.
  - Использование пользовательских сертификатов PKI вместо самоподписанных сертификатов по умолчанию для усиленной безопасности.
- Использование внешних хранилищ секретов:
  - Интеграция с корпоративными хранилищами секретов (например, HashiCorp Vault) для централизованного управления учетными данными.
  - Отделение учетных данных администраторов от учетных данных, используемых для автоматизации, что улучшает аудит.

16.3 Целостность (Integrity)

Целостность обеспечивает сохранность данных – данные остаются точными, полными и не изменяются несанкционированным образом, как случайно, так и намеренно. Это гарантирует надежность и достоверность информации на всем протяжении ее жизненного цикла.

Внедрение целостности в Astra Automation включает:

- Аудит:
  - Централизованная регистрация всех действий в платформе с использованием встроенного журнала активности (activity stream).
  - Регистрация всех изменений конфигурации, выполнение заданий, изменения пользователей и доступа.
  - Синхронизация времени через NTP на всех компонентах для обеспечения корректных временных меток событий в журналах.
- Криптографическая защита целостности:
  - Использование цифровых подписей для верификации подлинности журналов и предотвращения их подделки.
  - Применение контрольных сумм (checksums) и криптографических хешей для обнаружения несанкционированного изменения данных.

  - Защита инструментов ведения журналов от модификации путем применения хешей и цифровых подписей.
- Контроль версий и доступа:
  - Управление версиями для критических файлов конфигурации и scripts с целью обнаружения неавторизованных изменений.
  - Файловые разрешения (Unix permissions) для ограничения доступа к конфиденциальным файлам на уровне операционной системы.
  - Ограничение административного доступа только авторизованным персоналу.
- Взаимная аутентификация TLS:
  - Использование TLS для внутреннего взаимодействия между компонентами платформы, что предотвращает man-in-the-middle атаки.
  - Верификация подлинности как клиента, так и сервера при установлении соединений.
- Резервное копирование и восстановление:
  - Регулярное резервное копирование критических данных и конфигураций.
  - Шифрование резервных копий и безопасное хранение ключей восстановления.
  - Проверка целостности резервных копий перед восстановлением.

16.4 Доступность (Availability)

Доступность гарантирует, что авторизованные пользователи и процессы имеют надежный и своевременный доступ к информации и ресурсам, когда это необходимо. Это включает обеспечение функционирования систем без перерывов, а также возможность быстрого восстановления после сбоев.

Обеспечение доступности в Red Hat Ansible Automation Platform реализуется следующим образом:

- Архитектура высокой доступности:
  - Использование тестированных архитектур развертывания (tested reference architectures) для обеспечения максимальной надежности.
  - Реализация отказоустойчивых конфигураций для критических компонентов (Automation Controller, Private Automation Hub, Event-Driven Ansible).
  - Развертывание каждого компонента на нескольких узлах для распределения нагрузки и исключения единой точки отказа.
- Балансировка нагрузки:
  - Использование подсистем балансировки нагрузки (load balancers) для распределения входящих запросов между несколькими узлами компонентов.
  - Мониторинг доступности каждого узла и автоматическое перенаправление трафика при обнаружении проблем.
  - Конфигурирование DNS для масштабируемого доступа к компонентам платформы.
- Отказоустойчивость базы данных:
  - Поддержка высокодоступной конфигурации PostgreSQL 15 с репликацией.
  - Использование взаимной аутентификации TLS для защиты соединений с базой данных.
  - Возможность восстановления от сбоев базы данных без потери данных.
- Синхронизация и репликация состояния:

  - Использование механизма синхронизации состояния между компонентами (например, между контроллером и шлюзом платформы).
  - Обеспечение консистентности данных во всех реплицированных узлах.
  - Механизм восстановления при разрыве синхронизации.
- Мониторинг и оповещение:
  - Непрерывный мониторинг доступности всех компонентов платформы.
  - Система оповещений для критических событий и состояний.
  - Быстрое обнаружение аномалий и потенциальных проблем до того, как это повлияет на пользователей.
- Сетевая безопасность и доступность:
  - Ограничение доступа к критическим портам TCP (PostgreSQL порт 5432, SSH, HTTPS) только для доверенных сетей и клиентов
  - Использование сетевых экранов для защиты от DDoS-атак и несанкционированного доступа.
  - Контролируемый доступ через VPN или защищенные сетевые каналы.
- Планирование восстановления после сбоев (Disaster Recovery):
  - Разработка и тестирование планов восстановления для различных сценариев отказов.
  - Определение RTO (Recovery Time Objective) и RPO (Recovery Point Objective) в соответствии с требованиями бизнеса.
  - Регулярное проведение учений по восстановлению для проверки готовности.

16.5 Взаимодействие компонентов CIA

Три компонента CIA являются взаимозависимыми и часто входят в конфликт при реализации, например:

- Конфиденциальность и доступность:

  Строгие меры контроля доступа для защиты конфиденциальности могут затруднить легальный доступ авторизованных пользователей к требуемым ресурсам.
- Целостность и доступность:

  Жесткие проверки целостности и валидация данных могут замедлить обработку и снизить доступность системы.
- Конфиденциальность и целостность:

  Шифрование данных для конфиденциальности усложняет проверку целостности без дешифрования.

Astra Automation предоставляет гибкие инструменты и конфигурируемые параметры, позволяющие организациям находить баланс между этими требованиями в соответствии со своими специфическими потребностями и политиками безопасности.

16.5.1 Краткие рекомендации

Приведенные рекомендации предназначены для специалистов по внедрению, администрированию и эксплуатации Astra Automation и описывают рекомендуемые практики по обеспечению повышения уровня защищённости платформы при установке, настройке и обслуживании. Рекомендации учитывают отраслевые стандарты безопасности.

Примечание

Для достижения максимального уровня защищённости необходимо учесть корпоративные политики и провести дополнительный анализ рисков.

Соблюдение приведенных рекомендаций позволяет минимизировать риски несанкционированного доступа, компрометации данных и нарушения целостности платформы Astra Automation. Представленные рекомендации охватывают следующие ключевые аспекты:

- планирование развертывания:
- сетевые ограничения;
- аутентификацию и управление доступом;
- запись в журналы и мониторинг;
- настройку узлов;
- безопасность соединений;
- управление настройками системы;
- обновление и резервное копирование.

Планирование развертывания

Astra Automation является модульной системой с основными компонентами:

- контроллер автоматизации (Automation Controller);
- сеть автоматизации (Automation Mesh);
- частное хранилище контента (Private Automation Hub);
- контроллер автоматизации на основе событий (Event-Driven Controller);
- встроенная СУБД PostgreSQL (по желанию возможна внешняя СУБД).

Необходимо устанавливать все компоненты для получения полной функциональности и обеспечения масштабируемости и надёжности.

При планировании развертывания платформы (day-0) учитывайте следующие рекомендации:

- Для производственных систем используйте типовую топологию, протестированную разработчиками Astra Automation.
- Все серверы Astra Automation должны иметь корректные DNS-записи (FQDN).
- Для балансировки используйте отдельную запись DNS для балансировщика нагрузки.
- Все узлы должны синхронизировать время через NTP/Chrony.

Настройка узлов, развертывание и обслуживание

Следующие рекомендации помогут вам настроить узлы Astra Automation таким образом, чтобы минимизировать риски эксплуатации уязвимостей:

- Следуйте рекомендациям по безопасности Astra Linux.
- Отключайте все неиспользуемые сервисы.
- Используйте только разрешённое ПО, применяйте контроль доступа и сегментируйте уровни привилегий.
- Настройте узлы с помощью коллекции astra.hardening[212], как приведено в *примерах*.

[212] https://hub.astra-automation.ru/ui/repo/validated/astra/hardening/

- Совершайте развертывание платформы с выделенного сервера, который имеет доступ ко всем узлам контролера и будет работать далее для обслуживания с помощью `aa-setup` (обновление, резервное копирование). Доступ к этому узлу должен быть ограничен.
- При запуске `aa-setup` используйте аргумент `--ask-vault-pass` для защиты конфиденциальных данных.

Ограничения по периметру

Для защиты платформы Astra Automation по периметру важно ограничить сетевой доступ, оставив доступными только необходимые сервисы. Следующие рекомендации помогут снизить риск несанкционированного доступа и потенциальных атак на уязвимые службы:

- Оставьте открытыми для входящих подключений только необходимые порты TCP, как рекомендовано в *типовых топологиях*.
- Ограничьте прямой доступ по SSH к узлам Astra Automation.
- Разрешите доступ по HTTPS к компонентам системы только из доверенных сетей.

Аутентификация и управление доступом

Чтобы минимизировать риски компрометации учетных данных и защитить критические компоненты платформы Astra Automation от несанкционированного доступа, воспользуйтесь следующими рекомендациями:

- Используйте внешнюю аутентификацию (SSO/LDAPS) для всех учетных записей, включая сервисные. Рекомендуется оставить только одну локальную учетную запись администратора, например, `admin`, на случай аварий и недоступности SSO.
- Минимизируйте круг лиц с правами администратора.
- Разграничьте учетные данные для администраторов и автоматизации.
- Внедрите разноуровневые ключи и доступы для выполнения наборов сценариев с разными привилегиями.

Управление учётными данными и секретами

Платформа использует три класса секретных данных:

- пароли локальных пользователей Astra Automation;
- операционные секреты (пароли к БД, очередям сообщений и др.);
- секреты, используемые в автоматизации (ключи SSH, облачные токены и др.).

Рекомендации:

- Используйте внешние хранилища секретов для хранения чувствительных данных (например, HashiCorp Vault).
- Для передачи секретов в задачах Ansible используйте опцию `no_log: true` для предотвращения утечек в логах.
- Изолируйте права доступа к файлам секретов. Следите за изменениями и несанкционированным доступом.
- Используйте уникальные ключи для различных сценариев автоматизации.
- Запрещайте регистрацию в журналах конфиденциальных данных. Пример использования `no_log` в playbook:

```
name: Example sensitive task
ansible.builtin.command: /usr/bin/do_sensitive_thing
no_log: true
```

Аудит

Следующие рекомендации помогают оперативно выявлять и предотвращать инциденты, угрожающие безопасности системы:

- Используйте параметр `no_log` в местах, где обрабатываются конфиденциальные данные.
- Настройте централизованный сбор журналов.
- Используйте криптографически подписанные инструменты ведения журналов, чтобы защитить их от подмены и модификации.
- Обеспечьте регистрацию всех пользовательских сессий.

Обновления и резервное копирование

Следующие рекомендации позволят вам поддерживать актуальность программного обеспечения и минимизировать риски утраты данных в случае инцидентов или сбоев:

- Переходите не более чем на две основные версии Astra Automation за раз, предварительно ознакомившись с особенностями такого перехода для *конкретной версии*.
- Настройте регулярное резервное копирование.
- Храните резервные копии в надежном месте.

16.5.2 Сетевые настройки

Существуют различные способы атак и защиты в сети, которые учитывают в корпоративных инфраструктурах. Некоторые из них следует учитывать при настройке узлов контроллера.

Безопасность соединений

Следующие рекомендации помогут защитить передаваемые данные от перехвата и несанкционированного доступа, обеспечив целостность и конфиденциальность информации:

- Включите SSL для защиты подключений к СУБД PostgreSQL, добавив в инвентарь установщика платформы следующие переменные:

  INI

  ```
  [all:vars]
  pg_sslmod='verify-full'
  postgres_use_ssl='true'
  ```

  YAML

  ```
  all:
    vars:
      pg_sslmod: verify-full
      postgres_use_ssl: true
  ```

- Обеспечьте обязательное использование HTTPS, добавив в инвентарь установщика платформы следующие переменные:

  INI

  ```
  [all:vars]
  automationedacontroller_disable_https='false'
  automationhub_disable_https='false'
  nginx_disable_https='false'
  ```

YAML

```yaml
all:
  vars:
    automationedacontroller_disable_https: false
    automationhub_disable_https: false
    nginx_disable_https: false
```

- Используйте собственную инфраструктуру управления ключами (PKI, Public Key Infrastructure) для управления сертификатами и публичными ключами.

Использование специальных каналов связи

В контурах сети, где накладываются особые ограничения на связь с внешним миром, используют такие каналы связи, которые обеспечивают минимальный размер периметра возможных атак.

Особенности ограничений

Специальный канал связи типично включает следующие ограничения (примеры приведены применительно к Astra Automation):

- Разрешается подключение только к определенным серверам.
- Разрешается подключение только по определенному протоколу, например, по SSH.
- Подключение разрешено определенному пользователю.

Реализация такого канала для контроллера требует учета следующих особенностей:

- Сервисы контроллера работают с привилегиями пользователя awx.
- Синхронизация проектов контроллера с Private Automation Hub или другими реестрами происходит с использованием той системной среды исполнения (EE), которая добавляется к контроллеру в процессе его установки. Системная среда исполнения использует по умолчанию образ `aa-full-ee`, версия которого может изменяться вместе с версией контроллера.

> **Примечание**
>
> Добавление какой-либо другой EE в настройках проекта влияет только на исполнение сценариев автоматизации, но не на синхронизацию проектов с репозиториями.

Пример настройки

Предположим, что выход в интернет по SSH должен происходить через прокси-сервер с адресом 10.111.222.1 по порту 2022. Для установления связи через прокси используется универсальная утилита ncat[213].

> **Примечание**
>
> Это внешняя утилита, которая должна выполняться внутри EE, однако она не установлена там по умолчанию.

Для использования утилиты `ncat` необходимо создать собственный образ EE и использовать его в качестве системного. Соответственно процесс состоит из следующих шагов:

1. *Создайте собственный образ EE*, включающий утилиту `ncat`, с помощью Ansible Builder.
2. Обеспечьте использование собственного образа EE в качестве системного при развертывании платформы. Для этого воспользуйтесь дополнительной переменной Ansible

[213] https://nmap.org/ncat/

control_plane_execution_environment, значением которой должен быть URL требуемого образа EE, например:

```
./aa-setup -- -e control_plane_execution_environment=private-hub.example.com/ee/
↪image:<version>
```

Для подключения контроллера к внешнему серверу по SSH выполните следующие настройки в контроллере:

1. В каталоге /var/lib/awx/.ssh/ создайте файл config со следующим содержимым:

   ```
   Host <address>
      ProxyCommand ncat --proxy 10.111.222.1:2022 %h %p
      PreferredAuthentications publickey
      IdentityFile /var/lib/awx/.ssh/id_key
   ```

   Особенности настройки:

   - Подключение к прокси происходит с помощью утилиты ncat.
   - IdentityFile ссылается на приватный ключ, публичная часть которого должна быть расположена на сервере, указанном в параметре Host.

2. Обеспечьте подключение к EE тех компонентов, которые в нем отсутствуют. Это необходимо выполнить через графический интерфейс контроллера.

   Путь навигации: **Настройки** (Settings) > **Настройки заданий** (Jobs) > **Пути доступа к изолированным заданиям** (Paths to expose to isolated jobs).

   Дополните список монтирования следующими парами:

   ```
   "/var/lib/awx/.ssh/config:/root/.ssh/config:ro",
   "/var/lib/awx/.ssh/id_key:/root/.ssh/id_key:ro".
   ```

   В этом примере в EE попадут следующие недостающие компоненты:

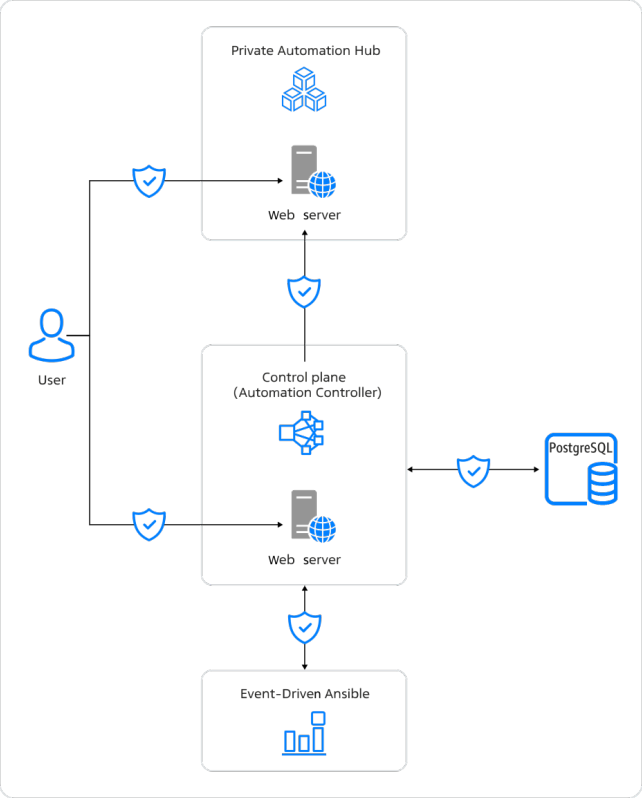
   - файл настроек SSH config,
   - приватный ключ id_key.

С использованием приведенных настроек контроллер будет иметь доступ к Automation Hub через указанный прокси-сервер для получения необходимого инфраструктурного кода.

16.5.3 Сертификаты TLS/SSL

При работе платформы в продуктовой среде для защиты подключения необходимо использовать сертификаты, выданные удостоверяющим центром. Они необходимы для защиты следующих соединений:

- пользователя с веб-сервером Automation Controller;
- пользователя с веб-сервером Private Automation Hub;
- Automation Controller с Private Automation Hub;
- Automation Controller с контроллером Event-Driven Automation;
- Automation Controller с сервером СУБД PostgreSQL, развернутым средствами Astra Automation;
- между узлами кластера Automation Controller (сеть Mesh).



Если Automation Controller или Private Automation Hub развернуты на нескольких узлах, но доступ к ним осуществляется через балансировщики нагрузки, в поле сертификата Common Name (CN) должно быть указано *FQDN* соответствующего балансировщика, а не узлов Automation Controller или Private Automation Hub.

Допускается использовать Wildcard-сертификат для узлов, находящихся в одном домене.

Если политики организации требуют использования уникального сертификата для каждого компонента Astra Automation, необходимо соблюдение следующих условий:

- Если используется балансировщик нагрузки, в данных каждого сертификата должно присутствовать поле Subject Alt Name (SAN), указывающее на FQDN балансировщика.
- На каждый узел следует установить соответствующий сертификат.

Пошаговые инструкции по управлению сертификатами приведены в документе plan_vm_inventory_sso.

16.5.4 Типовые инциденты ИБ

Этот документ описывает наиболее распространенные инциденты информационной безопасности (ИБ) в Astra Automation, признаки их проявления и рекомендуемые процедуры реагирования.

Компрометация секретов и учетных данных

Один из наиболее опасных инцидентов в системе – компрометация секретов или учетных данных пользователей. Она может привести к несанкционированному доступу к критическим ресурсам и выполнению нежелательных действий от имени легитимных пользователей.

Следующие признаки указывают на компрометацию секретов и учетных данных:

- неудачные попытки аутентификации с известных учетных записей (события в журнале контроллера);
- успешная аутентификация с подозрительных IP-адресов или в нехарактерное время;
- создание новых пользователей или изменение привилегий без соответствующих заявок;

- выполнение нехарактерных для пользователя операций.

При обнаружении компрометации секретов и учетных данных выполните следующие действия:

1. Заблокируйте скомпрометированную учетную запись в системе аутентификации.
2. Принудительно сбросьте ключи и пароли у скомпрометированных учетных записей.
3. Обновите секреты в Ansible Vault, HashiCorp Vault или другом хранилище секретов.
4. Проанализируйте журналы активности пользователя.
5. Восстановите доступы.

Злоупотребление доверенными ролями и политиками доступа

Злоупотребление доверенными ролями или ошибочно настроенными политиками доступа может позволить злоумышленникам или недобросовестным пользователям выполнять операции с чрезмерными привилегиями, что угрожает безопасности всей платформы.

Следующие признаки указывают на злоупотребление доверенными ролями и политиками доступа:

- выполнение наборов сценариев с подозрительными операциями, например, удаление критических файлов, установка ПО;
- наборы сценариев, не соответствующие корпоративным стандартам автоматизации;
- неавторизованное выполнение наборов сценариев с повышенными привилегиями;
- массовое выполнение однотипных деструктивных операций.

При подозрении на злоупотребление доверенными ролями и политиками доступа выполните следующие действия:

1. Проведите мониторинг подозрительной активности и запусков наборов сценариев с повышенными привилегиями.
2. При обнаружении таких случаев немедленно ограничьте доступ подозрительным пользователям или группам и отзовите у них лишние права.
3. Проверьте, какие действия были совершены с повышенными привилегиями.
4. Проведите расследование инцидента.
5. Проведите внеочередной аудит ролей и политик и восстановите их корректность.

Несанкционированное изменение наборов сценариев

Наборы сценариев являются ключевыми элементами автоматизации. Их несанкционированное изменение может привести к внедрению вредоносного кода, нарушению рабочих процессов и прямым угрозам безопасности.

Следующие признаки указывают на несанкционированное изменение наборов сценариев:

- неавторизованные коммиты в репозитории проектов;
- изменение наборов сценариев без соответствующих процедур утверждения;
- обнаружение вредоносного кода в наборах сценариев при их анализе;
- несоответствие хеш-сумм файлов наборов сценариев эталонным значениям.

При обнаружении несанкционированного изменения наборов сценариев выполните следующие действия:

1. Заблокируйте синхронизацию проектов со скомпрометированным репозиторием.
2. Откатите изменения к последней известной безопасной версии.
3. Проанализируйте все измененные файлы на наличие вредоносного содержимого.
4. Идентифицируйте источник несанкционированных изменений.

5. Приостановите выполнение всех наборов сценариев из затронутого репозитория.

Нарушение ролевой модели доступа

Ролевая модель доступа (RBAC) определяет, какие действия пользователи могут выполнять в системе. Ее нарушение приводит к перераспределению полномочий и появлению избыточных прав у пользователей, что повышает риск злоупотреблений и атак.

Следующие признаки указывают на нарушение ролевой модели доступа:

- пользователи выполняют операции вне своих полномочий;
- несанкционированные изменения в настройках ролей и разрешений;
- доступ к проектам или инвентарю без соответствующих прав;
- создание пользователей с избыточными привилегиями.

При обнаружении нарушений выполните следующие действия:

1. Проведите аудит текущих разрешений всех пользователей и ролей.
2. Отзовите избыточные привилегии.
3. Заблокируйте учетные записи с подозрительной активностью.
4. Восстановите корректную конфигурацию RBAC из резервной копии.
5. Принудительно выполните повторную авторизацию всех пользователей.

Общие рекомендации для всех сценариев

Независимо от характера инцидента важно придерживаться унифицированного подхода к реагированию и последующему анализу. Это помогает минимизировать ущерб и повышает устойчивость платформы к атакам.

Следующие рекомендации применимы ко всем типам инцидентов:

- поддерживайте и регулярно тестируйте план реагирования на инциденты;
- внедрите автоматизацию для сбора доказательств, уведомлений и изоляции при инцидентах;
- проводите пост-инцидентный анализ и обновляйте процессы на основе прошлых инцидентов.

Такой подход позволит быстро реагировать на инциденты, минимизировать ущерб от них и повысить устойчивость платформы к атакам.

ГЛАВА 17

Ресурсы разработчика

Значимой частью Astra Automation является набор средств для разработки инфраструктурного кода, включая:

- сценарии;
- коллекции;
- образы служебных контейнеров.

Состав

Инструментальные средства для работы с Ansible делятся на два типа: основные и дополнительные. Основные инструменты распространяются через репозиторий Astra Automation в составе пакета `astra-automation-cdk`. Действия, необходимые для использования дополнительных инструментов могут различаться и описаны в справочнике каждого из инструментов.

Основные инструменты

В состав Astra Automation входят следующие основные инструментальные средства для работы с Ansible:

- *Ansible Builder* – создание образов среды исполнения.
- *Ansible Creator* – создание коллекций и наборов сценариев.
- *Ansible Lint* – статический анализ кода на наличие ошибок.
- *Ansible Molecule* – тестирование ролей.
- *Ansible Navigator* – создание, проверка и запуск контента, а также поиск и устранение проблем в его работе.
- *Ansible Pytest* – тестирование рабочих окружений с помощью pytest[214] до и после применения ролей.
- *Ansible Sign* – контроль целостности содержимого Ansible с помощью GPG.
- *Tox Ansible* – тестирование коллекций с различными версиями Python и Ansible.
- *Ansible Test* – тестирование кода перед его передачей в проект Ansible[215].

[214] https://docs.pytest.org/en/stable/index.html
[215] https://github.com/ansible

Дополнительные инструменты

В состав Astra Automation входят следующие дополнительные инструментальные средства для работы с Ansible:

- *export_collections.py*.

Установка

Для установки всех *основных* инструментов выполните следующие действия:

С доступом к интернету

1. Подключите репозиторий Astra Automation.

   #### Инструкция по подключению репозитория

   1. В каталоге `/etc/apt/sources.list.d/` создайте файл `astra-automation.list` со ссылкой на репозиторий Astra Automation:

      ```
      deb https://dl.astralinux.ru/aa/aa-debs-for-alse-1.8 <version> main
      ```

      Вместо <version> необходимо подставить версию устанавливаемой платформы, например, `2.0`.

      Доступные версии продукта опубликованы в таблице *История обновлений*.

   2. Обновите список доступных пакетов:

      ```
      sudo apt update
      ```

2. Установите пакет `astra-automation-cdk`:

   ```
   sudo apt install astra-automation-cdk --yes
   ```

Если необходима установка отдельных утилит, следуйте инструкции в описании соответствующей утилиты.

Без доступа к интернету (Offline Bundle)

1. Загрузите архив с установщиком на установочный узел.

   Примечание

   Архив доступен для загрузки в Личном кабинете[216] при наличии действующей лицензии на продукт.

2. Создайте каталог `/opt/rbta/aa/CDK-setup/`:

   ```
   sudo mkdir -p /opt/rbta/aa/CDK-setup/
   ```

3. Распакуйте содержимое архива с установщиком в каталог `/opt/rbta/aa/CDK-setup/`:

   ```
   sudo tar -xvzf <archive>.tar.gz -C /opt/rbta/aa/CDK-setup/
   ```

4. Перейдите в каталог с распакованными файлами:

   ```
   cd /opt/rbta/aa/CDK-setup/
   ```

5. Сделайте файл `install_cdk_bundle.sh` исполняемым:

   ```
   sudo chmod +x install_cdk_bundle.sh
   ```

[216] https://lk.astra.ru

6. Для установки всех доступных утилит выполните команду:

```
sudo ./install_cdk_bundle.sh
```

7. Чтобы установить отдельную утилиту, выполните команду:

```
sudo ./install_cdk_bundle.sh <utility_name>
```

Здесь <utility_name> – название утилиты, которую необходимо установить. Поддерживаемые значения:

- `ansible-navigator`;
- `ansible-builder`;
- `ansible-lint`;
- `molecule`;
- `pytest-ansible`.

Примечание

Утилита `ansible-test` входит в состав Ansible Core. При его наличии установка дополнительных пакетов не требуется.

8. Проведите дополнительные настройки Podman согласно *инструкции*.
9. Если у вас установлен Podman версии 4.3.1, добавьте `aa-creator-ee` в список локальных образов:

```
sudo podman load -i /opt/rbta/aa/CDK-setup/aa-creator-ee/aa-creator-ee.tar
```

Здесь `/opt/rbta/aa/CDK-setup/aa-creator-ee/aa-creator-ee.tar` – путь к архиву с образом `aa-creator-ee`.

17.1 Рабочее окружение

Рабочее окружение предназначено для разработки контента Ansible, используемого прежде всего в Astra Automation. Основой среды разработки является пакет Astra Automation CDK (Content Development Kit), который можно использовать любым из следующих способов:

- установкой пакета на рабочую станцию под управлением операционной системы Astra Linux;
- использованием служебного контейнера из образа `aa-cdk`, содержащего все необходимые утилиты из состава Astra Automation CDK.

17.1.1 Достоинства предлагаемого подхода

Основным инструментом разработки является служебный контейнер `aa-cdk`, называемый контейнером разработки (dev container[217]). В отличие от «классического» подхода к разработке, основанного на написании кода на рабочей станции в одном из текстовых редакторов и тестировании с помощью утилиты `ansible-playbook`, использование специализированного контейнера дает ряд преимуществ:

- контейнер можно использовать в разных операционных системах, прежде всего в семействах Linux, Microsoft Windows и macOS;
- нет необходимости самостоятельно устанавливать и настраивать программное обеспечение – Python, Ansible и инструменты разработки;
- разрабатываемый код ориентирован на его дальнейшее применение в Automation Controller с использованием среды исполнения (EE, Execution Environment);

[217] https://containers.dev/

- инструменты разработки и тестирования интегрированы с Visual Studio Code, что повышает удобство разработки.

Расширение Ansible предоставляет помощь при разработке кода в редакторе, включая:

- разметку кода с выделением компонентов;
- показ документации при наведении на используемый модуль;
- обнаружение ошибок в синтаксисе кода;
- обнаружение нарушений принятого стиля написания кода.

17.1.2 Требования к рабочему окружению

Среда разработки, основанная на использовании контейнера `aa-cdk`, должна удовлетворять следующим требованиям:

- рабочим местом является рабочая станция с установленной операционной системой Linux, Windows или macOS;
- на рабочей станции установлена система управления контейнерами в виде Podman (предпочтительно) или Docker;
- на рабочей станции установлена интегрированная среда разработки (IDE, Integrated Development Environment) Visual Studio Code;
- рабочая станция имеет доступ к приватному реестру коллекций Ansible и образов служебных контейнеров;
- рабочая станция имеет доступ в интернет для загрузки необходимых дополнительных коллекций Ansible, модулей Python и debian-пакетов операционной системы Astra Linux и платформы Astra Automation, в частности, доступ к реестру `dl.astralinux.ru` или его зеркалам.

17.1.3 Состав образа

Образ `aa-cdk` включает следующие утилиты и приложения для разработки контента Ansible:

- Ansible Creator,
- Ansible Builder,
- Ansible Lint,
- Ansible Molecule,
- Ansible Navigator.
- Ansible Sign,
- Tox Ansible,
- Pytest-ansible,
- Podman,
- Pre-commit,
- Git.

Для тестирования разрабатываемого контента с помощью Molecule образ содержит набор драйверов для различных сред тестирования:

- azure,
- containers,
- default,
- docker,
- ec2,

- gce,
- molecule_brest,
- molecule_yandexcloud,
- libvirt,
- openstack,
- podman,
- vagrant.

Для получения последней информации обращайтесь к странице образа в реестре[218].

17.1.4 Установка и настройка

Образ aa-cdk можно использовать в любой современной операционной системе, поддерживающей системы контейнеризации. На рабочей станции должна быть установлена интегрированная система разработки кода Visual Studio Code. Установка другого обязательного компонента – системы управления контейнерами, в частности Podman – зависит от операционной системы.

Astra Linux

В Linux для установки Podman используйте систему управления пакетами, принятую для соответствующего семейства операционных систем. Например, в Astra Linux Special Edition выполните следующую команду:

```
sudo apt install podman
```

Microsoft Windows

Настройка рабочего окружения в операционной системе Microsoft Windows для использования aa-cdk характерна следующими особенностями:

- необходима среда виртуализации WSL 2 (Windows Subsystem for Linux);
- система контейнеризации Podman, устанавливаемая в WSL, должна быть настроена на систему управления ресурсами cgroups2.

Один из вариантов установки состоит из следующих шагов:

1. Установите среду WSL 2 (Windows Subsystem for Linux) одним из следующих способов:
   - В Windows 10 версии 2004 и выше (сборка 19041 и выше) или Windows 11 воспользуйтесь следующей командой в терминале Windows:

     ```
     wsl --install --no-distribution
     ```

     После установки выполните перезагрузку компьютера.
   - В более ранних версиях установите ее согласно инструкции Microsoft[219].
2. Загрузите пакет Podman CLI for Windows[220].
3. Запустите установку Podman.
4. На запрос о выборе системы виртуализации (WSLv2 or Hyper-V) выберите пункт **Windows Linux Subsystem (WSLv2)**.
5. При получении запроса на разрешение вносить изменения на устройстве предоставьте эту возможность системе.

После установки настройте Podman:

[218] https://hub.astra-automation.ru/ui/containers/
[219] https://learn.microsoft.com/ru-ru/windows/wsl/install-manual
[220] https://podman.io/

1. Создайте виртуальную машину Podman:

```
podman machine init
```

2. Установите режим `rootful` для запуска контейнеров:

```
podman machine set --rootful
```

3. Запустите сервис Podman:

```
podman machine start
```

4. Настройте и активируйте систему управления `cgroups2`. Для этого создайте файл `.wslconfig` в домашнем каталоге пользователя со следующим содержимым:

```
[wsl2]
kernelCommandLine = cgroup_no_v1="all"
```

5. Остановите текущий процесс WSL и запустите заново в терминале Windows:

```
wsl --shutdown
```

```
wsl
```

6. После успешного входа в оболочку WSL убедитесь, что Podman использует `cgroups2`.

```
podman info | findstr cgroups
```

Вывод на экран имеет следующий вид:

```
cgroupControllers:
cgroupManager: cgroupfs
cgroupVersion: v2
```

macOS

Предупреждение

На текущий момент для моделей M, использующих процессоры с архитектурой ARM64, есть ограничения, связанные с невозможностью запускать контейнер в контейнере:

- Ansible Navigator необходимо использовать без Execution Environment;
- Сборка образа Execution Environment не поддерживается.

Для установки и запуска Podman на macOS можно воспользоваться следующими командами с использованием Homebrew:

1. Установите Podman:

```
brew install podman
```

2. Создайте виртуальную машину Podman:

```
podman machine init
```

3. Запустите сервис Podman:

```
podman machine start
```

17.2 Последовательность разработки

Разработка кода Ansible состоит из следующих этапов:

1. *Настройка среды разработки*.
2. *Создание структуры проекта*.
3. *Разработка кода*.
4. *Отладка*.
5. *Разработка тестов*.
6. *Подготовка к использованию в производственной среде*.

Последовательность этапов может отличаться в зависимости от принятого подхода к разработке. На каждом из этих этапов потребуются инструменты, содержащиеся в образе `aa-cdk`.

17.2.1 Настройка среды разработки

Проект по разработке контента Ansible необходимо проводить в Visual Studio Code (VS Code). Проект требует предварительной настройки, которую можно выполнить различными способами, один из которых приведен далее:

1. Создайте рабочий каталог и откройте его в VS Code.
2. В рабочем каталоге создайте каталог `.devcontainer/` и в нем файл `devcontainer.json` со следующим содержимым:

```
{
  "name": "Astra Automation Content Development Environment",
  "image": "hub.astra-automation.ru/aa-1.2/aa-cdk:latest",
  "containerUser": "root",
  "privileged": true,
  "mounts": [
    "source=${localWorkspaceFolder},target=/tmp/podman/oscap,type=bind,
↪consistency=cached"
  ],
  "customizations": {
    "vscode": {
      "extensions": ["redhat.ansible", "ms-azuretools.vscode-docker"]
    }
  }
}
```

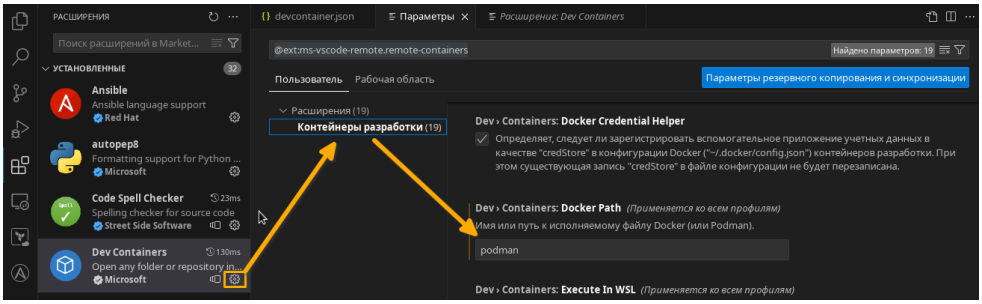
Примечание

1. В поле `image` необходимо указать путь к требуемому образу `aa-cdk` в любом реестре контейнеров, где он доступен.
2. Монтирование рабочего каталога в `/tmp/podman/oscap/` необходимо для сканирования системой OpenScap[221] содержимого проекта на соответствие требованиям безопасности при создании контейнера в контейнере.

3. Если расширение **Dev Containers** не установлено, VS Code предложит установить его. Установите его. Если вы не заметили это уведомление, и расширение не установлено, найдите его в магазине расширений и выполните установку.

   Для переключения на использование Podman в качестве системы контейнеризации перейдите в настройки расширения и измените значение параметра **Docker Path** на **podman**:

[221] https://static.open-scap.org/openscap-1.3/oscap_user_manual.html



4. Откройте рабочий каталог в *контейнере*. Один из способов это сделать – нажать комбинацию клавиш Ctrl+Shift+P (Cmd+Shift+P в macOS) и выбрать **Dev Containers: Reopen in Container**.

   Если вы настраиваете первый такой проект на рабочей станции, в результате этого действия VS Code выполнит следующие операции:

   - загрузит образ контейнера `aa-cdk` из указанного реестра;
   - запустит контейнер `aa-cdk` с подключением (монтированием) к нему каталога проекта (каталог проекта будет внутри `/workspaces/`);
   - установит расширения:
     - Ansible;
     - YAML;
     - Python.

   VS Code подключится к контейнеру и все дальнейшие действия будут происходить в его среде.

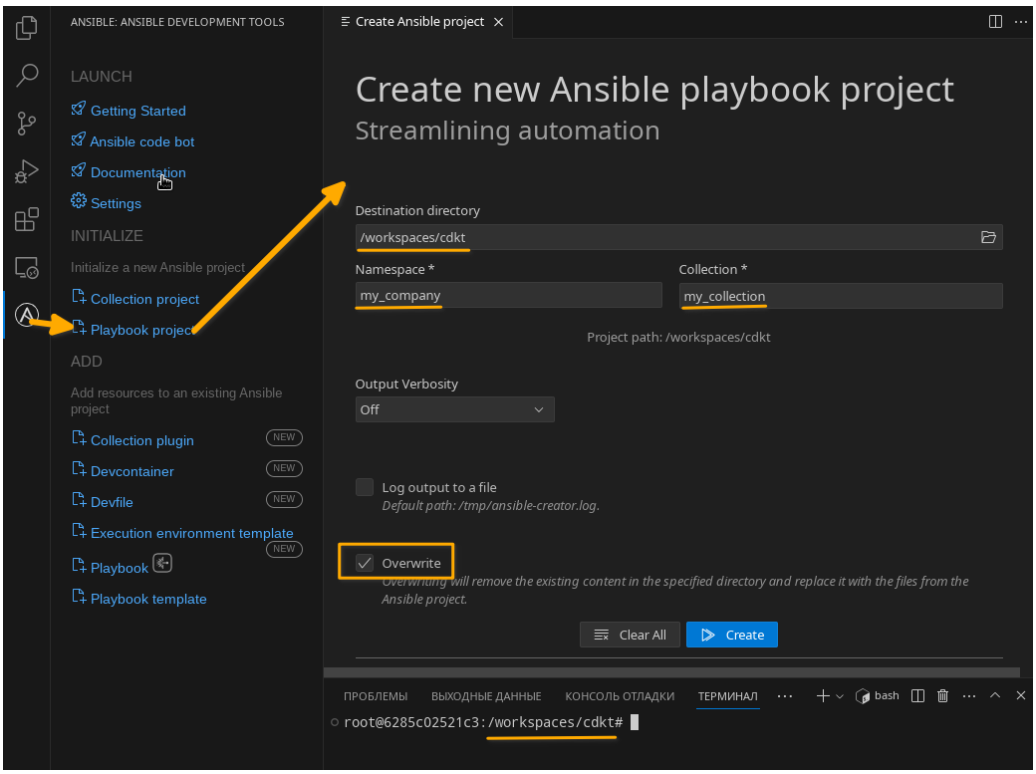
   Актуальная инструкция по подключению контейнера приводится также во встроенном документе реестра[222].

17.2.2 Создание структуры проекта

Расширение **Ansible** в VS Code позволяет подготовить типовую структуру каталога для разработки коллекции или сценариев автоматизации. Для этого оно использует утилиту `ansible-creator` из состава Astra Automation CDK.

Для создания структуры откройте страницу настройки **Ansible Development Tools** и следуйте следующим рекомендациям в зависимости от типа контента, который вы будете создавать.

[222] https://hub.astra-automation.ru/ui/containers/aa-1.2/aa-cdk/



Создание набора сценариев

Для настройки проекта на разработку отдельного набора сценариев выполните следующие действия:

1. В настройках инструментов выберите **Playbook project**.
2. Задайте необходимые параметры:
   - **Destination directory**: путь внутри контейнера к каталогу проекта, который должен иметь вид `/workspaces/<project_dir>`. Благодаря монтированию каталога, содержимое проекта сохранится также на рабочей станции.
   - **Namespace**: название пространства имен, например, «my_company»;
   - **Collection**: название коллекции, например, «my_collection».
3. Установите опцию `Overwrite`. Это необходимо для обновления файла `.devcontainer/devcontainer.json`.
4. Нажмите кнопку **Create**. Расширение создает необходимую структуру проекта для разработки набора сценариев.

Создание коллекции

Для настройки проекта на разработку коллекции, выполните следующие действия:

1. В настройках инструментов выберите **Collection project**.
2. Задайте необходимые параметры:
   - **Namespace**: название пространства имен, например, «my_company»;
   - **Collection**: название коллекции, например, «my_collection»;
   - **Init path**: путь внутри контейнера, где должна быть создана коллекции (рекомендуется `/workspaces/<project_dir>`).
3. Установите опцию `Overwrite`. Это необходимо для обновления файла `.devcontainer/devcontainer.json`.
4. Нажмите кнопку **Create**. В рабочем каталоге создается структура коллекции.

Дополнительные действия

Внутри каталога `.devcontainer/` появляются также настройки для запуска контейнера в среде Docker и Podman:

```
.devcontainer
├── devcontainer.json
├── docker
│   └── devcontainer.json
└── podman
    └── devcontainer.json
```

Система изменяет настройки контейнера. Например, файл `podman/devcontainer.json` приобретает следующий вид:

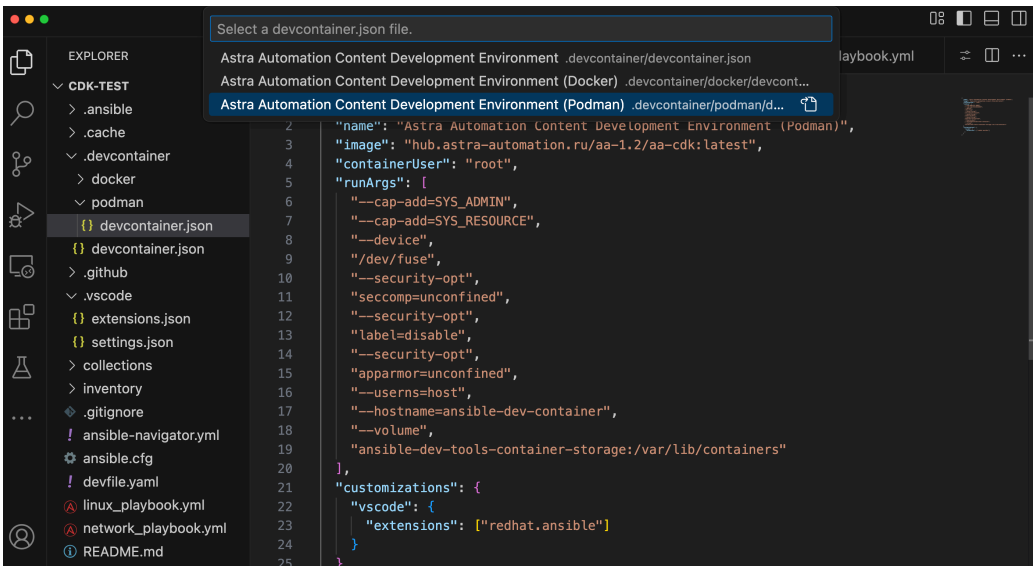
```
{
   "name": "Astra Automation Content Development Environment (Podman)",
   "image": "hub.astra-automation.ru/aa-1.2/aa-cdk:latest",
   "containerUser": "root",
   "runArgs": [
      "--cap-add=SYS_ADMIN",
      "--cap-add=SYS_RESOURCE",
      "--device",
      "/dev/fuse",
      "--security-opt",
      "seccomp=unconfined",
      "--security-opt",
      "label=disable",
      "--security-opt",
      "apparmor=unconfined",
      "--userns=host",
      "--hostname=ansible-dev-container",
      "--volume",
      "ansible-dev-tools-container-storage:/var/lib/containers"
   ],
   "customizations": {
      "vscode": {
         "extensions": ["redhat.ansible", "ms-azuretools.vscode-docker"]
      }
   }
}
```

Если разработка ведется в Microsoft Windows, необходимо добавить следующую строку в этом файле:

```
"privileged": true,
```

VS Code обнаружит обновление настроек и предложит пересоздать контейнер, с чем следует согласиться. Если соответствующее уведомление пропущено, нужно сделать это вручную, открыв палитру команд сочетанием клавиш Ctrl+Shift+P и выбрав пункт **Rebuild Container**.

В дальнейшем, при запуске контейнера рекомендуется использовать настройки Podman:



17.2.3 Разработка кода

Разработайте код вашего проекта для решения поставленной бизнес-задачи.

Зависимости

Разрабатываемый код Ansible может зависеть от сторонних компонентов – коллекций Ansible и модулей Python. Установите эти компоненты в образе `aa-cdk` следующим образом:

1. Создайте файлы, описывающие зависимости вашего проекта:
   - `requirements.yml` – перечень необходимых коллекций Ansible;
   - `requirements.txt` – перечень необходимых модулей Python.
2. Для установки требуемых компонентов нажмите комбинацию клавиш Ctrl+Shift+P и выберите **Create New Terminal (With Profile)**. VS Code откроет встроенный терминал. Убедитесь, что вы находитесь в командной строке контейнера, о чем свидетельствует приглашение вида:

```
root@ansible-dev-container:
```

3. Установите требуемые компоненты стандартными командами:

```
ansible-galaxy install -r requirements.yml
```

```
pip install -r requirements.txt
```

Примечание

1. Повторяйте приведенные шаги по мере изменения списка зависимостей.
2. Все последующие действия в командной строке, которые приводятся в этой инструкции, выполняются в оболочке контейнера `aa-cdk`, аналогично тому, как описано ранее. Описание процесса запуска оболочки в среде `aa-cdk` в последующих примерах будет опущена.

Проверка

Во время разработки система будет автоматически проверять код в редакторе на наличие ошибок. Для этого расширение Ansible использует утилиту `ansible-lint` из состава Astra Automation CDK. Вы можете также вызывать эту утилиту вручную из командной строки. Подробнее о ее использовании см. *Ansible Lint*.

Кроме того, в VS Code вы можете получать справку по модулям Ansible путем наведения курсора на соответствующие строки. Расширение также обеспечивает автоматическое дополнение названия модулей.

17.2.4 Отладка

Для проверки и отладки кода Ansible в контейнере `aa-cdk` предусмотрены несколько способов.

Использование ansible-navigator

Использование `ansible-navigator` для запуска и отладки набора сценариев является рекомендованным способом. Эта утилита позволяет запускать код Ansible в двух режимах:

- В том окружении, где она непосредственно установлена (рекомендуется на этапе отладки). В данном случае – это среда контейнера `aa-cdk`, содержащая надежные проверенные утилиты.
- В среде исполнения (рекомендуется в производственном процессе).

При создании *структуры проекта* расширение Ansible создает файл `.vscode/settings.json`:

```
{
   "ansible.executionEnvironment.enabled": true,
   "ansible.executionEnvironment.image": "hub.astra-automation.ru/aa-1.2/aa-full-
↪ee:latest",
}
```

В поле `ansible.executionEnvironment.enabled` необходимо установить значение `false`. Это означает, что `ansible-navigator` будет запускать файлы «локально» в контейнере `aa-cdk`, как требуется на этапе отладки.

Для запуска набора сценариев с помощью `ansible-navigator` в интерфейсе VS Code достаточно кликнуть правой клавишей мыши в тексте набора сценариев или на названии файла и выбрать пункт **Run Ansible Playbook via... > Run playbook via ansible-navigator**.

Ansible Navigator имеет удобные средства отладки для каждой задачи в наборе сценариев. После (или во время) выполнения кода можно перемещаться по списку задач и наблюдать весь контекст их запуска, включая значения переменных и другие метаданные. Подробности смотрите в *описании Ansible Navigator*.

Использование ansible-playbook

В образе `aa-cdk` сохранена возможность непосредственного запуска набора сценариев с помощью утилиты `ansible-playbook`. Вы можете запускать ее следующими способами:

- из командной строки служебного контейнера:

  ```
  ansible-playbook -i <inventory> <playbook.yml>
  ```

- из редактора VS Code, кликнув правой клавишей мыши на файле сценария и выбрав **Run Ansible Playbook via... > Run playbook via ansible-playbook**.

17.2.5 Разработка тестов

Для тестирования коллекций в контейнере установлено средство тестирования *Molecule*.

При создании структуры проекта для разработки коллекции в рабочем каталоге уже присутствует шаблон конфигурации для Molecule, тестовый набор сценариев `molecule/utils/playbooks/converge.yml` и другие файлы, необходимые для тестирования. Этот шаблон предусматривает проверки в локальной среде тестирования, однако Molecule в составе Astra Automation CDK поставляется с набором драйверов для различных систем виртуализации, в том числе:

- `molecule-yandex`;

- `molecule-brest`;
- `molecule-libvirt`.

Запускать тесты Molecule рекомендуется в среде контейнера `aa-cdk`, находясь в корне рабочего каталога, с помощью следующей команды:

```
molecule test
```

17.2.6 Подготовка к использованию в производственной среде

До этого момента запуск разрабатываемого кода происходил в контейнере `aa-cdk`. При применении проекта в Automation Controller для запуска необходимо использовать *среду исполнения*, в которой предварительно установлены все компоненты инфраструктурного кода, необходимые для проекта.

Использование среды исполнения дает ряд преимуществ:

- Переносимость кода. Разработанный проект Ansible будет одинаково функционировать на разных рабочих станциях и разных операционных системах, так как он выполняется внутри контейнера, в котором присутствуют нужные компоненты.
- При использовании в закрытом контуре нет необходимости загружать требуемые компоненты из внешних источников, так как все они установлены в контейнер.

Сборка среды исполнения

Для сборки требуемой среды исполнения образ `aa-cdk` содержит утилиту *ansible-builder* из состава Astra Automation CDK.

Используйте ее для создания нового образа среды исполнения:

1. Создайте в корне каталога проекта файл `execution-environment.yml` со следующим содержимым:

   ```
   ---
   version: 3

   images:
     base_image:
       name: hub.astra-automation.ru/aa-1.2/aa-minimal-ee:1.0.4

   dependencies:
     python: requirements.txt
     galaxy: requirements.yml

   additional_build_files:
     - src: ansible.cfg
       dest: configs

   additional_build_steps:
      prepend_galaxy:
        - COPY _build/configs/ansible.cfg /etc/ansible/ansible.cfg
        - ARG ANSIBLE_GALAXY_SERVER_VALIDATED_TOKEN
   ```

   Здесь:

   - `aa-minimal-ee` – образ, рекомендуемый для сборки собственных образов исполнения;
   - `dependencies` – список файлов с зависимостями, указывающими на компоненты, которые требуется установить в среду исполнения.

2. Для получения коллекций из приватного реестра коллекций передайте значение токена в переменную среды ANSIBLE_GALAXY_SERVER_VALIDATED_TOKEN. Используйте сохраненный ранее токен доступа к реестру коллекций или *создайте новый*.

```
export ANSIBLE_GALAXY_SERVER_VALIDATED_TOKEN=<your_token>
```

3. Создайте файл `ansible.cfg`, в котором укажите адреса репозиториев, содержащих требуемые коллекции Ansible. Подробный пример приведен в секции *Добавление коллекций из Private Automation Hub*.
4. Для сборки среды выполните следующую команду в среде `aa-cdk`:

```
ansible-builder build -t my-ee:0.0.1 --build-arg ANSIBLE_GALAXY_SERVER_VALIDATED_
↪TOKEN
```

Проверка работы в среде исполнения

Для проверки работы проекта с помощью `ansible-navigator` в созданной среде исполнения необходимо выполнить следующие действия:

1. Приведите файл `.vscode/settings.json` к следующему виду:

```
{
   "ansible.executionEnvironment.enabled": true,
   "ansible.executionEnvironment.image": "localhost/my-ee:0.0.1",
}
```

2. Выполните набор сценариев с помощью `ansible-navigator`.

   `ansible-navigator` создает новый контейнер из образа, указанного на предыдущем шаге, монтирует в него рабочий каталог и исполняет набор сценариев.

Публикация среды исполнения

Для дальнейшего использования получившийся образ необходимо загрузить в приватный реестр, как описано в инструкции *Загрузка с рабочей станции*. Кратко необходимые действия выглядят следующим образом:

1. Авторизуйтесь в приватном реестре:

```
podman login -u=<username> -p=<password> <hub>
```

2. Задайте образу правильное название, которое содержит доменное имя приватного реестра (<hub>):

```
podman tag my-ee:0.0.1 <hub>/my-ee:0.0.1
```

3. Загрузите образ в приватный реестр:

```
podman push <hub>/my-ee:0.0.1
```

Публикация коллекции

После разработки и тестирования коллекции Ansible необходимо загрузить ее в приватный реестр коллекций:

1. Упакуйте коллекцию в формате tarball, выполнив следующую команду в среде контейнера `aa-cdk`:

```
ansible-galaxy collection build
```

   Эта команда создает файл с расширением `.tar.gz` – архив коллекции.

2. Настройте утилиту *ansible-galaxy* из состава `aa-cdk` на работу с приватным реестром, как описано в *настройке проекта*.
3. Загрузите упакованную коллекцию в реестр коллекций командой:

```
ansible-galaxy collection publish --server https://<hub>/<repository> <file-name>.
↪tar.gz [--token <token>]
```

Здесь:

- <hub> – доменное имя приватного реестра коллекций.
- <repository> – полный путь к репозиторию, в котором необходимо разместить коллекцию. Репозиторий должен быть создан заранее с помощью графического пользовательского интерфейса. Если необходимо согласование, то подставьте `api/galaxy`.
- <file-name> – название файла, содержащего упакованную коллекцию.
- <token> – токен доступа к приватному реестру.

Токен доступа к приватному реестру можно предоставить различными способами:

- в командной строке (не рекомендуется, поскольку это наименее защищенный способ);
- поместив его в `ansible.cfg`.

Публикация проекта

После разработки набора сценариев его следует сделать доступным в системе контроля исходного кода (SCM, Source Code Management), в качестве которой обычно используется Git. Набор сценариев должен быть помещен в репозиторий SCM в составе отдельного проекта или добавлен в существующий проект вместе с сопутствующими каталогами и файлами. В таком виде он будет доступен для дальнейшего использования в Automation Controller.

17.2.7 Заключение

В данном руководстве рассмотрен рекомендуемый процесс разработки кода Ansible с использованием контейнера для разработки `aa-cdk` и отдельных инструментов из Astra Automation CDK. Показано, как подготовить среду исполнения. Приведен процесс публикации артефактов в соответствующих реестрах для дальнейшего их применения в Astra Automation.

17.3 Средства разработки

В состав Astra Automation входят следующие средства для разработки контента Ansible:

- *Ansible Navigator*;
- *Ansible Builder*;
- *Ansible Creator*;
- *Ansible Sign*.

17.3.1 Ansible Navigator

Ansible Navigator предоставляет удобный единый интерфейс для использования возможностей многих утилит Ansible в командной строке. Ansible Navigator может вызывать эти утилиты из среды исполнения (основной рекомендуемый вариант использования) или непосредственно из рабочей станции.

Установка

Для установки Ansible Navigator выполните следующие действия:

1. Подключите репозиторий Astra Automation.

Инструкция по подключению репозитория

1. В каталоге `/etc/apt/sources.list.d/` создайте файл `astra-automation.list` со ссылкой на репозиторий Astra Automation:

   ```
   deb https://dl.astralinux.ru/aa/aa-debs-for-alse-1.8 <version> main
   ```

   Вместо <version> необходимо подставить версию устанавливаемой платформы, например, `2.0`.

   Доступные версии продукта опубликованы в таблице *История обновлений*.

2. Обновите список доступных пакетов:

   ```
   sudo apt update
   ```

2. Выполните команду:

   ```
   sudo apt-get install ansible-navigator --yes
   ```

Вместе с Ansible Navigator также устанавливаются:

- *Ansible Builder*;
- *Ansible Lint*;
- Ansible Runner;
- Podman.

Установка при отсутствии доступа к интернету описана в документе *Средства разработки*.

Команды и утилиты

Основу Ansible Navigator составляет утилита `ansible-navigator`, вызов которой имеет следующий вид:

```
ansible-navigator <command> [<arguments>]
```

Здесь:

- <command> – команда;
- <arguments> – аргументы.

Команды, используемые для вызова других утилит Ansible:

Команда	Вызываемая утилита	Пример
`builder`	*ansible-builder*	`ansible-navigator builder␣ ↪build --tag ee:0.0.1`
`config`	ansible-config[223]	`ansible-navigator config␣ ↪init`
`doc`	ansible-doc[224]	`ansible-navigator doc␣ ↪ansible.builtin.apt`
`inventory`	ansible-inventory[225]	`ansible-navigator␣ ↪inventory -i ./inventory. ↪ini`
`lint`	ansible-lint[226]	`ansible-navigator lint ./ ↪playbooks/*.yml`
`run`	ansible-playbook[227]	`ansible-navigator run ./ ↪playbook.yml -i ./ ↪inventory.yml`

Собственные команды Ansible Navigator:

- `collections` – просмотр сведений об имеющихся коллекциях Ansible.
- `exec` – запуск команды Ansible в *среде исполнения*.
- `images` – управление образами среды исполнения.
- `replay` – просмотр артефактов, собранных во время выполнения сценариев.
- `settings` – просмотр действующих значений настроек Ansible Navigator.
- `welcome` – интерактивный режим работы.

Подробное описание команд и возможных аргументов см. в *справочных данных*.

Режимы работы

Ansible Navigator может работать в интерактивном (по умолчанию) и неинтерактивном режимах.

Для запуска Ansible Navigator в неинтерактивном режиме добавьте аргумент `--mode` со значением `stdout`, например:

```
ansible-navigator images list --mode stdout
```

Интерактивный режим работы

При запуске в интерактивном режиме Ansible Navigator предоставляет псевдографический интерфейс следующего вида:

[223] https://docs.ansible.com/ansible/latest/cli/ansible-config.html
[224] https://docs.ansible.com/ansible/latest/cli/ansible-doc.html
[225] https://docs.ansible.com/ansible/latest/cli/ansible-inventory.html
[226] https://ansible.readthedocs.io/projects/lint/
[227] https://docs.ansible.com/ansible/latest/cli/ansible-playbook.html

```
 0|Welcome
 1|──────────────────────────────────────────────────────────────────────────────
──────────────
 2|
 3|Some things you can try from here:
 4|- :collections                                    Explore available collections
 5|- :config                                         Explore the current ansible␣
↪configuration
 6|- :doc <plugin>                                   Review documentation for a module␣
↪or plugin
 7|- :help                                           Show the main help page
 8|- :images                                         Explore execution environment images
 9|- :inventory -i <inventory>                       Explore an inventory
10|- :log                                            Review the application log
11|- :lint <file or directory>                       Lint Ansible/YAML files␣
↪(experimental)
12|- :open                                           Open current page in the editor
13|- :replay                                         Explore a previous run using a␣
↪playbook artifact
14|- :run <playbook> -i <inventory>                  Run a playbook in interactive mode
15|- :settings                                       Review the current ansible-
↪navigator settings
16|- :quit                                           Quit the application
17|
18|happy automating,
19|
20|-winston
^b/PgUp page up        ^f/PgDn page down        ↑↓ scroll        esc back        :help␣
↪help
```

Этот интерфейс состоит из следующих элементов, сверху вниз:

1. Заголовки столбцов. Первый столбец не имеет названия и используется для вывода номера пункта списка.
2. Список записей.
3. Строка с подсказками.

Интерактивный режим обладает следующими особенностями:

- Для прокрутки списка используются клавиши ↑, ↓, PgUp и PgDn.
- Для завершения работы с Ansible Navigator или перехода на уровень выше используется клавиша Esc.
- Для ввода команд и перехода к пунктам списка используется следующая последовательность действий:
  1. Ввести символ `:` и сразу же после него название команды или номер строки, например, `:help` или `:19`.
  2. Нажать Enter.

  Совет

  Для перехода к строкам с номерами от 0 до 9 можно использовать быстрый ввод – достаточно нажать соответствующую клавишу цифрового ряда клавиатуры.

- При работе с некоторыми разделами можно использовать клавиши `-` и `+` для перехода к предыдущему и следующему пунктам списка соответственно.

Интерактивный режим может быть полезен для получения детальных сведений о действующих настройках Ansible Navigator, образах среды исполнения, коллекциях и так далее.

Неинтерактивный режим работы

При запуске в неинтерактивном режиме Ansible Navigator выполняет команды без взаимодействия с пользователем. Вывод выполняемых команд направляется в стандартный поток вывода `stdout`.

Неинтерактивный режим может быть полезен при использовании Ansible Navigator в сценариях командной строки, когда взаимодействие с пользователем не требуется.

Настройки

Значения настроек Ansible Navigator могут быть заданы следующими способами:

- аргументы командной строки;
- переменные окружения;
- конфигурационный файл.

При запуске Ansible Navigator выполняет поиск своего конфигурационного файла в следующем порядке:

1. Путь, указанный в переменной окружения `ANSIBLE_NAVIGATOR_CONFIG`.
2. Файл `ansible-navigator.<ext>` в каталоге проекта.
3. Файл `.ansible-navigator.<ext>` в домашнем каталоге.

Здесь <ext> – расширение имени файла, `.yml`, `.yaml` или `.json`.

Поиск прекращается при первом совпадении. Например, если в каталоге проекта существует файл `ansible-navigator.yml`, настройки из файла `~/.ansible-navigator.yml`, расположенного в домашнем каталоге, загружены **не будут**.

Особенности хранения настроек в конфигурационных файлах:

- Доступны форматы YAML и JSON.
- При использовании формата JSON название файла должно иметь расширение `.json`.
- При использовании формата YAML название файла должно иметь расширение `.yml` или `.yaml`.
- В каталоге проекта и домашнем каталоге должен быть только один конфигурационный файл Ansible Navigator. В противном случае Ansible Navigator выводит сообщение об ошибке.

Пример файла с настройками

Список 1: `ansible-navigator.yml`

```yaml
---
ansible-navigator:
  # Основные настройки Ansible
  ansible:
    cmdline: -vv # Более подробный вывод чем обычно
    config:
      path: ./ansible.cfg # Путь к конфигурационному файлу Ansible
    inventory:
      entries: # Пути к файлам с инвентарными списками
        - ./inventory.yml

  # Настройки Ansible Builder
  ansible-builder:
    workdir: ./execution-environment/ # Рабочий каталог

  # Настройки Ansible Runner
  ansible-runner:
    artifact-dir: ./artifacts/ # Каталог для сохранения артефактов
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
    timeout: 600 # Ограничение по времени - 10 минут
    job-events: true # Сохранение событий в журнал

  # Настройки цветового оформления
  color:
    enable: true # Включено
    osc4: true # Использовать формат OSC4

  # Настройки текстового редактора
  editor:
    console: true # Запустить в TUI
    command: emacs +{line_number} {filename} # GNU Emacs

  # Настройки среды исполнения
  execution-environment:
    container-engine: podman # Инструмент контейнеризации - Podman
    enabled: true # Использование сред исполнения разрешено
    environment-variables:
      pass: # Список переменных, импортируемых из текущего окружения в контейнер
        - ANSIBLE_GALAXY_SERVER_VALIDATED_TOKEN
      set: # Значения переменных, которые должны быть переданы в контейнер
        USER: aa-service-account
        HOST: hub.example.com
    image: private-hub.example.com/aa-full-ee:latest # Образ EE
    pull:
      policy: missing # Загрузить образ только при его отсутствии
    volume-mounts: # Монтирование каталогов в файловую систему контейнера
      - src: ./playbooks/
        dest: /playbooks

  # Настройки журналирования
  logging:
    file: ./logs/ansible-navigator.log # Путь к файлу журнала
    level: error # Игнорировать сообщения с уровнем ниже Error

  # Порядок обработки настроек самого Ansible Navigator
  settings:
    effective: true # Показ фактических значений
    sources: true # Отображать источники значений

  # Часовой пояс
  time-zone: Europe/Moscow
```

Подробное описание настроек см. в *справочных данных*.

Примеры

Изучите использование Ansible Navigator в неинтерактивном и интерактивном режимах на примерах.

Использование Ansible Navigator в неинтерактивном режиме

Для работы с Ansible Navigator в неинтерактивном режиме используйте аргумент `--mode` со значением `stdout`.

- Получение справочной информации об использовании команды `ansible-config` в неинтерактивном режиме:

```
ansible-navigator config --hc
```

- Получение сведений о коллекциях, доступных в окружении используемой ОС:

```
ansible-navigator collections --mode stdout
```

- Получение сведений о коллекциях, доступных в образе `aa-full-ee:latest`:

```
ansible-navigator collections \
   --eei hub.astra-automation.ru/aa-1.2/aa-full-ee:latest \
   --mode stdout
```

- Получение сведений о загруженных образах среды исполнения:

```
ansible-navigator images --mode stdout
```

- Загрузка самой новой версии образа `aa-cdk` из реестра:

```
ansible-navigator images \
   --mode stdout \
   --eei hub.astra-automation.ru/aa-1.2/aa-cdk:latest
```

- Загрузка образа `aa-cdk:0.1.2`:

```
ansible-navigator images \
   --mode stdout \
   --eei hub.astra-automation.ru/aa-1.2/aa-cdk:0.1.2
```

Использование Ansible Navigator в интерактивном режиме

Для использования Ansible Navigator в интерактивном режиме используйте аргумент `--mode` со значением `interactive`.

Получение сведений об образах среды исполнения

Для получения сведения о составе образов среды исполнения выполните следующие действия:

1. Выполните команду `ansible-navigator` с аргументами `images` и `--mode`:

```
ansible-navigator images --mode interactive
```

   В терминал выводится список образов, например:

```
  Image          Tag     Execution environment   Created      Size
0|aa-full-ee     latest  True                    2 weeks ago  1.09 GB
1|aa-full-ee     0.1.1   True                    4 weeks ago  998 MB
2|aa-minimal-ee  latest  True                    5 weeks ago  284 MB
3|aa-minimal-ee  1.0.3   True                    5 weeks ago  281 MB
4|aa-minimal-ee  1.0.2   True                    5 weeks ago  273 MB
```

2. Для просмотра сведений об образе `aa-full-ee:0.1.1` выполните следующие действия:

   1. Последовательно нажмите клавиши `:`, `1` и `Enter`.

      В терминал выводится меню следующего вида:

```
  Image: aa-full-ee:0.1.1          Description
0|Image information                Information collected from image inspection
1|General information              OS and python version information
2|Ansible version and collections  Information about ansible and ansible␣
↪collections
3|Python packages                  Information about python and python packages
4|Operating system packages        Information about operating system packages
5|Everything                       All image information
```

   2. Нажмите соответствующую клавишу для перехода в нужный раздел. Например, для выбора раздела **Python packages** нажмите 3.

Дождитесь загрузки сведений о пакетах Python. По окончании загрузки в терминал выводятся строки следующего вида:

```
   Name                           Version   Summary
 0|ansible-compat                 24.9.1    Ansible compatibility goodies
↪                                    ▒
 1|ansible-core                   2.15.10   Radically simple IT automation
↪                                    ▒
 2|ansible-lint                   24.9.2    Checks playbooks for practices and
↪behavior that could potentially be im▒
 3|ansible-runner                 2.4.0     "Consistent Ansible Python API and
↪CLI with container and process isolat▒
 4|ansible-sign                   0.1.1     Ansible content validation library
↪and CLI                                ▒
 5|attrs                          22.2.0    Classes Without Boilerplate
↪                                    ▒
 6|backports.ssl_match_hostname 3.7.0.1   The ssl.match_hostname() function
↪from Python 3.5                        ▒
 7|bcrypt                         3.2.2     Modern password hashing for your
↪software and your servers
 8|black                          24.10.0   The uncompromising code formatter.
 9|bracex                         2.5.post1 Bash style brace expander.
10|certifi                        2022.9.24 Python package for providing Mozilla
↪'s CA Bundle.
11|cffi                           1.15.1    Foreign Function Interface for Python
↪calling C code.
12|chardet                        5.1.0     Universal encoding detector for
↪Python 3
13|charset-normalizer             3.0.1     The Real First Universal Charset
↪Detector. Open, modern and actively mai
14|click                          8.1.3     Composable command line interface
↪toolkit
15|colorama                       0.4.6     Cross-platform colored terminal text.
16|cryptography                   42.0.8    cryptography is a package which
↪provides cryptographic recipes and primi
17|distlib                        0.3.6     Distribution utilities
18|dnspython                      2.3.0     DNS toolkit
19|docker                         7.1.0     A Python library for the Docker
↪Engine API.
20|dumb-init                      1.2.5     Simple wrapper script which proxies
↪signals to a child
21|filelock                       3.9.0     A platform independent file lock.
^b/PgUp page up       ^f/PgDn page down       ↑↓ scroll       esc back       [0-
↪9] goto       :help help
```

3. Для просмотра сведений о пакете `cffi` введите строку `:11` и нажмите Enter.

3. Для возврата в предыдущее меню нажмите Esc.

Получение сведений о коллекциях

Для получения сведения о коллекциях, входящих в образ среды исполнения `aa-full-ee:latest`, выполните следующие действия:

1. Выполните команду `ansible-navigator` с аргументами `collections` и `--eei`:

```
ansible-navigator collections \
   --eei hub.astra-automation.ru/aa-1.2/aa-full-ee:latest
```

В терминал выводится список коллекций, например:

```
   Name               Version Shadowed Type      Path
 0|ansible.builtin    2.15.10 False    contained /usr/lib/python3/dist-packages/
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
↪ansible                                    ▒
 1|ansible.posix       1.5.4   False    contained /usr/share/ansible/collections/
↪ansible_collections/ansible/posix          ▒
 2|ansible.utils       3.0.0   False    contained /usr/share/ansible/collections/
↪ansible_collections/ansible/utils          ▒
 3|astra.aa_controller 0.14.0  False    contained /usr/share/ansible/collections/
↪ansible_collections/astra/aa_controller    ▒
 4|astra.ald_pro       3.0.0   False    contained /usr/share/ansible/collections/
↪ansible_collections/astra/ald_pro          ▒
 5|astra.astralinux    0.5.0   False    contained /usr/share/ansible/collections/
↪ansible_collections/astra/astralinux       ▒
 6|astra.brest         4.0.3   False    contained /usr/share/ansible/collections/
↪ansible_collections/astra/brest            ▒
 7|astra.ceph          3.0.3   False    contained /usr/share/ansible/collections/
↪ansible_collections/astra/ceph             ▒
 8|astra.chrony        0.2.1   False    contained /usr/share/ansible/collections/
↪ansible_collections/astra/chrony           ▒
 9|astra.cups          1.3.4   False    contained /usr/share/ansible/collections/
↪ansible_collections/astra/cups             ▒
10|astra.dcimanager    0.2.1   False    contained /usr/share/ansible/collections/
↪ansible_collections/astra/dcimanager       ▒
11|astra.dhcp          2.0.2   False    contained /usr/share/ansible/collections/
↪ansible_collections/astra/dhcp             ▒
12|astra.docker        4.0.1   False    contained /usr/share/ansible/collections/
↪ansible_collections/astra/docker
13|astra.etcd          0.2.1   False    contained /usr/share/ansible/collections/
↪ansible_collections/astra/etcd
14|astra.freeipa       0.4.1   False    contained /usr/share/ansible/collections/
↪ansible_collections/astra/freeipa
15|astra.grafana       2.0.1   False    contained /usr/share/ansible/collections/
↪ansible_collections/astra/grafana
16|astra.haproxy       2.0.1   False    contained /usr/share/ansible/collections/
↪ansible_collections/astra/haproxy
17|astra.hardening     0.3.1   False    contained /usr/share/ansible/collections/
↪ansible_collections/astra/hardening
18|astra.iscsi         0.4.2   False    contained /usr/share/ansible/collections/
↪ansible_collections/astra/iscsi
19|astra.keepalived    0.3.0   False    contained /usr/share/ansible/collections/
↪ansible_collections/astra/keepalived
20|astra.keycloak      2.0.2   False    contained /usr/share/ansible/collections/
↪ansible_collections/astra/keycloak
21|astra.memcached     3.0.1   False    contained /usr/share/ansible/collections/
↪ansible_collections/astra/memcached
```

2. Для просмотра сведений о коллекции `astra.grafana` введите строку `:15` и нажмите `Enter`.

   В терминал выводится список модулей и ролей, входящих в коллекцию.

3. Для просмотра сведений о нужном модуле или роли введите `:` и номер нужной строки, после чего нажмите `Enter`.

4. Для возврата в предыдущее меню нажмите `Esc`.

17.3.2 Ansible Builder

Ansible Builder – это инструмент командной строки, предназначенный для создания образов, используемых в качестве среды исполнения (EE, execution environment). Непосредственное создание образов осуществляется с помощью *Podman* (рекомендуемый вариант) или *Docker* (например, когда необходима *настройка размера разделяемой памяти*, `--shm-size`).

Создаваемый образ необходимо описать с помощью метаданных в специальном *файле*

определения среды исполнения.

Установка

Для установки Ansible Builder выполните следующие действия:

1. Подключите репозиторий Astra Automation.

   #### Инструкция по подключению репозитория

   1. В каталоге `/etc/apt/sources.list.d/` создайте файл `astra-automation.list` со ссылкой на репозиторий Astra Automation:

      ```
      deb https://dl.astralinux.ru/aa/aa-debs-for-alse-1.8 <version> main
      ```

      Вместо <version> необходимо подставить версию устанавливаемой платформы, например, `2.0`.

      Доступные версии продукта опубликованы в таблице *История обновлений*.

   2. Обновите список доступных пакетов:

      ```
      sudo apt update
      ```

2. Установите пакет `ansible-builder`:

   ```
   sudo apt install ansible-builder --yes
   ```

3. Установите и настройте *Podman* (рекомендуется) или *Docker*.

Установка при отсутствии доступа к интернету описана в документе *Средства разработки*.

Файл определения среды исполнения

Файл определения среды исполнения (EE definition) содержит описание создаваемого образа среды исполнения. По умолчанию Ansible Builder использует для сборки образа описание из файла `execution-environment.yml` или `execution-environment.yaml`, хранящегося в текущем каталоге.

Файл определения среды исполнения может содержать следующие разделы:

- `version`;
- `build_arg_defaults`;
- `dependencies`;
- `images`;
- `additional_build_files`;
- `additional_build_steps`;
- `options`.

Пример заполнения файла определения среды исполнения

Список 2: `execution-environment.yml`

```
---
version: 3

build_arg_defaults:
  ANSIBLE_GALAXY_CLI_COLLECTION_OPTS: '--pre'

dependencies:
  ansible_core:
    package_pip: ansible-core==2.15.6
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  ansible_runner:
    package_pip: ansible-runner
  galaxy: requirements.yml
  python:
    - six
    - psutil
  system: bindep.txt
  exclude:
    python:
      - docker
    system:
      - python3-Cython

images:
  base_image:
    name: hub.astra-automation.ru/aa/aa-creator-ee:0.1.0

additional_build_files:
  - src: files/ansible.cfg
    dest: configs

additional_build_steps:
  prepend_base:
    - RUN echo This is a prepend base command!

  prepend_galaxy:
    - COPY _build/configs/ansible.cfg /etc/ansible/ansible.cfg

  prepend_final:
    - RUN whoami
    - RUN cat /etc/os-release

  append_final:
    - RUN echo This is a post-install command!
    - RUN ls -la /etc
```

Подробное описание файла определения среды исполнения см. в *справочнике*.

Фазы создания

Процесс создания образа с помощью Ansible Builder включает две фазы.

1. Подготовка контекста сборки при выполнении команды `create`:

   ```
   ansible-builder create
   ```

   В результате выполнения этой команды Ansible Builder формирует каталог `context/`, содержащий файл с командами сборки и вспомогательные материалы. Этот этап не выполняет сборку образа, а лишь подготавливает все необходимые артефакты. Созданная структура необходима для корректной работы инструмента контейнеризации на следующем этапе.

   Содержимое каталога `context/` зависит от содержимого файла определения среды исполнения.

   Список 3: Пример структуры файлов и каталогов контекста

   ```
   context/
   ├── _build/
   │   ├── requirements.yml
   │   └── scripts/
   │       ├── assemble
   │       ├── check_ansible
   ```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
│       ├── check_galaxy
│       ├── entrypoint
│       ├── install-from-bindep
│       ├── introspect.py
│       └── pip_install
└── Containerfile
```

Здесь:

- `Containerfile` – файл с инструкциями сборки образа с помощью Podman. Содержит последовательность директив, управляющих сборкой конечного образа.

  Примечание

  При использовании Docker файл называется `Dockerfile`.

- `_build/requirements.yml` – объединенный список зависимостей Ansible, составленный на основе входных данных.
- `_build/scripts/assemble` – сценарий сборки окружения. Выполняется на завершающем этапе сборки и объединяет установленные компоненты в рабочее окружение.
- `_build/scripts/check_ansible` – проверка корректности установки Ansible в образе.
- `_build/scripts/check_galaxy` – проверка доступности и настроек `ansible-galaxy` внутри образа.
- `_build/scripts/entrypoint` – точка входа контейнера. Этот сценарий выполняется при запуске контейнера с собранным образом.
- `_build/scripts/install-from-bindep` – установка системных зависимостей, определенных в файле `bindep.txt`. Используется для обеспечения необходимых библиотек на уровне ОС.
- `_build/scripts/introspect.py` – сценарий анализа среды. Выполняет диагностику и подготовку информации о текущем окружении.
- `_build/scripts/pip_install` – сценарий для установки модулей Python, указанных в списке зависимостей `requirements.txt` или других конфигурационных файлах.

Подробное описание аргументов команды `create` см. в *справочнике*.

2. Сборка образа при выполнении команды `build`:

```
ansible-builder build -t <name>:<tag>
```

Здесь <name> и <tag> соответственно название и тег создаваемого образа.

Внутри этой фазы Ansible Builder передает управление инструменту контейнеризации, который выполняет все инструкции, описанные в `Containerfile` (при использовании Docker – `Dockerfile`), и формирует итоговый образ.

Команда `build` может быть вызвана сразу, без предварительного выполнения `create` – в этом случае обе фазы выполняются последовательно: сначала создается контекст сборки, затем запускается сборка образа.

Список 4: Пример команды сборки образа

```
ansible-builder build -v 3 -t my-project-ee:1.0.0
```

Подробное описание аргументов команды `build` см. в *справочнике*.

Этапы сборки

Ansible Builder выполняет несколько этапов при запуске инструмента контейнеризации для создания образа.

1. **Base**: использует Docker или Podman для загрузки базового образа, который был определен в файле определения среды исполнения, затем устанавливает версию Python (если она определена и отличается от всех версий Python в базовом образе), `pip`, `ansible-runner` и `ansible-core` или `ansible`. Все три последующих этапа сборки используют результаты этапа Base.
2. **Galaxy**: устанавливает коллекции Ansible, определенные в списке зависимостей.
3. **Builder**: устанавливает пакеты Python и системные пакеты, определенные в списке зависимостей.
4. **Final**: производит окончательную сборку образа.

Зависимости

Все зависимости определяются в настройках блока `dependencies`. Они определяют пакеты, библиотеки и коллекции, которые необходимо добавить в образ. Для установки зависимостей используется соответствующий инструмент – системный менеджер пакетов, PIP или `ansible-galaxy`.

Настройка	Описание	Менеджер пакетов
`dependencies.ansible_core`	Версия Python-пакета `ansible-core` в формате **PEP 508**[228]	PIP
`dependencies.ansible_runner`	Версия Python-пакета `ansible-runner` в формате **PEP 508**[229].	PIP
`dependencies.galaxy`	Зависимости Ansible	`ansible-galaxy`
`dependencies.python`	Зависимости Python	PIP
`dependencies.system`	Системные пакеты	Системный менеджер пакетов

Важно

Образы EE, используемые в Astra Automation текущей версии, основаны на операционной системе Astra Linux Special Edition 1.8. При создании новых образов на базе этих EE необходимо обеспечить доступ к репозиториям DEB-пакетов Astra Linux Special Edition 1.8, если вы используете зависимости типа `system`. Это особенно важно помнить при развертывании платформы Astra Automation в закрытом контуре с использованием Astra Linux Special Edition 1.7.

Операции с файлами

Контекст сборки – множество файлов, которые могут быть использованы во время сборки образа.

Чтобы указать контекст сборки, в файле определения среды исполнения укажите необходимые значения в настройке `additional_build_files` в следующем формате:

```
additional_build_files:
  - src: <source>
    dest: <target>
```

Здесь:

[228] https://peps.python.org/pep-0508/
[229] https://peps.python.org/pep-0508/

- <source> – путь к файлу или каталогу в локальной файловой системе;
- <target> – путь назначения внутри контекста сборки, относительно корня каталога `context/`.

Все файлы, указанные в этой настройке, будут скопированы в соответствующее место в каталоге `context/_build` и будут доступны во время сборки. Для обращения к этим файлам в `additional_build_steps` или в командах `RUN` и `COPY` внутри `Containerfile` (при использовании Docker – `Dockerfile`) укажите относительный путь от корня контекста с указанием каталога `_build/`.

Например, чтобы скопировать файл `ansible.cfg` из локальной файловой системы в расположение `/etc/ansible/ansible.cfg` внутри образа, настройте файл среды исполнения следующим образом:

```
additional_build_files:
  - src: ansible.cfg
    dest: config/
additional_build_steps:
  prepend_galaxy:
    - COPY _build/config/ansible.cfg /etc/ansible/ansible.cfg
```

В данном случае файл будет размещен в каталоге `context/_build/config/ansible.cfg` и будет скопирован в файловую систему образа с помощью инструкции `COPY` перед выполнением этапа **Galaxy**.

Дополнительные шаги

Ansible Builder не позволяет управлять файлом с описанием образа напрямую, однако, необходимые шаги можно указать в одной из настроек блока `additional_build_steps`. Этот блок представляет собой структуру, в которой можно задать команды для выполнения на определенной стадии процесса.

Список настроек в порядке выполнения в соответствии с *этапами сборки*:

1. `prepend_base` – до выполнения всех инструкций базового контейнера;
2. `append_base` – после выполнения всех инструкций базового контейнера;
3. `prepend_galaxy` – до установки зависимостей Ansible;
4. `append_galaxy` – после установки зависимостей Ansible;
5. `prepend_builder` – до установки зависимостей Python;
6. `append_builder` – после установки зависимостей Python;
7. `prepend_final` – до финальной сборки;
8. `append_final` – после финальной сборки.

Например, с помощью `append_builder` можно установить дополнительные системные пакеты. В следующем примере после этапа установки требуемых пакетов Python добавляется команда для установки утилиты `htop` через `apt`:

```
additional_build_steps:
  append_builder:
    - RUN apt-get update && apt-get install -y htop
```

В следующем примере с помощью `prepend_final` до финального этапа сборки в образ копируется и выполняется пользовательская инструкция:

```
additional_build_steps:
   prepend_final:
      - COPY scripts/setup.sh /usr/local/bin/setup.sh
      - RUN chmod +x /usr/local/bin/setup.sh && /usr/local/bin/setup.sh
```

Сборка образа

Для запуска сборки образа в корневом каталоге проекта выполните команду:

```
ansible-builder build -t <image>:<tag>
```

Здесь:

- <image> – название создаваемого образа;
- <tag> – метка образа.

Полный перечень возможных команд и опций см. в *справочнике*.

Особенности использования в окружении без доступа к интернету

Установка требуемых компонентов в окружении без доступа к Интернету имеет следующие особенности:

- Для корректной работы PIP необходим доступ к индексу пакетов Python. По умолчанию PIP использует индекс PyPi[230], доступный по адресу `https://pypi.org/simple/`. Чтобы использовать собственный индекс, укажите его в формате **PEP 503**[231] в одной из переменных окружения:
  - `PIP_INDEX_URL` – вместо PyPi;
  - `PIP_EXTRA_INDEX_URL` – в дополнение к PyPi.

  Подробности см. в документации PIP[232].
- Для корректной работы системного менеджера пакетов необходим доступ к соответствующему репозиторию или его зеркалу. Настройку системного менеджера пакетов рекомендуется выполнять с помощью команд дополнительного этапа `additional_build_steps.prepend_base`. Пример настройки см. ниже.
- Для загрузки зависимостей Ansible с помощью `ansible-galaxy` укажите в файле `ansible.cfg` параметры доступа к реестрам коллекций. Пример настройки для загрузки зависимостей Ansible из Private Automation Hub см. ниже.

Примеры

Изучите на примерах применение Ansible Builder для сборки собственных образов среды исполнения и среды принятия решений. Предварительно выполните следующие действия:

1. Настройте *доступ к Private Automation Hub*.
2. Создайте каталог для хранения файлов проекта.

> **Примечание**
>
> Создавать файлы и выполнять команды далее следует в каталоге проекта, если явно не указано иное.

Установка системных пакетов из репозиториев

Чтобы добавить в образ на базе `hub.astra-automation.ru/aa-1.2/aa-minimal-ee:latest` пакеты из DEB-репозиториев Astra Linux Special Edition, используйте системный менеджер пакетов:

1. Определите версию Astra Linux Special Edition, на основе которой построен базовый образ:

[230] https://pypi.org/
[231] https://peps.python.org/pep-0503/
[232] https://pip.pypa.io/en/stable/cli/pip_install/

```
podman run --rm <image>:<tag> cat /etc/astra_version
```

Здесь <image> и <tag> – соответственно название и метка базового образа.

2. Убедитесь, что соответствующие репозитории Astra Linux Special Edition или их зеркала доступны с локальной машины.
3. Создайте файл `bindep.txt` со списком системных пакетов.
4. Файл определения среды исполнения заполните следующим образом:

Список 5: `execution-environment.yml`

```
---
version: 3
images:
  base_image:
    name: hub.astra-automation.ru/aa-1.2/aa-minimal-ee:latest
dependencies:
  system: bindep.txt
additional_build_steps:
  prepend_base:
    - RUN apt-get update
  append_final:
    - RUN apt-get clean && rm -rf /var/lib/apt/lists/*
```

При использовании Ansible Builder в закрытом контуре добавьте в блок `additional_build_steps.prepend_base` дополнительные шаги по настройке APT на работу с собственными зеркалами репозиториев, например:

Список 6: `execution-environment.yml`

```
---
version: 3
images:
  base_image:
    name: hub.astra-automation.ru/aa-1.2/aa-minimal-ee:latest
dependencies:
  system: bindep.txt
additional_build_steps:
  prepend_base: |
    RUN echo 'deb https://dl.local/astra/stable/main-repository 1.8_x86-64 main non-
↪free contrib' | tee /etc/apt/sources.list
    RUN echo 'deb https://dl.local/astra/stable/extended-repository 1.8_x86-64 main␣
↪non-free-contrib' | tee -a /etc/apt/sources.list
    RUN apt-get update
  append_final:
    - RUN apt-get clean && rm -rf /var/lib/apt/lists/*
```

Здесь:

1. На дополнительном шаге `prepend_base` содержимое файла `/etc/apt/sources.list` заменяется ссылками на локальные зеркала репозиториев, после чего обновляется список доступных пакетов. Это обеспечивает возможность установки с помощью APT пакетов, указанных в файле системных зависимостей `bindep.txt`.
2. На этапе `prepend_base` кеш APT очищается, тем самым позволяя уменьшить размер образа.

5. Запустите сборку образа.

Установка библиотек Python

Чтобы добавить в образ библиотеки Python, выполните следующие действия:

1. Создайте файл зависимостей Python `requirements.txt` и заполните его в соответствии с требованиями **PEP 508**[233], например:

Список 7: `requirements.txt`

```
yamllint
black==24.10.0
```

2. Файл определения среды исполнения заполните следующим образом:

Список 8: `execution-environment.yml`

```
---
version: 3
images:
  base_image:
    name: hub.astra-automation.ru/aa-1.2/aa-minimal-ee:latest
dependencies:
  python: requirements.txt
```

3. Запустите сборку образа.

Добавление коллекции из архива

Чтобы добавить в образ коллекцию Ansible в формате архива `.tar.gz`, выполните следующие действия:

1. Создайте файл со списком зависимостей Ansible, например:

Список 9: `requirements.yml`

```
---
collections:
  - name: astra.ald_pro
  - name: astra.nginx
    version: 1.8.2
```

2. Загрузите коллекции:

```
ansible-navigator exec -- ansible-galaxy collection download -r requirements.yml
```

В каталоге проекта будет создан подкаталог `collections/`, в который будут загружены указанные коллекции.

3. Файл определения среды исполнения заполните следующим образом:

Список 10: `execution-environment.yml`

```
---
version: 3

images:
  base_image:
    name: hub.astra-automation.ru/aa-1.2/aa-minimal-ee:latest

dependencies:
  galaxy: collections/requirements.yml
```

4. Создайте контекст сборки:

[233] https://peps.python.org/pep-0508/

```
ansible-builder create
```

5. Скопируйте загруженные коллекции в контекст сборки:

```
cp collections/*.tar.gz context/_build/
```

6. Запустите сборку образа.

Добавление коллекций из приватного репозитория Git

Доступ к приватному репозиторию Git можно получить разными способами. В данном примере рассматривается использование приватного ключа SSH.

Чтобы добавить коллекцию из приватного репозитория Git, на этапе сборки необходимо передать полномочия в образ. Для этого выполните следующие действия:

1. В каталоге проекта разместите файл приватного ключа SSH `id_rsa`, позволяющий получить доступ к репозиторию.

> **Примечание**
>
> Приватный ключ не должен быть защищен паролем.

2. Создайте файл зависимостей Ansible, например:

Список 11: `requirements.yml`

```yaml
---
collections:
  - name: my.collection
    url: https://git.example.com/private/my-collection.git
```

3. Файл определения среды исполнения заполните следующим образом:

Список 12: `execution-environment.yml`

```yaml
---
version: 3

images:
  base_image:
    name: hub.astra-automation.ru/aa-1.2/aa-minimal-ee:latest

dependencies:
  galaxy: requirements.yml

additional_build_files:
  - src: id_rsa
    dest: ssh

additional_build_steps:
  prepend_galaxy:
    - RUN mkdir /root/.ssh/
    - RUN printf 'Host git.example.com\n\tIdentityFile /root/.ssh/id_rsa\
↪nStrictHostKeyChecking no\n' > /root/.ssh/config
    - COPY --chmod=0600 ./_build/ssh/id_rsa /root/.ssh/id_rsa
  append_galaxy:
    - RUN rm -rf /root/.ssh
```

4. Запустите сборку образа.

Добавление коллекций из Private Automation Hub

Чтобы добавить в образ среды исполнения коллекции, размещенные в Private Automation Hub, выполните следующие действия:

1. Создайте токен для доступа к Private Automation Hub согласно *инструкции*.
2. Создайте подкаталог `files/`, а в нем – файл `ansible.cfg` с настройками подключения к репозиториям Private Automation Hub, например:

Список 13: `ansible.cfg`

```
[galaxy]
server_list = galaxy, validated

[galaxy_server.galaxy]
url = https://galaxy.ansible.com/

[galaxy_server.validated]
url = https://private-hub.example.com/api/galaxy/content/validated/
```

Здесь указаны публично доступный репозиторий Ansible Galaxy[234] и репозиторий `validated`, размещенный в Private Automation Hub.

3. Создайте файл зависимостей Ansible, например:

Список 14: `requirements.yml`

```
collections:
  - name: astra.ald_pro
    version: 2.0.0
  - name: astra.cups
    version: 1.3.3
  - name: astra.rubackup
  - name: astra.rupost
```

4. Файл определения среды исполнения заполните следующим образом:

Список 15: `execution-environment.yml`

```
---
version: 3

dependencies:
  galaxy: requirements.yml

images:
  base_image:
    name: hub.astra-automation.ru/aa-1.2/aa-minimal-ee:latest

additional_build_files:
  - src: ansible.cfg
    dest: configs

additional_build_steps:
  prepend_galaxy:
    - COPY _build/configs/ansible.cfg /etc/ansible/ansible.cfg
    - ARG ANSIBLE_GALAXY_SERVER_VALIDATED_TOKEN
```

Для получения доступа к Private Automation Hub используется токен, заданный в переменной окружения `ANSIBLE_GALAXY_SERVER_VALIDATED_TOKEN`. Значение этой переменной импортируется в образ из хостовой системы.

Название переменной окружения, в которой хранится токен для доступа к реестру, формируется по шаблону:

[234] https://galaxy.ansible.com/

```
ANSIBLE_GALAXY_SERVER_<NAME>_TOKEN
```

Здесь <NAME> – набранное в верхнем регистре название сервера, указанное в ansible.cfg. Например, если бы серверы назывались published и testing_dev, соответствующие переменные окружения имели бы названия ANSIBLE_GALAXY_SERVER_PUBLISHED_TOKEN и ANSIBLE_GALAXY_SERVER_TESTING_DEV_TOKEN.

5. Экспортируйте токен для доступа к Private Automation Hub в соответствующую переменную окружения:

```
export ANSIBLE_GALAXY_SERVER_VALIDATED_TOKEN=<token>
```

6. Запустите сборку образа, передав в аргументе --build-arg название переменной окружения, в которой хранится токен для доступа к Private Automation Hub:

```
ansible-navigator builder build \
  -t <image>:<tag> \
  --build-arg ANSIBLE_GALAXY_SERVER_VALIDATED_TOKEN
```

Создание образа среды принятия решений

Для создания нового образа среды принятия решений выполните следующие действия:

1. В файле определения среды исполнения укажите в значении поля images.base_image.name ссылку на один из образов среды принятия решений, доступных в реестре Automation Hub[235], например:

Список 16: execution-environment.yml

```
---
version: 3

images:
  base_image:
    name: hub.astra-automation.ru/aa-1.2/aa-minimal-de:latest

dependencies:
  galaxy:
    collections:
      - ansible.eda
    python_interpreter:
      package_system: "python3.11"
```

2. Запустите сборку образа.

17.3.3 Ansible Creator

Ansible Creator – это инструмент командной строки, упрощающий создание коллекций и наборов сценариев.

Установка

Для установки Ansible Creator выполните следующие действия:

1. Подключите репозиторий Astra Automation.

Инструкция по подключению репозитория

1. В каталоге /etc/apt/sources.list.d/ создайте файл astra-automation.list со ссылкой на репозиторий Astra Automation:

[235] https://hub.astra-automation.ru/

```
deb https://dl.astralinux.ru/aa/aa-debs-for-alse-1.8 <version> main
```

Вместо <version> необходимо подставить версию устанавливаемой платформы, например, `2.0`.

Доступные версии продукта опубликованы в таблице *История обновлений*.

2. Обновите список доступных пакетов:

```
sudo apt update
```

2. Установите пакет `ansible-creator`:

```
sudo apt install ansible-creator --yes
```

Установка при отсутствии доступа к интернету описана в документе *Средства разработки*.

Применение

Для инициализации структуры файлов и каталогов коллекции выполните команду:

```
ansible-creator init collection <namespace>.<collection> <dir>
```

Здесь:

- <namespace> – пространство имен;
- <collection> – название коллекции;
- <dir> – каталог, в котором следует разместить файлы и каталоги коллекции.

  Если этот параметр не указан, файлы и каталоги будут размещены в текущем каталоге.

Для инициализации структуры файлов и каталогов набора сценариев выполните команду:

```
ansible-creator init playbook <namespace>.<collection> <dir>
```

Параметры этой команды аналогичны параметрам предыдущей, однако, создаваемая структура файлов и каталогов будет отличаться.

Примечание

Одним из требований утилиты Ansible Creator является распространение набора сценариев в составе коллекции. По этой причине в аргументах команды необходимо указать ее название.

Полный перечень возможных команд и опций см. в *справочнике*.

Примеры

Изучите на примерах применение Ansible Creator для создания структуры файлов и каталогов ваших проектов Ansible.

Создание коллекции

Для создания коллекции выполните команду:

```
ansible-creator init collection astra.example ./astra-example
```

Эта команда создаст в текущем каталоге подкаталог `astra-example/`, а внутри него разместит файлы и каталоги, необходимые для начала работы над коллекцией `astra.example`.

Набор сценариев

Для создания структуры файлов и каталогов набора сценариев выполните команду:

```
ansible-creator init playbook astra.playbooks ./astra-playbooks
```

Эта команда создаст в текущем каталоге подкаталог `astra-playbooks/`, а внутри него разместит файлы и каталоги, необходимые для начала работы над сценариями.

17.3.4 Ansible Sign

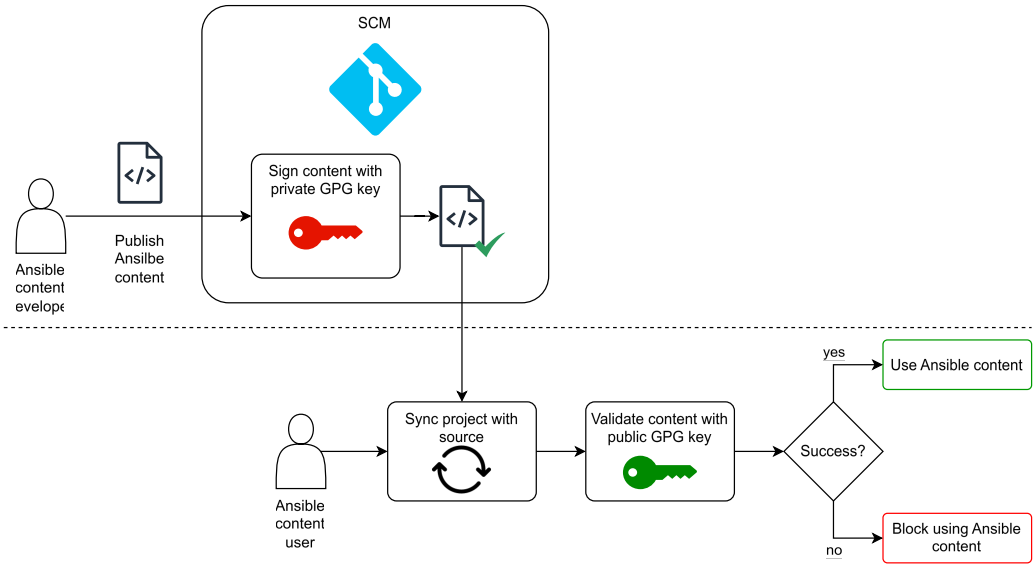
Ansible Sign – это инструмент командной строки, позволяющий обеспечить надежную передачу содержимого Ansible от разработчиков к пользователям с помощью цифровой подписи. Это простой способ обеспечить защиту содержимого, исключающий возможность его несанкционированного изменения и повышающий надежность автоматизации, особенно в критически важных или многопользовательских средах.

Для создания и проверки цифровой подписи Ansible Sign использует GPG – набор утилит от проекта GNU, реализующих протокол OpenGPG.

Примечание

Подробности использования GPG доступны на сайте проекта The GNU Privacy Guard[236], а описание протокола OpenPGP в **RFC 4880**[237].

Применение Ansible Sign для контроля целостности проекта показано на схеме:



1. Разработчик публикует проект Ansible в SCM-системе, например, в репозитории GitFlic.
2. SCM-система подписывает содержимое приватным ключом GPG.
3. Пользователь синхронизирует содержимое с SCM-системой.
4. Пользователь запускает проверку целостности полученного проекта, используя публичный ключ GPG. Утилита Ansible Sign проверяет цифровую подпись и контрольные суммы файлов.

   Проверка может быть выполнена следующими способами:

   - вручную;
   - автоматически.

[236] https://www.gnupg.org/
[237] https://datatracker.ietf.org/doc/html/rfc4880.html

Для автоматической проверки в Automation Controller или контроллере EDA создайте полномочие типа «Публичный ключ GPG». Если проверка целостности не будет успешной, Automation Controller и контроллер EDA заблокируют использование содержимого.

Установка

Для установки Ansible Sign выполните следующие действия:

1. Подключите репозиторий Astra Automation.

   #### Инструкция по подключению репозитория

   1. В каталоге `/etc/apt/sources.list.d/` создайте файл `astra-automation.list` со ссылкой на репозиторий Astra Automation:

      ```
      deb https://dl.astralinux.ru/aa/aa-debs-for-alse-1.8 <version> main
      ```

      Вместо <version> необходимо подставить версию устанавливаемой платформы, например, `2.0`.

      Доступные версии продукта опубликованы в таблице *История обновлений*.

   2. Обновите список доступных пакетов:

      ```
      sudo apt update
      ```

2. Установите пакет `ansible-sign`:

   ```
   sudo apt install ansible-sign --yes
   ```

Установка при отсутствии доступа к интернету описана в документе *Средства разработки*.

Создание хранилища и ключей

Для создания пары ключей GPG выполните следующие действия:

1. Проверьте наличие хранилища ключей:

   ```
   gpg --list-secret-keys
   ```

   Если хранилище не существует, утилита создаст его, о чем свидетельствует следующий вывод в терминал:

   ```
   gpg: создан каталог '/home/<user>/.gnupg'
   gpg: создан щит с ключами '/home/<user>/.gnupg/pubring.kbx'
   gpg: /home/<user>/.gnupg/trustdb.gpg: создана таблица доверия
   ```

   Если хранилище существует, но не содержит ключей, вывод утилиты будет пустым.

   Если в хранилище уже существуют ключи, утилита выводит информацию о них, например:

   ```
   gpg: проверка таблицы доверия
   gpg: marginals needed: 3  completes needed: 1  trust model: pgp
   gpg: глубина: 0  достоверных:   1  подписанных:   0  доверие: 0-, 0q, 0n, 0m, 0f, 1u
   gpg: срок следующей проверки таблицы доверия 2035-02-04
   /home/<user>/.gnupg/pubring.kbx
   --------------------------------
   sec   rsa4096 2025-02-06 [SC] [   годен до: 2035-02-04]
         6CE032B47A1EA571A25FC228455114CAF74D0C6E
   uid         [  абсолютно ] Example User (Astra Automation Example GPG Key)
    ↪<user@example.com>
   ssb   rsa4096 2025-02-06 [E] [   годен до: 2035-02-04]
   ```

2. Для создания пары ключей выполните команду:

```
gpg --full-generate-key
```

В терминал выводится приглашение для выбора типа ключа:

```
gpg (GnuPG) 2.2.27; Copyright (C) 2021 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Выберите тип ключа:
   (1) RSA и RSA (по умолчанию)
   (2) DSA и Elgamal
   (3) DSA (только для подписи)
   (4) RSA (только для подписи)
  (14) Имеющийся на карте ключ
  (14) GOST R34.10-2001 (sign only) OBSOLETE
  (15) GOST R34.10-2012 (sign only)
Ваш выбор?
```

3. Для выбора нужного типа ключа укажите число, стоящее в скобках рядом с названием, и нажмите Enter.

Совет

Рекомендуется использовать ключи типа RSA и RSA.

В терминал выводится приглашение для выбора длины ключа:

```
длина ключей RSA может быть от 1024 до 4096.
Какой размер ключа Вам необходим? (3072)
```

4. Введите нужную длину ключа и нажмите Enter.

Совет

Рекомендуется использовать ключи длиной не менее 4096 бит.

В терминал выводится приглашение для ввода срока действия ключа:

```
Выберите срок действия ключа.
         0 = не ограничен
      <n>  = срок действия ключа - n дней
      <n>w = срок действия ключа - n недель
      <n>m = срок действия ключа - n месяцев
      <n>y = срок действия ключа - n лет
Срок действия ключа?
```

5. Укажите срок действия ключа и нажмите Enter.

   В терминал выводится запрос на подтверждение корректности выбранного срока действия ключа:

```
Срок действия ключа не ограничен
Все верно? (y/N)
```

6. Для подтверждения корректности выбранного срока действия ключа введите y и нажмите Enter.

   В терминал выводится приглашение ко вводу вашего полного имени:

```
GnuPG должен составить идентификатор пользователя для идентификации ключа.

Ваше полное имя:
```

7. Введите латиницей ваше полное имя, например, `John Dow`, и нажмите `Enter`.

   В терминал выводится приглашение ко вводу адреса электронной почты:

```
Адрес электронной почты:
```

8. Введите адрес электронной почты и нажмите `Enter`.

   В терминал выводится приглашение ко вводу примечания.

9. Введите текст примечания и нажмите `Enter`.

   В терминал выводится запрос на подтверждение корректности введенных данных, например:

```
Ваше полное имя: John Dow
Адрес электронной почты: johndow@example.com
Примечание: John Dow GPG Key
Вы выбрали следующий идентификатор пользователя:
    "John Dow (John Dow GPG Key) <johndow@example.com>"

Сменить (N)Имя, (C)Примечание, (E)Адрес; (O)Принять/(Q)Выход?
```

10. Чтобы подтвердить корректность введенных данных, введите `O` и нажмите `Enter`.

    В терминал выводится диалоговое окно для ввода пароля для защиты ключа:

```
┌──────────────────────────────────────────────────────┐
│ Введите фразу-пароль                                 │
│ для защиты нового ключа                              │
│                                                      │
│ Фраза-пароль: _____________________________________  │
│                                                      │
│        <OK>                          <Отмена (C)>    │
└──────────────────────────────────────────────────────┘
```

11. Введите пароль и нажмите `Enter`.

    В терминал выводится диалоговое окно для ввода подтверждения пароля:

```
┌──────────────────────────────────────────────────────┐
│ Повторите фразу-пароль:                              │
│                                                      │
│ Фраза-пароль: _____________________________________  │
│                                                      │
│        <OK>                          <Отмена (C)>    │
└──────────────────────────────────────────────────────┘
```

12. Введите пароль еще раз и нажмите `Enter`.

    В терминал выводится сообщение о необходимости получения множества случайных чисел:

```
Необходимо получить много случайных чисел. Желательно, чтобы Вы
в процессе генерации выполняли какие-то другие действия (печать
на клавиатуре, движения мыши, обращения к дискам); это даст генератору
случайных чисел больше возможностей получить достаточное количество энтропии.
Необходимо получить много случайных чисел. Желательно, чтобы Вы
в процессе генерации выполняли какие-то другие действия (печать
на клавиатуре, движения мыши, обращения к дискам); это даст генератору
случайных чисел больше возможностей получить достаточное количество энтропии.
```

13. Для увеличения энтропии случайным образом двигайте мышь и нажимайте клавиши на клавиатуре до тех пор, пока в терминал не выводится сообщение о создании ключей и сертификата, например:

```
gpg: ключ A92BB253E671651D помечен как абсолютно доверенный
gpg: сертификат отзыва записан в '/home/<user>/.gnupg/openpgp-revocs.d/
↪A90B44D4C5967DDF9280A737A92BB253E671651D.rev'.
открытый и секретный ключи созданы и подписаны.

pub   rsa4096 2025-02-06 [SC]
      A90B44D4C5967DDF9280A737A92BB253E671651D
uid                      John Dow (John Dow GPG Key) <johndow@example.com>
sub   rsa4096 2025-02-06 [E]
```

Распространение публичного ключа

Чтобы подготовить публичный ключ GPG к передаче пользователям, выполните следующие действия:

1. Получите список существующих ключей:

   ```
   gpg --list-secret-keys
   ```

   Команда выводит в терминал список ключей и их *отпечатков* (fingerprints), например:

   ```
   sec   rsa4096 2025-02-06 [SC]
         A90B44D4C5967DDF9280A737A92BB253E671651D
   uid         [  абсолютно ] John Dow (John Dow Personal GPG key) <johndow@example.
   ↪com>
   ssb   rsa4096 2025-02-06 [E]
   ```

   Здесь A90B44D4C5967DDF9280A737A92BB253E671651D – отпечаток ключа.

2. Выполните команду экспорта публичной части ключа:

   ```
   gpg --export --armor <fingerprint> > <file>
   ```

   Здесь:

   - <fingerprint> – отпечаток ключа;
   - <file> – путь к файлу для сохранения публичного ключа. Как правило, такие файлы имеют расширение `.asc`. Например, файл с ключом GPG, которым подписаны файлы в APT-репозиториях PostgreSQL, имеет название ACCC4CF8.asc[238].

3. Любым удобным способом передайте файл с публичным ключом пользователям содержимого Ansible.

   Чтобы использовать публичный ключ GPG для автоматической проверки содержимого при использовании в Automation Controller или контроллере EDA, выполните следующие действия:

   1. Используйте файл с публичным ключом GPG для создания полномочия типа «Открытый ключ GPG» (GPG Public Key).
   2. В настройках проекта заполните поле **Учетные данные для проверки подписи содержимого** (Content Signature Validation Credential), указав созданное полномочие.
   3. Синхронизируйте проект с источником.

   Примечание

   Инструкции по созданию полномочий, настройке и синхронизации проектов приведены в соответствующих разделах.

[238] https://www.postgresql.org/media/keys/ACCC4CF8.asc

Импорт ключей

Импорт ключей необходим для проверки целостности полученного содержимого Ansible. По умолчанию ключи импортируются в стандартное хранилище – файл `~/.gnupg/trustdb.gpg`. В некоторых случаях целесообразно использовать отдельное хранилище для ключей GPG. Чтобы импортировать ключ GPG в отдельное хранилище, в аргументах команды `gpg` укажите опции `--no-default-keyring` и `--keyring`.

Чтобы импортировать публичный ключ в отдельное хранилище, выполните следующие действия:

1. Загрузите ключ в формате `.asc` на компьютер.
2. Создайте хранилище ключей и импортируйте в него загруженный ключ:

   ```
   gpg \
       --no-default-keyring \
       --keyring path/to/custom-keyring.gpg \
       --import path/to/public/key.asc
   ```

   Здесь:

   - `--no-default-keyring` – запрет использования стандартного хранилища;
   - `--keyring` – путь к хранилищу, в которое импортируется ключ;
   - `--import` – путь к файлу ключа.

3. Убедитесь, что ключ успешно импортирован:

   ```
   gpg \
       --no-default-keyring \
       --keyring path/to/custom-keyring.gpg \
       --list-keys
   ```

   Если импорт выполнен успешно, в терминал выводится список ключей.

4. Чтобы использовать импортированные ключи для проверки целостности содержимого, укажите путь к хранилищу в опции `--keyring` команды `ansible-sign`:

   ```
   ansible-sign project gpg-verify . --keyring path/to/custom-keyring.gpg
   ```

Подписывание содержимого

Процесс подписывания содержимого Ansible состоит из двух этапов:

1. Создание файла `MANIFEST.in`.

   Этот файл содержит директивы, управляющие формирование списка файлов, которые необходимо контролировать.

2. Запуск утилиты `ansible-sign`.

   Утилита `ansible-sign` выполняет следующие операции:

   1. Формирует список файлов проекта на основе содержимого файла `MANIFEST.in`.
   2. Рассчитывает контрольные суммы указанных файлов.
   3. Создает в каталоге проекта подкаталог `.ansible-sign/`, содержащий следующие файлы:
      - `sha256sum.txt` – контрольные суммы файлов;
      - `sha256sum.txt.sig` – цифровая подпись файла с контрольными суммами.

Создание файла MANIFEST.in

В корневом каталоге проекта создайте файл MANIFEST.in. В этом файле укажите директивы, управляющие списком файлов для расчета контрольных сумм.

Директивы

В файле MANIFEST.in допускается использование следующих директив:

- **include** <templates>

  Включает размещенные в корневом каталоге проекта файлы, названия которых удовлетворяют шаблонам <templates>.

  Например, так можно включить все файлы с расширениями .yaml и .md в корневом каталоге:

  ```
  include *.yaml *.md
  ```

  recursive-include <directory> <templates>

  Рекурсивно включает из каталогов, названия которых удовлетворяют шаблону <directory>, все файлы, названия которых удовлетворяют шаблонам <templates>.

  Например, так можно включить все файлы с расширениями .yaml и .md, размещенные в каталоге roles/:

  ```
  recursive-include roles/ *.yaml *.md
  ```

  global-include <templates>

  Включает все файлы, названия которых удовлетворяет указанным шаблонам, независимо от каталога, в котором они находятся.

  Например, так можно включить все файлы с расширением .yaml:

  ```
  global-include *.yaml
  ```

  exclude <templates>

  Исключает размещенные в корневом каталоге проекта файлы, названия которых удовлетворяют указанным шаблонам <templates>.

  Например, так можно исключить файлы .log в корневом каталоге:

  ```
  exclude *.log
  ```

  recursive-exclude <directory> <templates>

  Исключает из каталога <directory> и его подкаталогов все файлы, названия которых удовлетворяют шаблонам. Например, так можно исключить все файлы с расширением .bak, размещенные в каталоге src/ и его подкаталогах:

  ```
  recursive-exclude src/ *.bak
  ```

  global-exclude <templates>

  Исключает все файлы проекта, названия которых удовлетворяют указанным шаблонам. Например, так можно исключить все файлы с расширениями .tmp и .log:

  ```
  global-exclude *.tmp *.log
  ```

  grafs <templates>

  Рекурсивно включает содержимое каталогов, названия которых удовлетворяют указанному шаблону.

  prune <templates>

  Рекурсивно исключает содержимое каталогов, названия которых удовлетворяют указанному шаблону.

Ansible Sign обрабатывает символы в шаблонах следующим образом:

- * – любой символ любое количество раз.
- ? – любой символ.
- [chars] – любой символ из перечисленных.

  Чтобы задать диапазон символов, используйте дефис, например:

  ```
  global-exclude *.py[cod]
  ```

  Эта директива исключит из списка подписываемых все файлы с расширениями .pyc, .pyo и .pyd.

Названия каталогов рассчитываются относительно корневого каталога проекта.

Директивы обрабатываются в порядке следования в файле MANIFEST.in, например:

```
graft tests
global-exclude *.py[cod]
```

Директива graft включит в список все содержимое каталога tests/ и его подкаталогов. Следующая после нее директива global-exclude исключит из списка файлы с расширениями .pyc, .pyo и .pyd.

Запуск ansible-sign

Чтобы подписать проект, в его корневом каталоге выполните команду:

```
ansible-sign project gpg-sign .
```

Чтобы использовать для подписывания проекта определенный приватный ключ, укажите его отпечаток в опции --fingerprint:

```
ansible-sign project gpg-sign . --fingerprint <fingerprint>
```

Подробное описание аргументов утилиты ansible-sign см. в *справочнике*.

Проверка целостности

Проверка целостности содержимого Ansible может быть выполнена вручную или автоматически.

Ручная проверка целостности

Для ручной проверки целостности содержимого Ansible выполните в корневом каталоге проекта команду:

```
ansible-sign project gpg-verify .
```

В терминал выводятся результаты проверки, например:

```
[OK   ] GPG signature verification succeeded.
[OK   ] Checksum validation succeeded.
```

Если контрольная сумма хотя бы одного файла не соответствует ожидаемой, в терминал выводится соответствующее сообщение:

```
[OK   ] GPG signature verification succeeded.
[ERROR] Checksum validation failed.
[ERROR] Checksum mismatch: README.md
```

Автоматическая проверка целостности

Automation Controller и контроллер Event-Driven Automation выполняют проверку целостности содержимого Ansible автоматически, если с проектом связано полномочие типа «Публичный ключ GPG» (GPG public key).

Инструкции по созданию полномочия:

- *для Automation Controller*;
- *для контроллера Event-Driven Automation*.

Примеры

Изучите использование Ansible Sign на примере подписывания файлов проекта.

1. Если у вас нет хранилища и пары ключей GPG, создайте их согласно *инструкции*.
2. В корневом каталоге проекта создайте файл `MANIFEST.in`, например:

```
include README.md
recursive-include playbooks *.yml
prune .git
```

3. Выполните команду:

```
ansible-sign project --gpg-sign .
```

   В терминал выводится диалоговое окно для ввода пароля секретного ключа OpenPGP:

```
┌──────────────────────────────────────────────────────────────────────┐
│ Введите фразу-пароль для разблокировки секретного ключа OpenPGP:     │
│ "Example User (Astra Automation Example GPG Key) <user@example.com>" │
│ 4096-битный ключ RSA, идентификатор 455114CAF74D0C6E,                │
│ создан 2025-02-06.                                                   │
│                                                                      │
│                                                                      │
│ Фраза-пароль: _____________________________________________________  │
│                                                                      │
│          <OK>                                  <Отмена (C)>          │
└──────────────────────────────────────────────────────────────────────┘
```

4. Введите пароль секретного ключа OpenPGP и нажмите `Enter`.

   В терминал будут выведены сообщения о ходе и результатах расчета контрольных сумм:

```
[OK   ] GPG signing successful!
[NOTE ] Checksum manifest: ./.ansible-sign/sha256sum.txt
[NOTE ] GPG summary: signature created
```

17.4 Средства тестирования

В состав Astra Automation входят следующие средства для тестирования контента Ansible:

- *Ansible Lint*;
- *Ansible Molecule*;
- *Ansible Pytest*;
- *Tox Ansible*;
- *Ansible Test*.

17.4.1 Сравнительные характеристики

Сравнительные характеристики утилит представлены в таблице:

Название	Назначение	Тестовая среда	Интеграция с другими утилитами тестирования	Другие особенности
Ansible Lint	• Проверка качества и стиля кода Ansible (сценариев, ролей, коллекций). • Обеспечение качества кода и соблюдения стандартов при разработке кода Ansible	Статический анализ кода без развертывания тестовых окружений	Интеграция с рабочими процессами CI/CD и редакторами, но без возможности запуска тестовых сценариев	• Автоматическая проверка кода Ansible на соответствие лучшим практикам. • Адаптация кода к новым версиям Ansible. • Быстрое выявление потенциальных проблем
Ansible Molecule	• Разработка и тестирование Ansible ролей: проверка корректности работы, идемпотентности и совместимости • Тестирование ролей Ansible в различных средах и конфигурациях	Изолированные тестовые среды (контейнеры, виртуальные машины) с использованием драйверов Docker, Vagrant, Podman и т. д.	• Интеграция с Ansible (использует его как компонент, который отвечает за настройку тестовой среды после ее создания). • Позволяет использовать верификаторы типа Testinfra и Goss, может быть интегрирована с Pytest	• Модульная архитектура с последовательным жизненным циклом: 1. destroy; 2. dependency; 3. syntax; 4. create; 5. converge; 6. idempotence; 7. verify; 8. cleanup. • Поддержка множества операционных систем и систем виртуализации
Ansible Pytest	• Тестирование задач Ansible посредством pytest с использованием	Тестовое окружение pytest, позволяющее выполнять тесты на локальных и удаленных уз-	• Интеграция с Ansible через готовые модули – фикстуры	• Гибкие возможности параметризации тестов.

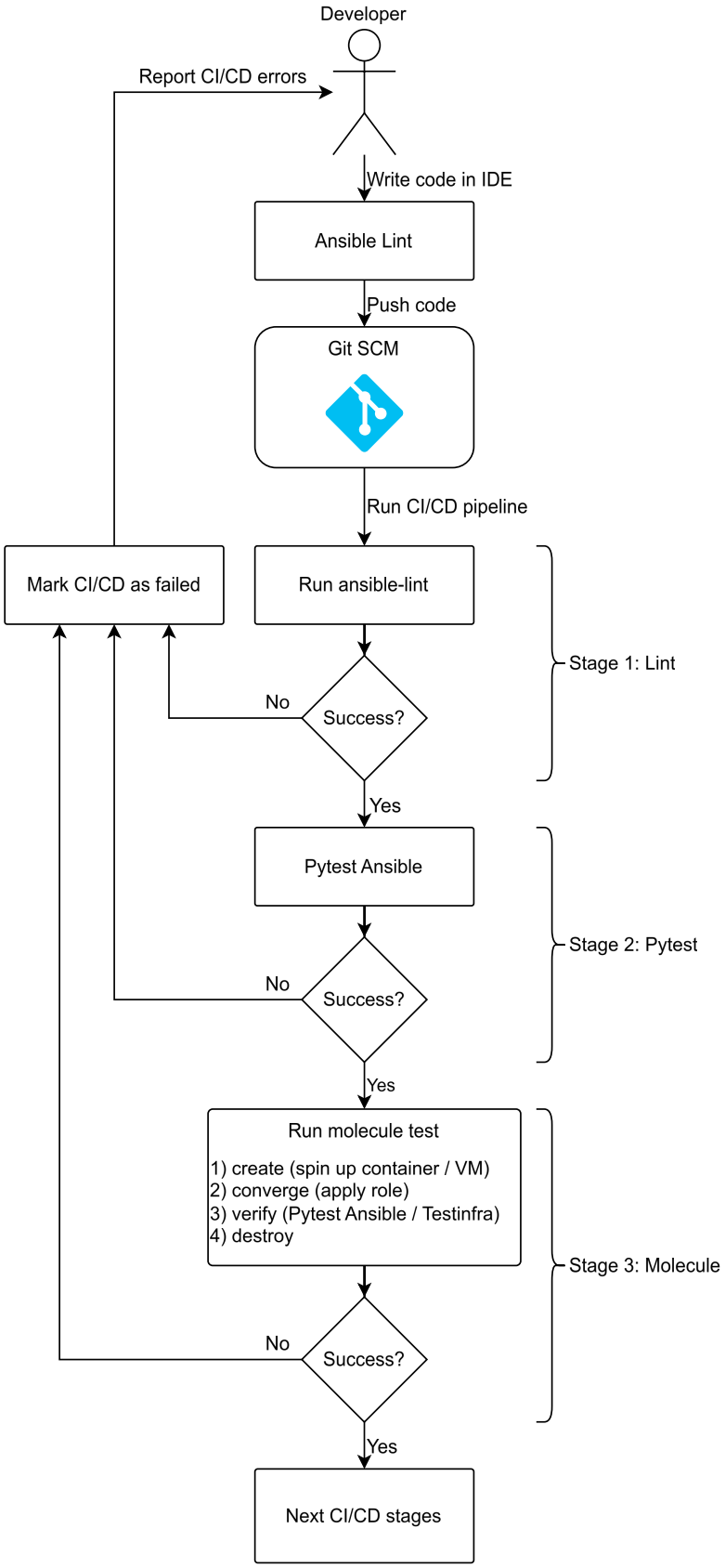
17.4.2 Рабочий процесс

Рабочий процесс может включать как все указанные средства, так и только часть из них.

Ansible Lint, Ansilbe Pytest, Molecule

Рабочий процесс с использованием Ansible Lint, Ansible Pytest и Molecule показан на схеме:

[239] https://github.com/ansible



Здесь показана следующая последовательность использования средств тестирования кода Ansible:

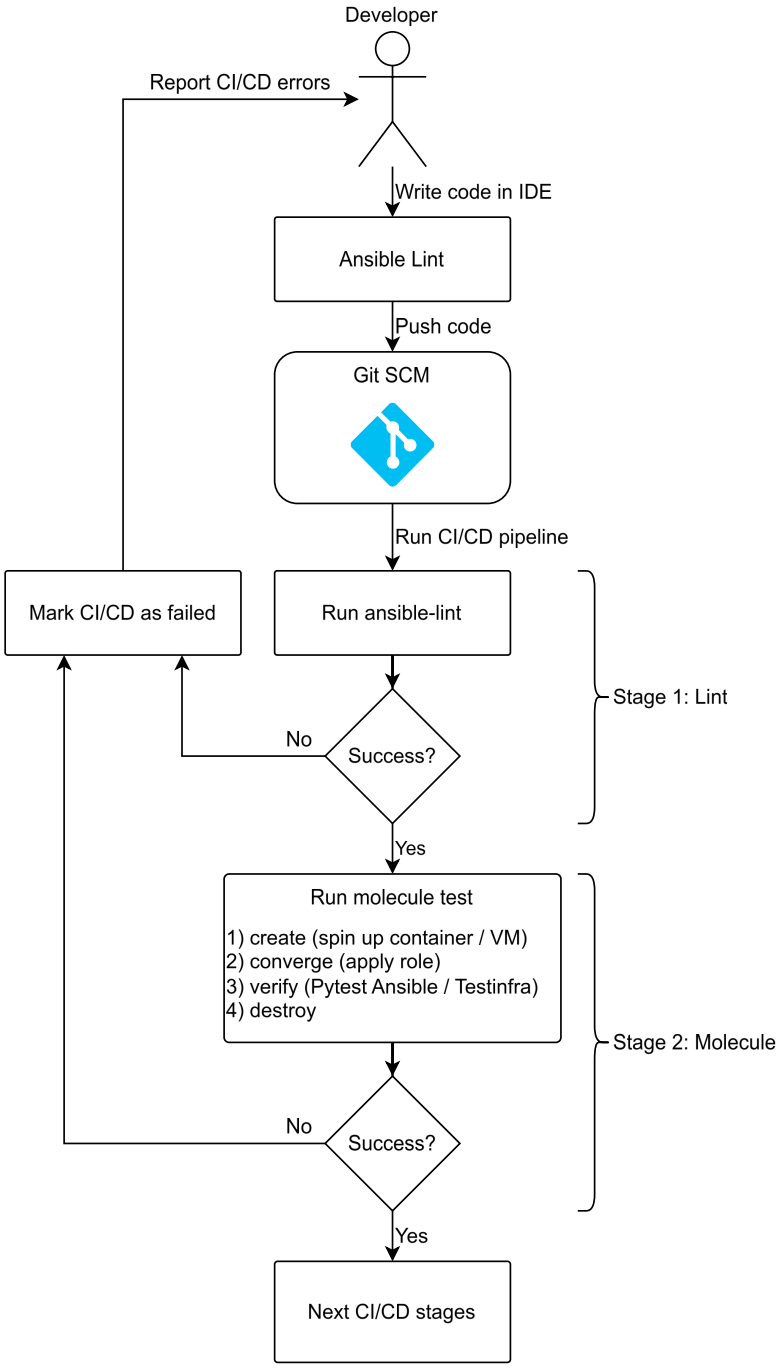
1. Разработчик использует Ansible Lint на этапе написания кода. Это позволяет выявить часть ошибок до попадания кода в репозиторий проекта.
2. Разработчик публикует код в репозиторий проекта. Это приводит к срабатыванию триггера SCM и запуску процесса CI/CD.

3. Проверка кода проекта с помощью Ansible Lint. Повторная проверка позволяет предотвратить ситуацию, когда разработчик не использует Ansible Lint на своей рабочей станции.
4. Запуск тестов Pytest Ansible.
5. Запуск тестов Molecule для проверки работоспособности ролей:
    1. Molecule создает необходимую для тестов инфраструктуру.
    2. Pytest Ansible проверяет корректность развертывания инфраструктуры.
    3. Molecule применяет роли.
    4. Pytest Ansible или Testinfra проверяет результат применения ролей.
    5. Molecule уничтожает созданную ранее инфраструктуру для тестов.

Если все этапы завершены без ошибок, тестирование считается выполненным успешно. Если хотя бы на одном из этапов возникнут ошибки, тестирование будет остановлено и отмечено как неуспешное, а сведения об ошибках переданы разработчику.

Ansible Lint, Molecule

Рабочий процесс с использованием Ansible Lint и Molecule показан на схеме:



Здесь показана следующая последовательность использования средств тестирования кода Ansible:

1. Разработчик использует Ansible Lint на этапе написания кода. Это позволяет выявить часть ошибок до попадания кода в репозиторий проекта.
2. Разработчик публикует код в репозиторий проекта. Это приводит к срабатыванию триггера SCM и запуску процесса CI/CD.
3. Проверка кода проекта с помощью Ansible Lint. Повторная проверка позволяет предотвратить ситуацию, когда разработчик не использует Ansible Lint на своей рабочей станции.
4. Запуск тестов Molecule для проверки работоспособности ролей:
    1. Molecule создает необходимую для тестов инфраструктуру.
    2. Pytest Ansible проверяет корректность развертывания инфраструктуры.
    3. Molecule применяет роли.

4. Pytest Ansible или Testinfra проверяет результат применения ролей.
5. Molecule уничтожает созданную ранее инфраструктуру для тестов.

Если все этапы завершены без ошибок, тестирование считается выполненным успешно. Если хотя бы на одном из этапов возникнут ошибки, тестирование будет остановлено и отмечено как неуспешное, а сведения об ошибках переданы разработчику.

Ansible Molecule

Утилита Ansible Molecule предназначена для облегчения разработки и тестирования ролей Ansible, что способствует повышению качества и надежности проектов автоматизации инфраструктуры. Ansible Molecule позволяет создавать рабочую среду для тестирования ролей (тестовую среду) в различных виртуальных средах, как облачных, так и локальных, построенных как на виртуальных машинах, так и на контейнерах.

Ключевые особенности и достоинства

Ansible Molecule имеет следующие ключевые особенности и достоинства:

- Позволяет создавать тестовые среды в различных виртуальных окружениях.

  Ansible Molecule позволяет тестировать роли Ansible в различных средах, включая множество поставщиков виртуализации и контейнеризации, операционные системы и дистрибутивы. Эта универсальность позволяет тестировать роли в разнообразных условиях.

- Используется как самостоятельный инструмент так и в составе тестовых платформ, что позволяет значительно расширить возможности Ansible Molecule путем запуска параллельных заданий, форматирования отчетов и использования других возможностей платформ.

- Ansible Molecule позволяет создавать несколько сценариев тестирования. Сценарий тестирования ориентирован на проверку выполнения сценариев автоматизации, содержащихся в указанных файлах. Тестирование в пределах одного сценария основано на выполнении определенной последовательности функциональных тестов.

  Разработчику тестов необходимо определить каждый сценарий (Ansible play) в виде набора задач (Ansible tasks), используя знакомые ему возможности Ansible. Можно запускать полную последовательность тестов или часть ее.

- Обеспечивает объявленную политику поддержки версий Ansible – поддерживает две последние основные версии Ansible. Эта политика гарантирует, что тестовая среда остается актуальной с учетом последних функций Ansible, сохраняя при этом обратную совместимость с предыдущей основной версией.

- Обеспечивает понятный синтаксис командной строки на базе утилиты `molecule`, которую можно также запускать как модуль Python `python3 -m molecule`.

  Утилита предоставляет подсказки как на глобальном уровне, так и по отдельным ее командам. Возможна настройка режима автодополнения.

Применение

Утилиту можно применять для выполнения следующих задач:

- Разработка новых ролей Ansible – предоставляет стандартизированную среду для разработки ролей с нуля, обеспечивая их соответствие стандартам качества.
- Тестирование существующих ролей Ansible – может помочь выявить проблемы совместимости с новыми версиями Ansible или в различных средах, которые не были предусмотрены изначально.
- Интеграция в CI/CD конвейеры – можно интегрировать в процессы непрерывной интеграции и развертывания для автоматизации тестирования ролей Ansible, делая процесс более надежным и эффективным.

Принцип работы

Ansible Molecule построена на основе модульной архитектуры, которая позволяет ей быть гибкой и расширяемой. Основными компонентами Ansible Molecule являются:

- Сценарии (Scenarios).

  Являются основными строительными блоками данного функционального тестирования. Они выстроены в определенную последовательность, описываемую далее.

- Драйверы (Drivers или Providers).

  Отвечают за создание, управление и удаление тестовых сред. Примеры драйверов: Docker, Podman, Vagrant, Libvirt, LXC, EC2.

- Исполнитель сценариев (Provisioners).

  Используются для исполнения сценариев в созданных тестовых средах. По умолчанию Ansible Molecule использует Ansible в качестве исполнителя.

- Средства проверки (Verifiers).

  Инструменты для контроля успешности выполнения сценариев. Позволяет использовать Testinfra, Goss и Ansible.

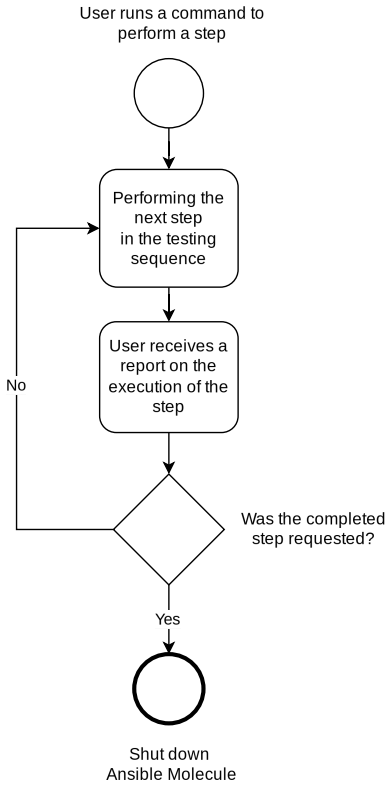
Схема работы Ansible Molecule

Ansible Molecule выполняет тестирование ролей с помощью последовательности этапов. На каждом этапе выполняется определенный сценарий. Название этапа и файла сценариев (playbook) соответствуют друг другу. Например, на этапе `destroy` выполняется сценарий из файла `destroy.yml`.

1. *destroy* удаляет предыдущую тестовую среду, если она существует, чтобы обеспечить отсутствие изменений, сделанных на предыдущих запусках тестирования.
2. *dependency* устанавливает зависимости роли, например, указанные в файле `requirements.yml`.
3. *syntax* проверяет синтаксис сценариев и файлов роли.
4. *create* создает тестовую среду с использованием выбранного драйвера.
5. *prepare* выполняет дополнительные подготовительные задачи, если они определены.
6. *converge* применяет роль к тестовой среде с помощью Ansible.
7. *idempotence* проверяет идемпотентность роли, применяя ее повторно и ожидая отсутствия изменений.
8. *verify* выполняет тесты для проверки корректности работы роли.
9. *cleanup* выполняет задачи очистки, если они определены.
10. *destroy* удаляет тестовую среду после завершения тестирования.

Этапы осуществляются в строгой последовательности, после выполнения запрошенного этапа работа утилиты завершается.

Процесс тестирования показан на схеме:



На локальном узле выполняются следующие этапы:

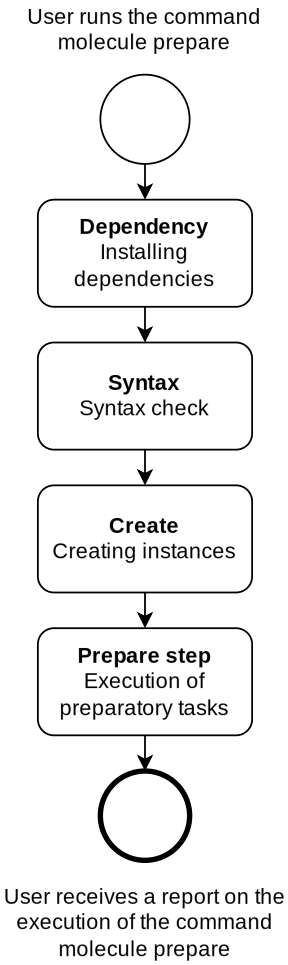
- *dependency*;
- *syntax*;
- *create*.

После создания тестовой среды в ней выполняются следующие этапы:

- *prepare*;
- *converge*;
- *idempotence*;
- *verify*;
- *cleanup*;
- *destroy*.

Если пользователь запрашивает выполнение определенного этапа, например, *prepare*, утилита выполнит также все предшествующие ему этапы.

Например, процесс выполнения этапа *prepare* показан на схеме:



Тестовые среды

В Ansible Molecule тестовые среды определяются драйвером. Их можно разделить на несколько типов в зависимости от того, какие инструменты используются:

- Контейнеры.

  При использовании драйверов типа Docker или Podman Ansible Molecule создает отдельные контейнеры для каждой тестовой платформы, определенной в `molecule.yml`. Если необходимо протестировать роль на нескольких операционных системах, Molecule создаст по одному контейнеру для каждой из них. Один контейнер используется для всех этапов (*create*, *converge*, *verify*) в рамках одного сценария. Это означает, что контейнер создается на этапе *create* и уничтожается на этапе *destroy*, а все остальные действия выполняются внутри этого контейнера.

- Виртуальные машины.

  При использовании драйверов типа Vagrant или Libvirt, Ansible Molecule создает виртуальные машины для каждой платформы.

- Виртуальные окружения.

  Ansible Molecule по умолчанию не использует виртуальные окружения Python (venv) внутри тестовых сред. Однако, если роль требует использования пакетов Python или виртуальных окружений на целевых узлах, вы можете настроить это в своих задачах Ansible.

Установка

Для установки Ansible Molecule выполните следующие действия:

1. Подключите репозиторий Astra Automation.

Инструкция по подключению репозитория

1. В каталоге `/etc/apt/sources.list.d/` создайте файл `astra-automation.list` со ссылкой на репозиторий Astra Automation:

   ```
   deb https://dl.astralinux.ru/aa/aa-debs-for-alse-1.8 <version> main
   ```

   Вместо <version> необходимо подставить версию устанавливаемой платформы, например, `2.0`.

   Доступные версии продукта опубликованы в таблице *История обновлений*.

2. Обновите список доступных пакетов:

   ```
   sudo apt update
   ```

2. Установите пакет `molecule`:

   ```
   sudo apt install molecule --yes
   ```

Настройка

Ansible Molecule можно настроить с помощью следующих компонентов:

- *команды*;
- *файл настроек*.

При вычислении действующих значений аргументов используется следующий порядок (значения, указанные на более поздних этапах, заменяют значения, указанные ранее):

1. Файл настроек.
2. Команды.

По умолчанию Ansible Molecule начинает поиск файла настроек с текущего каталога. Если в каталоге несколько сценариев, утилита проверяет структуру подкаталогов. Если указан аргумент `--scenario-name`, утилита ищет файл `molecule.yml` в соответствующем каталоге сценария.

Параметры в файле настроек задаются следующим образом:

Список 17: `molecule.yml`

```
---
dependency:
  name: galaxy
  options:
    requirements-file: requirements.yml

platforms:
  - name: molecule-astra
    image: hub.astra-automation.ru/aa-1.2/aa-full-ee
    privileged: true
    groups:
      - all

driver:
  options:
    managed: false
    login_cmd_template: "podman exec -ti {instance} bash"
    ansible_connection_options:
      ansible_connection: podman
```

Подробное описание параметров см. в *справочнике*.

Пример

В этом примере рассматривается создание роли Ansible, которая получает текущую версию Ansible и записывает ее в файл `/tmp/ansible_version_output.txt`. В качестве тестовой среды используется контейнер, запускаемый с помощью Podman. Подробное описание команд и возможных аргументов утилиты см. в *справочнике*.

Создание структуры роли

Для создания структуры роли выполните следующие действия:

1. Создайте каталог для роли и перейдите в него:

   ```
   mkdir ansible_version_role && cd ansible_version_role
   ```

2. Создайте сценарий:

   ```
   molecule init scenario
   ```

   Эта команда создаст структуру файлов и каталогов для сценария с поддержкой драйвера по умолчанию (`default`) в каталоге текущей роли.

3. Создайте каталог `tasks` для хранения задач роли:

   ```
   mkdir tasks
   ```

4. В каталоге `tasks` создайте файл `main.yml`.

В результате выполнения действий получится следующая структура:

```
ansible_version_role/
├── molecule
│   └── default
│       ├── converge.yml
│       ├── create.yml
│       ├── destroy.yml
│       ├── molecule.yml
│       ├── requirements.yml
│       └── verify.yml
└── tasks
    └── main.yml
```

Заполнение файлов роли

`molecule.yml` – файл настроек утилиты. Пример заполнения файла:

```
---
dependency:
  name: galaxy
  options:
    requirements-file: requirements.yml

platforms:
  - name: molecule-astra
    image: hub.astra-automation.ru/aa-1.2/aa-full-ee
    privileged: true
    groups:
      - all

driver:
  options:
    managed: false
    login_cmd_template: "podman exec -ti {instance} bash"
    ansible_connection_options:
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
      ansible_connection: podman

provisioner:
  name: ansible
  config_options:
    defaults:
      remote_tmp: '/tmp/.ansible-${USER}/tmp'
  playbooks:
    create: create.yml
    converge: converge.yml
    destroy: destroy.yml
    verify: verify.yml
  inventory:
    group_vars:
      all:
        ansible_python_interpreter: /usr/bin/python3.9  # Версия Python контейнера
        ansible_user: '1000'  # Пользователь в контейнере
```

Файл `converge.yml` используется для запуска роли, которая выполняет задачу получения версии Ansible. Пример заполнения файла:

```
- name: Converge
  hosts: all
  become: false
  gather_facts: false
  roles:
    - ansible_version_role
```

Файл `create.yml` отвечает за создание контейнера. В этом примере для создания контейнера используется модуль `containers.podman.podman_container`:

```
- name: Create
  hosts: localhost
  gather_facts: false

  tasks:
    - name: Create a container
      containers.podman.podman_container:
        name: "{{ item.name }}"
        image: "{{ item.image }}"
        privileged: "{{ item.privileged | default(omit) }}"
        volumes: "{{ item.volumes | default(omit) }}"
        capabilities: "{{ item.capabilities | default(omit) }}"
        systemd: "{{ item.systemd | default(omit) }}"
        state: started
        command: "{{ item.command | default('sleep 1d') }}"
        log_driver: json-file
      register: result
      loop: "{{ molecule_yml.platforms }}"

    - name: Create remote_tmp directory
      ansible.builtin.file:
        path: '/tmp/.ansible-{{ ansible_user }}/tmp'
        state: directory
        mode: '0755'
```

Файл `destroy.yml` отвечает за удаление контейнера по окончании тестов. Пример заполнения:

```
- name: Destroy molecule containers
  hosts: localhost
  gather_facts: false
  tasks:
    - name: Stop and remove container
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
    containers.podman.podman_container:
      name: molecule-astra
      state: absent
      rm: true
```

В файле `verify.yml` содержится задача проверки существования файла с версией Ansible в тестовом окружении. Пример заполнения:

```
- name: Verify Ansible version output
  hosts: all
  tasks:
    - name: Check if ansible version output contains 'ansible'
      ansible.builtin.command: "cat /tmp/ansible_version_output.txt"
      register: ansible_version_check
      failed_when: "'ansible' not in ansible_version_check.stdout"
      changed_when: false
```

В файле `requirements.yml` указываются зависимости Ansible. В данном примере файл указывает на необходимость использования коллекции `containers.podman`:

```
collections:
  - containers.podman
```

В файле `main.yml` указываются задачи, которые выполняет роль. В данном примере роль выполняет команду `ansible --version` и сохраняет результат в файл `/tmp/ansible_version_output.txt`.

```
- name: Get Ansible version
  ansible.builtin.command: ansible --version
  register: ansible_version_output
  changed_when: false

- name: Save Ansible version output to file in /tmp directory
  ansible.builtin.copy:
    content: "{{ ansible_version_output.stdout }}"
    dest: /tmp/ansible_version_output.txt
```

Запуск тестов

Чтобы запустить полный цикл тестирования, выполните команду:

```
molecule test
```

Запуск этой команды приведет к выполнению следующих этапов тестирования:

1. `create`;
2. `converge`;
3. `verify`;
4. `destroy`.

Для подключения к контейнеру и ручной проверки выполните команду:

```
molecule login
```

Tox Ansible

Tox Ansible – это расширение для Tox[240], позволяющее использовать его для тестирования коллекций Ansible с различными версиями Python и Ansible без использования контейнеров, виртуальных машин и тестовых стендов.

[240] https://tox.wiki/en/latest/index.html

Ключевые особенности и достоинства

Tox Ansible имеет следующие ключевые особенности и достоинства:

- Поддержка мультиверсионного тестирования.

  Tox Ansible позволяет тестировать коллекции сразу на нескольких версиях Python и Ansible.

- Интеграция с инструментами тестирования Python.

  Tox Ansible позволяет проводить следующие типы тестов:

  - `sanity` – проверка качества кода и соответствия стандартам;
  - `unit` – модульное тестирование;
  - `integration` – интеграционное тестирование.

  Для проведения тестирования используются следующие инструменты:

  - `tox` – управляет тестовыми средами;
  - `ansible-test` – проводит тесты типа `sanity`;
  - `pytest` – выполняет тесты типов `unit` и `integration`.

- Локальное тестирование.

  Тестовые окружения сохраняются после завершения тестирования. Это сокращает время на развертывание окружений.

- Простота управления тестовыми окружениями.

  Все виртуальные окружения создаются внутри каталога `.tox/`. Если нужно, эти окружения можно легко удалить или пересоздать без дополнительных настроек.

Установка

Для установки Tox Ansible выполните следующие действия:

1. Подключите репозиторий Astra Automation.

   #### Инструкция по подключению репозитория

   1. В каталоге `/etc/apt/sources.list.d/` создайте файл `astra-automation.list` со ссылкой на репозиторий Astra Automation:

      ```
      deb https://dl.astralinux.ru/aa/aa-debs-for-alse-1.8 <version> main
      ```

      Вместо <version> необходимо подставить версию устанавливаемой платформы, например, `2.0`.

      Доступные версии продукта опубликованы в таблице *История обновлений*.

   2. Обновите список доступных пакетов:

      ```
      sudo apt update
      ```

2. Выполните команду:

   ```
   sudo apt-get install tox-ansible --yes
   ```

Установка при отсутствии доступа к интернету описана в документе *Ресурсы разработчика*.

Принцип работы

Работа с Tox Ansible начинается с создания в корневом каталоге проекта файла настроек `tox-ansible.ini`. В этом файле указываются следующие параметры:

- тестовые окружения;
- зависимости;
- тесты, которые должны быть выполнены в каждом виртуальном окружении.

Если не требуется переопределение настроек, файл может быть пустым, но его присутствие обязательно.

Параметры запуска Tox Ansible также можно задать с помощью аргументов командной строки. Настройки из аргументов командной строки имеют более высокий приоритет, чем указанные в файле `tox-ansible.ini`.

Параметры в файле настроек задаются как в следующем примере:

```
[tox]
envlist =
    unit-py37-ansible2.9,
    unit-py37-ansible2.10,
    integration-py37-ansible2.9,
    integration-py37-ansible2.10

[testenv]
basepython =
    py37: python3.7
deps =
    ansible2.9: ansible==2.9.*
    ansible2.10: ansible==2.10.*
commands = pytest tests
```

Подробное описание параметров файла настроек см. в *справочнике*.

При запуске Tox Ansible выполняет следующие действия:

1. Определяет список тестовых окружений, указанных в файле настроек.
2. Для каждого тестового окружения создает изолированное виртуальное окружение с соответствующей версией Python.
3. Устанавливает зависимости, указанные для каждого тестового окружения.
4. Запускает тесты внутри каждого виртуального окружения.
5. Собирает и выводит результаты тестирования.

Примеры

Для изучения работы Tox Ansible на практике в вашей системе должен быть предварительно установлен Python версий 3.11. При отсутствии в системе необходимой версии Python тесты для нее будут пропущены.

Базовый сценарий

Типичный сценарий применения Tox Ansible содержит следующие шаги:

1. Создайте коллекцию `my_collection`:

   ```
   ansible-galaxy collection init my_namespace.my_collection
   ```

2. Перейдите в каталог `roles/`:

   ```
   cd my_namespace/my_collection/roles/
   ```

3. Создайте роль `test_role`:

   ```
   ansible-galaxy init test_role
   ```

4. Добавьте в файл `main.yml` следующие данные:

```
---
- name: Message Hello Role
  ansible.builtin.debug:
    msg: "Hello collection"
```

5. Перейдите в каталог `my_collection/`:

```
cd ..
```

6. В файле `/my_namespace/my_collectionmeta/runtime.yml` раскомментируйте строку:

```
requires_ansible: '>=2.9.10'
```

7. В каталоге `my_collection/` создайте сценарий `hello_collection.yml` со следующим содержимым:

```
---
- name: Test Role
  hosts: localhost
  roles:
    - test_role
```

8. Создайте файл `tests/test_sanity.py` с тестом типа `sanity`

Список 18: test_sanity.py

```
import ansible_runner


def test_hello_collection():
    result = ansible_runner.run(private_data_dir=".", playbook="hello_collection.yml
↪")

    assert result.rc == 0, f"Сценарий завершился с ошибкой: {result.rc}"

    # Проверка наличия "Hello collection" в stdout
    assert any(
        "Hello collection" in event.get("stdout", "") for event in result.events
    ), "Сообщение 'Hello collection' не найдено в выводе"
```

9. Создайте файл `tests/integration/test_integration.py` с тестом типа `integration`:

Список 19: test_integration.py

```
import subprocess
import unittest


class TestHelloCollectionIntegration(unittest.TestCase):
    def test_playbook_integration(self):
        """Проверка интеграции сценария и роли."""

        playbook_path = "<path_to_hello_collection.yml>"

        # Выполнение сценария через ansible-playbook
        result = subprocess.run(
            ["ansible-playbook", playbook_path],
            capture_output=True,
            text=True,
            check=True,
        )

        # Проверка кода завершения
        self.assertEqual(
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
            result.returncode, 0, f"Сценарий завершился с ошибкой: {result.stderr}"
        )

        # Проверка, что в выводе содержится ожидаемое сообщение
        self.assertIn(
            "Hello collection",
            result.stdout,
            "Сообщение 'Hello collection' не найдено в выводе.",
        )


if __name__ == "__main__":
    unittest.main()
```

Здесь <path_to_hello_collection.yml> – путь к файлу сценария `hello_collection.yml`, созданному ранее.

10. Создайте файл `tests/unit/test_unit.py` с тестом типа `unit`:

Список 20: test_unit.py

```
import unittest


def hello_message():
    return "Hello collection"


class TestHelloFunction(unittest.TestCase):
    def test_hello_message(self):
        """Проверка, что функция возвращает правильное сообщение."""

        self.assertEqual(hello_message(), "Hello collection")


if __name__ == "__main__":
    unittest.main()
```

11. Создайте файл `tox-ansible.ini` со следующим содержимым:

```
[tox]
envlist = sanity-py3.11-ansible2.9, unit-py3.11-ansible2.9, integration-py3.11-
↪ansible2.9
skipsdist = true

[testenv:sanity-py3.11-ansible2.9]
basepython = python3.11
deps =
   ansible==2.9
   pytest
   ansible-runner
commands =
   pytest tests/test_sanity.py
allowlist_externals =
   ansible-playbook
   mkdir
   bash

[testenv:unit-py3.11-ansible2.9]
basepython = python3.11
deps =
   ansible==2.9
   pytest
   ansible-runner
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
commands =
   pytest tests/unit/test_unit.py
allowlist_externals =
   ansible-playbook
   mkdir
   bash

[testenv:integration-py3.11-ansible2.9]
basepython = python3.11
deps =
   ansible==2.9
   pytest
   ansible-runner
commands =
   pytest tests/integration/test_integration.py
allowlist_externals =
   ansible-playbook
   mkdir
   bash
```

12. Просмотрите список всех окружений, определенных в файле настроек:

```
tox list --conf tox-ansible.ini
```

Пример вывода

```
default environments:
sanity-py3.11-ansible2.9      -> [no description]
unit-py3.11-ansible2.9        -> [no description]
integration-py3.11-ansible2.9 -> [no description]
```

13. Запустите утилиту:

    - Для всех типов тестов:

      ```
      tox -r --conf tox-ansible.ini
      ```

    - С фильтром по типу теста:

      ```
      tox -f unit -r --conf tox-ansible.ini
      ```

    Дождитесь завершения выполнения тестирования, это может занять некоторое время. Тестирование считается успешным, если для всех тестовых сред указано значение OK.

Пример части вывода

```
sanity-py3.11-ansible2.9: OK (14.09=setup[12.32]+cmd[1.77] seconds)
unit-py3.11-ansible2.9: OK (10.98=setup[10.80]+cmd[0.18] seconds)
integration-py3.11-ansible2.9: OK (13.33=setup[11.71]+cmd[1.62] seconds)
congratulations :) (38.47 seconds)
```

Передача аргументов

Tox Ansible автоматически запускает `ansible-test` для тестов типа `sanity` или `pytest` для тестов типа `unit` и `integration`. Однако иногда нужно настроить запуск этих тестов, добавив дополнительные аргументы.

Чтобы передать аргументы для `ansible-test`, используйте следующую команду:

```
tox -f sanity --ansible --conf tox-ansible.ini -- --test validate-modules -vvv
```

Здесь:

- `-f` – фильтр тестов. В этом примере используется фильтр типа `sanity`.
- `--ansible` – аргумент, активирующий поддержку Ansible.
- `--` – разделитель. Все, что идет после разделителя, передается как аргументы для `ansible-test`.
- `--test` – название теста который необходимо запустить.
- `-vvv` – повышение детализации вывода.

Чтобы передать аргументы для `pytest`, используйте следующую команду:

```
tox -e unit-py3.11-2.14 --ansible --conf tox-ansible.ini -- --junit-xml=tests/output/
↪junit/unit.xml
```

Здесь:

- `-e` – окружение для тестов:
  - `unit` – тип тестов.
  - `py3.11` – версия Python.
  - `2.14` – версия Ansible.
- `--` – разделитель. Все, что идет после разделителя, передается как аргументы для `pytest`.
- `--junit-xml=tests/output/junit/unit.xml` – путь для сохранения отчета о тестировании в формате XML.

Ansible Pytest

Ansible Pytest – это расширение для системы тестирования pytest[241], обеспечивающее ее бесшовную интеграцию с Ansible.

Основные возможности

Основные возможности Ansible Pytest:

- Модульное тестирование коллекций.

  С помощью Ansible Pytest вы можете проверить поведение отдельных ролей или целых коллекций Ansible, чтобы убедиться в корректности работы каждого компонента.

- Интеграция с *Ansible Molecule*.

  Ansible Pytest может использовать сценарии Molecule в связке с `pytest`. Это позволяет тестировать роли и сценарии Ansible в различных средах, упрощая выявление и устранение ошибок.

- Интеграция Ansible с тестами Pytest.

  Ansible Pytest позволяет использовать возможности Ansible в наборах тестов `pytest`, например, вызывать предоставляемые Ansible модули для выполнения операций с тестовыми стендами, запускать специальные команды (ad-hoc) и тому подобное.

Установка

Для установки Ansible Pytest выполните следующие действия:

1. Подключите репозиторий Astra Automation.

[241] https://docs.pytest.org/en/stable/index.html

Инструкция по подключению репозитория

1. В каталоге `/etc/apt/sources.list.d/` создайте файл `astra-automation.list` со ссылкой на репозиторий Astra Automation:

   ```
   deb https://dl.astralinux.ru/aa/aa-debs-for-alse-1.8 <version> main
   ```

   Вместо <version> необходимо подставить версию устанавливаемой платформы, например, `2.0`.

   Доступные версии продукта опубликованы в таблице *История обновлений*.

2. Обновите список доступных пакетов:

   ```
   sudo apt update
   ```

2. Выполните команду:

   ```
   sudo apt-get install pytest-ansible --yes
   ```

Установка при отсутствии доступа к интернету описана в документе *Средства тестирования*.

Применение

Утилиту можно применять для выполнения следующих задач:

- Проверка тестового окружения на соответствие требованиям, необходимым для корректного применения ролей и выполнения сценариев.
  1. Любым удобным способом создайте тестовое окружение.
  2. Создайте тесты, собирающие данные о созданном окружении и проверяющие соответствие фактических характеристик окружения ожидаемым.
  3. Запустите тесты и убедитесь, что окружение развернуто именно в той конфигурации, которая нужна.
- Сравнение фактических изменений в тестовом окружении с ожидаемыми.
  1. Любым удобным способом создайте тестовое окружение.
  2. Запустите сценарии Ansible, чтобы внести изменения в тестовое окружение.
  3. Создайте тесты, проверяющие состояние тестового окружения после его изменения с помощью сценариев Ansible.
  4. Запустите тесты и убедитесь, что ваши сценарии вносят все необходимые изменения.
- Использование модулей Ansible в тестах.

  Ansible Pytest позволяет выполнять в тестах `pytest` следующие операции:

  - вызов модулей Ansible;
  - сбор фактов;
  - управление инвентарем.

Принцип работы

Набор тестов (test suite) `pytest` это файл с кодом на языке Python. Имя файла с набором тестов должно удовлетворять шаблонам:

- `test_*.py`;
- `*_test.py`.

При запуске pytest ищет файлы с подходящими именами в текущем каталоге и всех его подкаталогах. Более подробно порядок поиска файлов описан в документации Pytest[242].

В самом простом случае тест – это функция, название которой начинается с префикса test_, например:

```
def test_connection():
    # ...
```

Также тесты могут быть методами класса. В этом случае необходимо выполнение следующих условий:

- название класса должно начинаться с префикса Test;
- название метода должно начинаться с префикса test_.

```
class TestConnectionClass:
    def test_connection(self):
        # ...

    def test_nginx_service(self):
        # ...
```

В коде теста должно выполняться сравнение фактического значения параметра с ожидаемым с помощью инструкции assert, например:

```
def test_hostname(hostname):
    assert hostname == 'localhost'
```

Если условие не выполняется, тест считается непройденным.

Инвентарь

Инвентарь в Ansible Pytest – это инвентарь Ansible в формате INI или JSON, либо сценарий, возвращающий список узлов в формате JSON.

Инвентарь может быть задан следующими способами:

- указание пути к файлу с инвентарным списком в опции --inventory при запуске pytest:

  ```
  pytest --inventory <inventory> --ansible-host-pattern <pattern>
  ```

- использование вспомогательного кода ansible_adhoc.

  Пример настройки инвентаря с помощью этого вспомогательного кода см. *ниже*.

Вспомогательный код

В pytest вспомогательным кодом (fixtures) называются функции, упрощающие работу с тестовым окружением. Код вспомогательных функций запускается автоматически перед выполнением тестов и может быть использован повторно. Это позволяет сократить объем кода тестов и облегчает его сопровождение.

Элементы вспомогательного кода Ansible Pytest и их описания приводятся в *справочнике*.

Декоратор pytest.mark.ansible

Декоратор pytest.mark.ansible используется для фильтрации списка узлов, на которых должен быть выполнен тест. Это может быть полезно, если инвентарный список содержит большое количество узлов, но отдельные тесты нужно выполнить только на некоторых из них.

[242] https://docs.pytest.org/en/stable/explanation/goodpractices.html#conventions-for-python-test-discovery

Интеграция со сценариями Molecule

Ansible Pytest обнаруживает все файлы `molecule.yml` в коде проекта и запускает задания из них как тесты. Таким образом можно включать сценарии Molecule в набор тестов `pytest` и тестировать содержимое Ansible в различных сценариях и окружениях.

Интеграция с Molecule может быть выполнена следующими способами:

- Использование объекта `molecule_scenario`.

  В каталоге выбранной коллекции создайте файл `tests/integration/test_integration.py` со следующим содержимым:

```
"""Tests for Molecule scenarios."""

from __future__ import absolute_import, division, print_function

from pytest_ansible.molecule import MoleculeScenario


def test_integration(molecule_scenario: MoleculeScenario) -> None:
    """Run molecule for each scenario.

    :param molecule_scenario: The molecule scenario object
    """
    proc = molecule_scenario.test()
    assert proc.returncode == 0
```

  Вспомогательный код `molecule_scenario` предоставляет сценарии Molecule, обнаруженные в каталоге `extensions/molecule` и других каталогах коллекции. Для каждого обнаруженного сценария выполняется команда тестирования:

```
molecule test -s <scenario>
```

  Здесь <scenario> – путь к файлу сценария Molecule.

  `pytest` запускает каждый сценарий в отдельном процессе с помощью метода `test()`. Если код завершения процесса равен `0`, тест считается пройденным успешно. Коды завершения, отличные от `0`, считаются признаком ошибки.

- Включение тестов `pytest` в блок `verify` сценария Molecule.

  В этом случае тесты `pytest` проверяют корректность развертывания стендов, сравнивая их фактические параметры с ожидаемыми. Это делает процесс CI/CD более надежным.

Примеры

Для изучения возможностей Ansible Pytest создайте каталог, в котором будут размещаться файлы тестов. Команды запуска тестов выполняйте внутри этого каталога.

Выполнение команды `ping` на всех узлах

Этот тест выполняет специальную команду `ping` на всех узлах, указанных в инвентаре:

Список 21: `test_all_ping.py`

```
def test_all_pings(ansible_adhoc):
    # Call ping() command for all nodes in inventory
    result = ansible_adhoc().all.ping()

    for node, outcome in result.items():
        if outcome.is_successful:
            print(f"Node {node} ping successfull: {outcome['ping']}")
        else:
            print(f"Node {node} ping failed: {outcome}")
```

Результаты выполнения команды сохраняются в переменной `result`. Метод `items` возвращает два списка: названия узлов и результаты выполнения команды. В цикле выполняется обход списков. Если поле `is_successfull` содержит значение `True`, тест считается пройденным успешно и в терминал выводится соответствующее сообщение. В противном случае выводится сообщение об ошибке.

Время работы системы с момента последней загрузки

Команда `uptime` возвращает время, прошедшее с момента загрузки операционной системы. Приведенный ниже тест проверяет возможность выполнения команды `uptime` на узлах, указанных в файле `inventory-special.ini`.

При этом в коде теста настройки исходного инвентаря заменяются на указанные:

- подключение к узлам выполняется от имени пользователя `administrator`;
- используется повышение привилегий;
- для выполнения команд с повышенными привилегиями используется учетная запись пользователя `root`.

Список 22: `test_uptime.py`

```
def test_uptime_with_custom_inventory(ansible_adhoc):
    """Run uptime command with custom inventory."""
    nodes = ansible_adhoc(
        inventory="inventory-special.ini",
        user="administrator",
        become=True,
        become_user="root",
    )

    # Print current settings
    print("HostManager settings:")
    print(nodes.options)

    # Run adhoc command `uptime`
    result = nodes.all.command("uptime")

    print("\nUptime:")
    for node, outcome in result.items():
        if outcome.is_successful:
            # Print uptime
            print(f"Node {node} uptime successful:\n{outcome['stdout']}")
        else:
            # Print error message
            print(f"Node {node} uptime failed:\n{outcome}")

        assert outcome.is_successful, f"Uptime command failed on node {node}"
```

Переменная `result` присваивает результаты выполнения команды `uptime`.

Метод `items` возвращает два списка: названия узлов и результаты выполнения команды. В цикле выполняется обход списков. Если поле `outcome.is_successful` содержит значение `True`, тест считается пройденным успешно, и в терминал выводится количество времени, прошедшего с момента загрузки ОС тестового узла. В противном случае выводится сообщение об ошибке.

Указание инвентарного списка в аргументах CLI для этого теста не требуется:

```
pytest --host-pattern all
```

Различные способы обращения к узлам

Этот тест демонстрирует различные способы обращения к узлам:

Список 23: test_host_manager.py

```
def test_host_manager(ansible_adhoc):
    """Using HostManager for localhost and remote nodes."""

    nodes = ansible_adhoc()

    # Run ping command on all nodes
    nodes["all"].ping()

    # Run ping command only on localhost
    nodes["localhost"].ping()

    # Iterating nodes:
    for node in nodes.keys():
        result = nodes[node].ping()
        assert result[node].is_successful, f"Ping failed on node {node}"

        # Print successful ping
        print(f"Node {node} ping successful with result: {result[node]}")
```

Переменная nodes получает результаты выполнения функции ansible_adhoc(), подготавливающей окружение к использованию специальных команд.

Путем обращения к ключу all тест выполняет команду ping на всех узлах, указанных в инвентаре. Затем, путем обращения к ключу localhost`, тест выполняет команду ``ping только на локальном узле.

В цикле выполняется проход по списку узлов. Тест запускает команду ping для каждого очередного узла, а результат ее выполнения сохраняет в переменную result.

Тест проверяет значение поля is_successful. Если оно равно True, тест считается выполненным успешно.

Использование вспомогательного кода localhost

Этот тест проверяет выполнение команды ping на локальном узле:

Список 24: test_local_ping.py

```
def test_local_ping(localhost):
    """Test ping() command on localhost."""

    result = localhost.ping()

    for node, outcome in result.items():
        assert outcome.get("ping") == "pong", f"Ping for node {node} is failed"

        print(f"Node {node} ping successful with result: {outcome}")
```

В отличие от предыдущих тестов, для получения доступа к локальному узлу вместо вспомогательного кода ansible_adhoc используется вспомогательный код localhost. Он также позволяет выполнять специальные команды, но только на локальном узле.

Указание инвентарного списка в аргументах CLI в этом случае не требуется:

```
pytest --ansible-host-pattern all
```

Использование модулей Ansible

Этот тест демонстрирует использования модуля ansible.builtin.ping[243] с помощью вспомогательного кода ansible_module.

Список 25: test_ansible_module.py

```
def test_ansible_module(ansible_module):
    """Call ping() command from Ansible module."""
    result = ansible_module.ping()

    for node, outcome in result.items():
        assert outcome.get("ping") == "pong", f"Ping on {node} is failed"
        print(f"Ping successful on node {node} with result: {outcome}")
```

Переменная result получает результаты выполнения команды ping на всех узлах. Метод items возвращает два списка: названия узлов и результаты выполнения команды. В цикле выполняется обход списков. Если результат выполнения команды ping равен pong, тест считается пройденным успешно. В противном случае выводится сообщение об ошибке.

Обработка фактов Ansible

Этот тест собирает и выводит в терминал следующие сведения об узлах, на которых выполняется тестирование:

Факт Ansible	Описание
ansible_os_family	Семейство ОС
ansible_distribution	Дистрибутив ОС
ansible_distribution_version	Версия дистрибутива ОС
ansible_hostname	Доменное имя узла
ansible_architecture	Архитектура
ansible_processor_count	Количество CPU
ansible_memtotal_mb	Объем установленной оперативной памяти, МБ
ansible_default_ipv4	Основной адрес IPv4
ansible_default_ipv6	Основной адрес IPv6
ansible_uptime_seconds	Время в секундах с момента загрузки ОС

Список 26: test_print_top_10_facts.py

```
def test_print_top_10_facts(ansible_facts):
    """Show 10 Ansible facts."""

    important_facts = [
        "ansible_os_family",
        "ansible_distribution",
        "ansible_distribution_version",
        "ansible_hostname",
        "ansible_architecture",
        "ansible_processor_count",
        "ansible_memtotal_mb",
        "ansible_default_ipv4",
        "ansible_default_ipv6",
        "ansible_uptime_seconds",
    ]

    for node, facts in ansible_facts.items():
        print(f"\nImportant facts for node {node}:")

        for fact in important_facts:
```

(продолжается на следующей странице)

[243] https://docs.ansible.com/ansible/latest/collections/ansible/builtin/ping_module.html

(продолжение с предыдущей страницы)

```
            value = facts["ansible_facts"].get(fact, "Unknown")
            print(f"{fact}: {value}")
```

Метод `items` возвращает два списка: названия узлов и факты Ansible о них. В цикле выполняется обход этих списков. Из списка фактов извлекаются только факты, перечисленные в списке `important_facts`. Переменной `value` присваивается значение факта, если оно известно, и `Unknown` в противном случае.

Использование декоратора `pytest.mark.ansible`

Этот тест демонстрирует применение декоратора `pytest.mark.ansible` для замены списка узлов из инвентарного списка `inventory.ini` на `localhost`:

Список 27: `test_pytest_mark_ansible.py`

```
import pytest


@pytest.mark.ansible(host_pattern="localhost", connection="local")
def test_local_ping(ansible_module):
    """Replace nodes from inventory to localhost."""

    result = ansible_module.ping()  # Only on localhost

    assert "localhost" in result, "Ping command on localhost failed"
    assert result["localhost"]["ping"] == "pong", "Node localhost returned not a pong"
```

Значения, указанные в опции CLI `--ansible-host-pattern`, игнорируются. Вместо этого фильтру узлов присваивается значение `localhost`, а типу подключения `connection` значение `local`.

Для запуска теста используйте команду:

```
pytest --inventory inventory.ini --ansible-host-pattern all
```

Проверка результатов выполнения команд

Этот тест демонстрирует использование методов класса `AdHocResult` для проверки результатов выполнения команд:

Список 28: `test_adhoc_result.py`

```
def test_adhoc_result(ansible_adhoc):
    """Run command date() and check results."""

    contacted = ansible_adhoc().all.command("date")

    # Walking over all nodes:
    for node, result in contacted.items():
        assert result.is_successful, f"Command is failed on node {node}"

    # Walking over result values
    for result in contacted.values():
        assert (
            result.is_successful
        ), f"Command is failed on one of the nodes with result {result}"

    # Check node name is 'localhost' or 'vm'
    for node in contacted.keys():
        assert node in ["localhost", "vm"], f"Unexpected {node} in results"

    # Check results on localhost only
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
    assert contacted.localhost.is_successful, "Command is failed on localhost"

    # Check nodes list length
    assert len(contacted) > 0, "Result list is empty"

    # Check localhost in result list
    assert "localhost" in contacted, "Host 'localhost' not found in results"

    # Check successful result on localhost with __getattr__
    assert (
        contacted.localhost.is_successful
    ), "Command is failed on localhost over __getattr__"

    # Check successful result on localhost with __getitem__
    assert contacted[
        "localhost"
    ].is_successful, "Command is failed on localhost over __getitem__"
```

Проверка параметров окружения

Этот тест проверяет соответствие фактических параметров узлов ожидаемым:

Список 29: test_adhoc_result.py

```
def test_environment(ansible_facts):
    """Validation of environment characteristic."""

    # Expected values
    expected_cores = 2
    expected_memory_mb = 1014 * 8  # 8 GB
    expected_disk_size_gb = 20

    # Check nodes parameters
    for node, facts in ansible_facts.items():
        # Check CPU cores count
        actual_cores = facts["ansible_facts"].get("ansible_processor_cores", 0)
        assert (
            actual_cores == expected_cores
        ), f"Expected {expected_cores} cores, got {actual_cores} for node {node}"

        # Check RAM size
        actual_memory_mb = facts["ansible_facts"].get("ansible_memtotal_mb", 0)
        assert (
            actual_memory_mb >= expected_memory_mb
        ), f"Expected {expected_memory_mb} RAM or greather, got {actual_memory_mb} for␣
↪node {node}"

        # Check storage size
        actual_disk_size_str = (
            facts["ansible_facts"]
            .get("ansible_devices", {})
            .get("vda", {})
            .get("size", "0.00 GB")
        )
        actual_disk_size_gb = float(
            actual_disk_size_str.split()[0]  # Convert string onto number
        )
        assert (
            actual_disk_size_gb >= expected_disk_size_gb
        ), f"Expected {expected_disk_size_gb} GB or greather storage size, got {actual_
↪disk_size_gb} for node {node}"
```

Ожидаемые значения параметров указаны в соответствующих переменных:

- expected_cores – количество ядер CPU;
- expected_memory_mb – количество установленной RAM в МБ;
- expected_disk_size_gb – объем хранилища в ГБ.

Сведения об узлах собираются с помощью вспомогательного кода ansible_facts. Метод items возвращает два списка: названия узлов и словари с фактами. В цикле выполняется обход обоих списков. Фактические значения ожидаемых фактов сравниваются с ожидаемыми. Если какой-то факт не удовлетворяет условию, тест считается не пройденным, и в терминал выводится сообщение об ошибке.

Проверка состояния службы

Этот тест проверяет состояние веб-сервера NGINX:

- пакет nginx должен быть установлен;
- служба nginx должна быть включена и запущена.

Список 30: test_service_status.py

```
def test_nginx_installed_and_running(ansible_adhoc):
    """Ensure NGINX is installed and runned."""

    nodes = ansible_adhoc()

    # Check nginx package status
    result_package = nodes.all.package(name="nginx", state="present")
    for node, result in result_package.items():
        assert result.is_successful, f"nginx package is not installed on node {node}"

    # Check nginx.service status
    result_service = nodes.all.service(name="nginx", state="started", enabled=True)

    for node, result in result_service.items():
        assert (
            result.is_succesful
        ), f"nginx.service is not runned or disabled on node {node}"
```

С помощью вспомогательного кода ansible_adhoc вызываются модули Ansible package и service.

Модуль package проверяет наличие в системе пакета nginx. Поле is_successful содержит значение True только если пакет nginx установлен (state = "present").

Модуль service проверяет состояние службы nginx. Поле is_successful содержит значение True, только если служба nginx запущена (state="started") и запускается при загрузке ОС (enabled=True).

Ansible Lint

Ansible Lint – это утилита командной строки, предназначенная для проверки качества и стиля кода в сценариях, ролях и коллекциях Ansible. Она помогает следовать лучшим практикам при написании кода, выявлять распространенные ошибки и потенциальные проблемы, а также поддерживать чистоту и читаемость кода.

Основные возможности Ansible Lint

Основные возможности Ansible Lint:

- Проверка на соответствие стандартам.

  Ansible Lint содержит правила написания кода из лучших практик, что позволяет избежать ошибок, ухудшающих качество и поддерживаемость кода.

- Автоматическое выявление ошибок.

Утилита анализирует контент Ansible, указывая на места, которые могут быть потенциально проблемными или нестандартными.

- Адаптация к новым версиям Ansible.

  Ansible Lint помогает адаптировать код к новым версиям Ansible, что упрощает его обновление и предотвращает возможные несовместимости.

Установка

Для установки Ansible Lint выполните следующие действия:

1. Подключите репозиторий Astra Automation.

   #### Инструкция по подключению репозитория

   1. В каталоге `/etc/apt/sources.list.d/` создайте файл `astra-automation.list` со ссылкой на репозиторий Astra Automation:

      ```
      deb https://dl.astralinux.ru/aa/aa-debs-for-alse-1.8 <version> main
      ```

      Вместо <version> необходимо подставить версию устанавливаемой платформы, например, `2.0`.

      Доступные версии продукта опубликованы в таблице *История обновлений*.

   2. Обновите список доступных пакетов:

      ```
      sudo apt update
      ```

2. Выполните команду:

   ```
   sudo apt-get install ansible-lint --yes
   ```

Установка при отсутствии доступа к интернету описана в документе *Средства тестирования*.

Настройка Ansible Lint

Ansible Lint предоставляет различные возможности для настройки в проектах:

- игнорирование определенных правил;
- включение дополнительных проверок;
- настройка профилей под конкретные нужды.

Ansible Lint можно настроить с помощью следующих компонентов:

- *аргументы команды*;
- *переменные окружения*;
- *файл настроек*.

При вычислении действующих значений аргументов используется следующий порядок (значения, указанные на более поздних этапах, заменяют значения, указанные ранее):

1. Переменные окружения.
2. Файл настроек.
3. Аргументы команды.

По умолчанию Ansible Lint начинает поиск файла настроек с текущего каталога. Возможные названия и расположения файлов:

- `.ansible-lint`;
- `.config/ansible-lint.yml`;
- `.config/ansible-lint.yaml`.

Если файл настроек не найден в текущем каталоге, Ansible Lint попытается найти его в родительских каталогах. Если используется Git, то поиск осуществляется только в пределах текущего репозитория. Путь к файлу настроек можно указать явно с помощью аргумента `-c`, например:

```
ansible-lint -c ansible-lint-config.yml
```

Настройки в конфигурационном файле задаются следующим образом:

Пример файла настроек

Список 31: .ansible-lint

```
---
# .ansible-lint

profile: null # min, basic, moderate, safety, shared, production

# Allows dumping of results in SARIF format
# sarif_file: result.sarif

# exclude_paths included in this file are parsed relative to this file's location
# and not relative to the CWD of execution. CLI arguments passed to the --exclude
# option are parsed relative to the CWD of execution.
exclude_paths:
  - .cache/ # implicit unless exclude_paths is defined in config
  - test/fixtures/formatting-before/
  - test/fixtures/formatting-prettier/
# parseable: true
# quiet: true
# strict: true
# verbosity: 1

# Mock modules or roles in order to pass ansible-playbook --syntax-check
mock_modules:
  - zuul_return
  # note the foo.bar is invalid as being neither a module or a collection
  - fake_namespace.fake_collection.fake_module
  - fake_namespace.fake_collection.fake_module.fake_submodule
mock_roles:
  - mocked_role
  - author.role_name # old standalone galaxy role
  - fake_namespace.fake_collection.fake_role # role within a collection

# Enable checking of loop variable prefixes in roles
loop_var_prefix: "^(__|{role}_)"

# Enforce variable names to follow pattern below, in addition to Ansible own
# requirements, like avoiding python identifiers. To disable add `var-naming`
# to skip_list.
var_naming_pattern: "^[a-z_][a-z0-9_]*$"

use_default_rules: true
# Load custom rules from this specific folder
# rulesdir:
#   - ./rule/directory/

# Ansible-lint is able to recognize and load skip rules stored inside
# `.ansible-lint-ignore` (or `.config/ansible-lint-ignore.txt`) files.
# To skip a rule just enter filename and tag, like "playbook.yml package-latest"
# on a new line.
# Optionally you can add comments after the tag, prefixed by "#". We discourage
# the use of skip_list below because that will hide violations from the output.
# When putting ignores inside the ignore file, they are marked as ignored, but
# still visible, making it easier to address later.
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
skip_list:
  - skip_this_tag

# Ansible-lint does not automatically load rules that have the 'opt-in' tag.
# You must enable opt-in rules by listing each rule 'id' below.
enable_list:
  - args
  - empty-string-compare # opt-in
  - no-log-password # opt-in
  - no-same-owner # opt-in
  - name[prefix] # opt-in
  - galaxy-version-incorrect # opt-in
  # add yaml here if you want to avoid ignoring yaml checks when yamllint
  # library is missing. Normally its absence just skips using that rule.
  - yaml
# Report only a subset of tags and fully ignore any others
# tags:
#   - jinja[spacing]

# Ansible-lint does not fail on warnings from the rules or tags listed below
warn_list:
  - skip_this_tag
  - experimental # experimental is included in the implicit list
# - role-name
# - yaml[document-start]  # you can also use sub-rule matches

# Some rules can transform files to fix (or make it easier to fix) identified
# errors. `ansible-lint --fix` will reformat YAML files and run these transforms.
# By default it will run all transforms (effectively `write_list: ["all"]`).
# You can disable running transforms by setting `write_list: ["none"]`.
# Or only enable a subset of rule transforms by listing rules/tags here.
# write_list:
#   - all

# Offline mode disables installation of requirements.yml and schema refreshing
offline: true

# Define required Ansible's variables to satisfy syntax check
extra_vars:
foo: bar
multiline_string_variable: |
  line1
  line2
complex_variable: ":{;\t$()"
# Uncomment to enforce action validation with tasks, usually is not
# needed as Ansible syntax check also covers it.
# skip_action_validation: false

# List of additional kind:pattern to be added at the top of the default
# match list, first match determines the file kind.
kinds:
  # - playbook: "**/examples/*.{yml,yaml}"
  # - galaxy: "**/folder/galaxy.yml"
  # - tasks: "**/tasks/*.yml"
  # - vars: "**/vars/*.yml"
  # - meta: "**/meta/main.yml"
  - yaml: "**/*.yaml-too"

# List of additional collections to allow in only-builtins rule.
# only_builtins_allow_collections:
#   - example_ns.example_collection

# List of additions modules to allow in only-builtins rule.
# only_builtins_allow_modules:
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
#   - example_module

# Allow setting custom prefix for name[prefix] rule
task_name_prefix: "{stem} | "
# Complexity related settings

# Limit the depth of the nested blocks:
# max_block_depth: 20

# Also recognize these versions of Ansible as supported:
# supported_ansible_also:
#   - "2.14"
```

Подробное описание параметров см. в *справочнике*.

Игнорирование правил для определенных файлов

Чтобы при проверке файлов Ansible Lint не использовал определенные правила, создайте файл `.ansible-lint-ignore` или `.config/ansible-lint-ignore.txt`. Созданный файл должен быть размещен в том же каталоге, что и файл основных настроек Ansible Lint. Каждая строка содержит путь к файлу и через пробел идентификатор правила.

Пример заполнения:

```
# Игнорирование правила package-latest для файла playbook.yml
playbook.yml package-latest

# Игнорирование правила deprecated-module для playbook.yml
playbook.yml deprecated-module

# Игнорирование правила no-tabs для роли my_role
roles/my_role/tasks/main.yml no-tabs
```

Создание правил

Ansible Lint позволяет создавать правила, содержащие собственные уникальные проверки, для поддержки стиля кодирования, принятого в вашей организации. Каждое правило необходимо создавать в отдельном файле Python и описывать в виде класса, который наследует `AnsibleLintRule`. Файл с правилом следует именовать согласно цели правила, например, `DeprecatedVariableRule.py`. Внутри каждого класса должны быть следующие поля:

- `id` – уникальный идентификатор правила;
- `description` – краткое описание правила (какой тип ошибок оно выявляет);
- `tags` – список тегов для группировки правил, близких по смыслу.

Для проверки кода по заданному правилу используют следующие методы:

- `match`;
- `matchtask`.

Метод `match` работает с каждой строкой и возвращает результат проверки. Если строка не соответствует критерию, метод возвращает `None` или `False`. Если строка соответствует критерию, метод возвращает `True` или настраиваемое сообщение.

Пример правила, реализующего метод `match`:

```
class DeprecatedVariableRule(AnsibleLintRule):
    """Deprecated variable declarations."""
    id = 'DEPRECATED_VAR'
    description = 'Check for deprecated variable syntax ${var}'
    tags = ['deprecations']
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
    def match(self, line: str) -> Union[bool, str]:
        return '${' in line
```

Метод `matchtask` обрабатывает каждую задачу или обработчик (handler). Все задачи стандартизируются, чтобы включать ключи `module` и `module_arguments`. Если в задаче присутствуют модификаторы `when`, `with_items`, `tags` и другие, они также доступны для проверки.

Пример правила, реализующего метод `matchtask`:

```
from ansiblelint.rules import AnsibleLintRule
from typing import Union

class TaskHasTag(AnsibleLintRule):
    """Ensure every task has a tag for better organization."""
    id = 'TASK_TAG'
    description = 'Tasks must have a tag to improve organization and filtering'
    tags = ['department', 'team']

    def matchtask(self, task, file=None) -> Union[bool, str]:
        if 'tags' not in task:
           return "Task missing a tag"
        return False
```

Примечание

Для создания более сложных правил рекомендуется изучить методы и свойства класса `AnsibleLintRule` и его родительских классов.

Пример

Для изучения Ansible Lint на практике, сделайте клон репозитория, содержащего примеры сценариев:

```
git clone https://github.com/ansible/ansible-lint.git
```

Проверка сценария

Для проверки сценария `example.yml` выполните команду:

```
ansible-lint --offline -p ansible-lint/examples/playbooks/example.yml
```

Пример вывода:

```
ansible-lint/examples/playbooks/example.yml:1: schema[playbook]: $[0].tasks[13] None is␣
↪not of type 'object'
ansible-lint/examples/playbooks/example.yml:9: jinja[spacing]: Jinja2 spacing could be␣
↪improved: echo {{this_variable}} is not set in this playbook -> echo {{ this_variable }
↪} is not set in this playbook (warning)
ansible-lint/examples/playbooks/example.yml:9: no-changed-when: Commands should not␣
↪change things if nothing needs doing.
ansible-lint/examples/playbooks/example.yml:12: no-changed-when: Commands should not␣
↪change things if nothing needs doing.
ansible-lint/examples/playbooks/example.yml:13: yaml[trailing-spaces]: Trailing spaces
ansible-lint/examples/playbooks/example.yml:15: args[module]: missing required␣
↪arguments: repo (warning)
ansible-lint/examples/playbooks/example.yml:15: latest[git]: Result of the command may␣
↪vary on subsequent runs.
ansible-lint/examples/playbooks/example.yml:18: args[module]: missing required␣
↪arguments: repo (warning)
ansible-lint/examples/playbooks/example.yml:18: latest[git]: Result of the command may␣
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
↪vary on subsequent runs.
ansible-lint/examples/playbooks/example.yml:21: args[module]: Unsupported parameters for␣
↪(basic.py) module: bobbins. Supported parameters include: accept_hostkey, accept_
↪newhostkey, archive, archive_prefix, bare, clone, depth, dest, executable, force, gpg_
↪whitelist, key_file, recursive, reference, refspec, remote, repo, separate_git_dir,␣
↪single_branch, ssh_opts, track_submodules, umask, update, verify_commit, version␣
↪(name). (warning)
ansible-lint/examples/playbooks/example.yml:21: no-free-form: Avoid using free-form when␣
↪calling module actions. (ansible.builtin.git)
ansible-lint/examples/playbooks/example.yml:24: command-instead-of-module: git used in␣
↪place of git module
ansible-lint/examples/playbooks/example.yml:24: no-changed-when: Commands should not␣
↪change things if nothing needs doing.
ansible-lint/examples/playbooks/example.yml:27: command-instead-of-module: git used in␣
↪place of git module
ansible-lint/examples/playbooks/example.yml:30: latest[git]: Result of the command may␣
↪vary on subsequent runs.
ansible-lint/examples/playbooks/example.yml:33: jinja[spacing]: Jinja2 spacing could be␣
↪improved: {{item}} -> {{ item }} (warning)
ansible-lint/examples/playbooks/example.yml:36: yaml[indentation]: Wrong indentation:␣
↪expected 8 but found 6
ansible-lint/examples/playbooks/example.yml:39: no-free-form: Avoid using free-form when␣
↪calling module actions. (ansible.builtin.dnf)
ansible-lint/examples/playbooks/example.yml:39: package-latest: Package installs should␣
↪not use latest.
ansible-lint/examples/playbooks/example.yml:42: name[missing]: All tasks should be named.
ansible-lint/examples/playbooks/example.yml:42: no-free-form: Avoid using free-form when␣
↪calling module actions. (ansible.builtin.debug)
ansible-lint/examples/playbooks/example.yml:44: no-free-form: Avoid using free-form when␣
↪calling module actions. (ansible.builtin.apt)
ansible-lint/examples/playbooks/example.yml:44: package-latest: Package installs should␣
↪not use latest.
ansible-lint/examples/playbooks/example.yml:47: name[missing]: All tasks should be named.
Read documentation for instructions on how to ignore specific rule violations.

                      Rule Violation Summary
count tag                       profile rule associated tags
    2 command-instead-of-module basic   command-shell, idiom
    2 jinja[spacing]            basic   formatting (warning)
    4 no-free-form              basic   syntax, risk
    1 schema[playbook]          basic   core
    2 name[missing]             basic   idiom
    1 yaml[indentation]         basic   formatting, yaml
    1 yaml[trailing-spaces]     basic   formatting, yaml
    3 latest[git]               safety  idempotency
    2 package-latest            safety  idempotency
    3 no-changed-when           shared  command-shell, idempotency
    3 args[module]                      syntax, experimental (warning)

Failed: 19 failure(s), 5 warning(s) on 1 files. Last profile that met the validation␣
↪criteria was 'min'.
```

Пример создания правила

Чтобы создать правила на основе методов `match` и `matchtask` и проверить с помощью этих правил сценарий `example.com`, выполните следующие действия:

1. Создайте каталог для правил и перейдите в него, например:

   ```
   mkdir my_custom_rules && cd my_custom_rules
   ```

2. Создайте файл `TaskHasTag.py`, содержащий правило для проверки наличия тега:

```
from __future__ import annotations
from typing import TYPE_CHECKING, Union

from ansiblelint.rules import AnsibleLintRule

if TYPE_CHECKING:
    from ansiblelint.file_utils import Lintable
    from ansiblelint.utils import Task


class TaskHasTag(AnsibleLintRule):
    """Tasks must have tag."""

    id = 'EXAMPLE001'
    description = 'Tasks must have tag'
    tags = ['custom', 'core']

    def matchtask(self,
                  task: Task,
                  file: Lintable | None = None
                  ) -> Union[bool, str]:
        # Эта проверка исключает задачи include и fail из требований наличия тегов,
        # так как для них теги не обязательны:
        # Если задача включает другие задачи (include) или намеренно завершает␣
↪выполнение (fail),
        # она исключается из проверки на наличие тегов, возвращая False.
        if not set(task.keys()).isdisjoint(['include', 'fail']):
            return False

        if not task.get("tags"):
            return True
        return False
```

3. Создайте файл TestRule.py, содержащий правило для проверки наличия определенного слова:

```
from typing import Union
from ansiblelint.rules import AnsibleLintRule


class TestRule(AnsibleLintRule):
    """Deprecated variable declarations."""

    id = 'EXAMPLE002'
    description = 'This rule triggers on tasks containing the word "task".'
    tags = ['custom', 'core']

    def match(self, line: str) -> Union[bool, str]:
        # Срабатывает только для строк, содержащих "task" — для тестирования
        return 'task' in line
```

4. Запустите проверку сценария example.com:

```
ansible-lint \
  --offline \
  -r my_custom_rules/ \
  -t custom ~/ansible-lint/examples/playbooks/example.yml
```

Пример вывода:

```
EXAMPLE002: Deprecated variable declarations.
ansible-lint/examples/playbooks/example.yml:8   tasks:

EXAMPLE001: Tasks must have tag.
ansible-lint/examples/playbooks/example.yml:9 Task/Handler: Unset variable
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
EXAMPLE001: Tasks must have tag.
ansible-lint/examples/playbooks/example.yml:12 Task/Handler: Trailing whitespace

EXAMPLE001: Tasks must have tag.
ansible-lint/examples/playbooks/example.yml:15 Task/Handler: Run git check

EXAMPLE001: Tasks must have tag.
ansible-lint/examples/playbooks/example.yml:18 Task/Handler: Run git check 2

EXAMPLE001: Tasks must have tag.
ansible-lint/examples/playbooks/example.yml:21 Task/Handler: Run git check 3

EXAMPLE001: Tasks must have tag.
ansible-lint/examples/playbooks/example.yml:24 Task/Handler: Executing git through␣
↪command

EXAMPLE001: Tasks must have tag.
ansible-lint/examples/playbooks/example.yml:27 Task/Handler: Executing git through␣
↪command

EXAMPLE001: Tasks must have tag.
ansible-lint/examples/playbooks/example.yml:30 Task/Handler: Using git module

EXAMPLE001: Tasks must have tag.
ansible-lint/examples/playbooks/example.yml:33 Task/Handler: Passing git as an␣
↪argument to another task

EXAMPLE002: Deprecated variable declarations.
ansible-lint/examples/playbooks/example.yml:33     - name: Passing git as an␣
↪argument to another task

EXAMPLE001: Tasks must have tag.
ansible-lint/examples/playbooks/example.yml:39 Task/Handler: Dnf latest

EXAMPLE001: Tasks must have tag.
ansible-lint/examples/playbooks/example.yml:42 Task/Handler: debug msg=debug task␣
↪without a name

EXAMPLE002: Deprecated variable declarations.
ansible-lint/examples/playbooks/example.yml:42     - ansible.builtin.debug: msg=
↪"debug task without a name"

EXAMPLE001: Tasks must have tag.
ansible-lint/examples/playbooks/example.yml:44 Task/Handler: Run apt latest

EXAMPLE001: Tasks must have tag.
ansible-lint/examples/playbooks/example.yml:47 Task/Handler: meta flush_handlers

Read documentation for instructions on how to ignore specific rule violations.

            Rule Violation Summary
count tag        profile rule associated tags
   13 EXAMPLE001         custom, core
   3 EXAMPLE002         custom, core

Failed: 16 failure(s), 0 warning(s) on 1 files. Last profile that met the␣
↪validation criteria was 'production'. Rating: 5/5 star
```

Ansible Test

Ansible Test – это утилита командной строки, предназначенная для проверки качества кода Ansible (ядра, модулей, расширений, коллекций). Она помогает разработчикам и контрибьюторам проверять код перед его отправкой в основной репозиторий проекта Ansible.

Основные возможности Ansible Test

Ansible Test позволяет проводить следующие типы тестов:

- `sanity` – проверка качества кода и соответствия стандартам;
- `unit` – модульное тестирование;
- `integration` – интеграционное тестирование.

Установка

Утилита `ansible-test` входит в состав Ansible Core. При его наличии установка дополнительных пакетов не требуется.

Совместимость с `pytest-ansible`

Команда `ansible-test unit` не работает, если в системе установлен пакет `pytest-ansible`. Удалите его, чтобы восстановить работоспособность команды:

```
sudo apt purge pytest-ansible --yes
```

17.5 Разработка коллекций

Использование Ansible Navigator и Ansible Creator для разработки контента.

Важно

Эта часть документации находится в стадии разработки.

17.6 Служебные контейнеры

Служебные контейнеры – это изолированные среды, запускаемые на основе заранее подготовленных образов. Эти образы содержат все необходимое для работы с платформой Astra Automation.

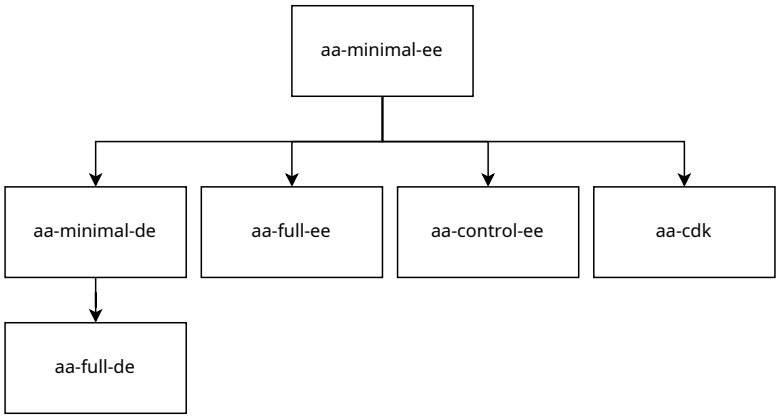
По умолчанию утилиты, Automation Controller и контроллер Event-Driven Automation используют *Podman* для управления контейнерами. Применение *Docker* в собственной инфраструктуре допускается, однако рекомендованным инструментом остается Podman. Большинство инструкций и примеров в документации приводятся именно для этого инструмента.

Рекомендуется использовать следующие образы, предоставляемые ПАО Группа Астра и доступные в Automation Hub[244]:

- *универсальный образ* – `aa-minimal-ee`;
- *образы среды исполнения* – `aa-control-ee` и `aa-full-ee`;
- *образы среды принятия решений* – `aa-minimal-de` и `aa-full-de`;
- *образ для разработчика* – `aa-cdk`.

[244] https://hub.astra-automation.ru/

Иерархия наследования образов представлена на схеме:



Примечание

С подробной информацией о составе рекомендуемых образов можно ознакомиться в *документации*.

Инструментом для работы с образом может быть *Podman* (рекомендуется) или Docker.

Предупреждение

Для запуска контейнера с указанным образом необходима версия Docker не ниже 19.03.

Использование образов дает следующие преимущества:

- Нет необходимости устанавливать дополнительное ПО: все необходимое уже есть в составе образа.
- Не требуется ручная настройка окружения.
- Контейнер с образом может быть запущен в любой ОС, поддерживаемой Podman.

Совет

Актуальный список версий образов доступен на Automation Hub[245].

17.6.1 Универсальный образ

Образ `aa-minimal-ee` является исходным образом для всех последующих и содержит минимальный набор утилит:

- предназначен для использования в качестве основы для разработки собственных образов среды исполнения;
- максимально оптимизирован по размеру и содержит только минимально необходимый набор коллекций Ansible и модулей Python.

17.6.2 Образы среды исполнения

Среда исполнения Astra Automation (Execution Environment, EE) – это контейнер с образом на базе Astra Linux Special Edition, который рекомендуется использовать для развертывания и тестирования кода, запускаемого с помощью Ansible.

В Automation Hub доступны следующие готовые образы среды исполнения:

- `aa-control-ee` – образ для плоскости управления Automation Controller:

[245] https://hub.astra-automation.ru/

  - предназначен для служебных задач;
  - позволяет формировать инвентарные списки на основе данных из сторонних систем.
- `aa-full-ee` – образ EE по умолчанию в Automation Controller и Ansible Navigator. Содержит все коллекции, доступные в Automation Hub, и позволяет использовать их без предварительной загрузки или необходимости сборки собственного образа.

17.6.3 Образы среды принятия решений

Среда принятия решений (Decision Environment, DE) – это контейнер, аналогичный среде исполнения, который включает в себя все необходимое для выполнения сводов правил (rulebooks).

В Automation Hub доступны следующие готовые образы среды принятия решений:

- `aa-minimal-de` – образ с минимальными возможностями:
  - предназначен для разработчиков сводов правил и собственных образов DE для них;
  - позволяет собирать собственные образы с необходимым набором коллекций Ansible и модулей Python.
- `aa-full-de` – образ с расширенными возможностями:
  - предназначен для пользователей контроллера Event-Driven Automation;
  - позволяет использовать своды правил со стандартными источниками событий (event sources).

17.6.4 Образ для разработчика

Образ `aa-cdk` предназначен для разработчиков инфраструктурного кода Ansible. Он интегрируется с Visual Studio Code в качестве Dev Container и позволяет вести разработку и тестирование контента Ansible (наборы сценариев, коллекции) с использованием инструментов из состава Astra Automation CDK.

Подробную информацию об использовании образа см. в следующих документах:

- *Последовательность разработки*;
- *Рабочее окружение*.

Примечание

Образ `aa-cdk` не предназначен для использования в качестве среды исполнения или среды принятия решений.

17.6.5 Загрузка образа

Для загрузки образа среды исполнения используйте команду:

```
podman pull <image_name>:<version>
```

Здесь:

- <image_name> – название образа.
- <version> – версия образа. Если версия не указана, загружается образ с меткой `latest`, которой соответствует новейшая стабильная версия образа.

Совет

С помощью команды `tag` образу можно задать дополнительное название, например:

```
podman tag hub.astra-automation.ru/aa-1.2/aa-minimal-ee aa-minimal-ee
```

Теперь к образу `hub.astra-automation.ru/aa-1.2/aa-minimal-ee:latest` можно обращаться через псевдоним `aa-minimal-ee`.

Для получения состава образа используйте команду:

```
ansible-navigator images <tag>
```

Здесь <tag> – название образа.

Например, для получения состава образа `aa-minimal-ee:1.0.4` выполните следующие действия:

1. Загрузите образ среды исполнения:

   ```
   podman pull hub.astra-automation.ru/aa-1.2/aa-minimal-ee:1.0.4
   ```

2. Получите состав образа:

   ```
   ansible-navigator images hub.astra-automation.ru/aa-1.2/aa-minimal-ee:1.0.4
   ```

17.6.6 Использование

Для выполнения определенной команды внутри контейнера используйте следующий синтаксис:

```
ansible-navigator exec \
   --eei <image_name>:<version> \
   -- <container_command> [options]
```

Здесь:

- <image_name> – название образа;
- <version> – версия образа;
- <container_command> – команда, которая должна быть выполнена в контейнере.

17.6.7 Примеры

Следующие примеры демонстрируют использование EE для выполнения различных команд.

Примечание

Использование Ansible Navigator является рекомендуемым способом работы в командной строке, потому что он по умолчанию использует утилиты из служебного контейнера.

Проверка доступности узлов

Для проверки доступности всех узлов инвентаря выполните в каталоге проекта команду:

```
ansible-navigator exec -- ansible all -m ping
```

Убедитесь, что по каждому узлу получен положительный ответ типа pong, например:

```
web01 | SUCCESS => {
   "ansible_facts": {
      "discovered_interpreter_python": "/usr/bin/python3.7"
   },
   "changed": false,
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
   "ping": "pong"
}
```

Установка зависимостей Ansible

Для установки необходимых ресурсов, согласно зависимостям, определенным в файле `requirements.yml`, используйте команду:

```
ansible-navigator exec \
   --eei hub.astra-automation.ru/aa-1.2/aa-minimal-ee \
   -- ansible-galaxy install -r requirements.yml
```

Здесь:

- `hub.astra-automation.ru/aa-1.2/aa-minimal-ee` – название образа среды исполнения;
- `ansible-galaxy` – утилита Ansible, используемая для управления зависимостями;
- `requirements.yml` – название файла с перечнем зависимостей Ansible.

Запуск набора сценариев

Для запуска набора сценариев используйте команду:

```
ansible-navigator run playbook.yml \
   -i inventory.yml \
   --eei hub.astra-automation.ru/aa-1.2/aa-minimal-ee \
   -m stdout
```

Здесь:

- `hub.astra-automation.ru/aa-1.2/aa-minimal-ee` – название образа среды исполнения;
- `inventory.yml` – файл инвентаря;
- `playbook.yml` – файл сценария.

17.7 Разработка собственного EE для замкнутой программной среды

Режим замкнутой программной среды (ЗПС) обеспечивает динамический контроль неизменности (целостности) и подлинности файлов. Он предназначен для решения следующих задач:

- выявление несанкционированного изменения исполняемых и других файлов;
- предотвращение загрузки и исполнения измененных исполняемых файлов;
- предотвращение открытия измененных неисполняемых файлов.

Контроль целостности файлов реализован в невыгружаемом модуле ядра Linux `digsig_verif` и производится путем проверки цифровых подписей файлов.

17.7.1 Особенности режима ЗПС для контейнеров

Контейнер – это изолированный процесс операционной системы. В отличие от виртуальной машины, контейнер использует возможности ядра хостовой ОС.

Режим ЗПС накладывает ряд ограничений на использование исполняемых файлов в контейнерах:

1. На узле, где создается контейнер, должен быть доступен публичный ключ цифровой подписи, позволяющий проверить целостность исполняемых файлов в контейнере.

2. Все исполняемые файлы в контейнере должны иметь цифровую подпись. При ее отсутствии ядро операционной системы заблокирует загрузку и выполнение исполняемого файла.

Все исполняемые файлы из состава Astra Linux Special Edition имеют цифровую подпись и корректно работают в режиме ЗПС. Образы среды исполнения, разработанные ПАО Группа Астра и доступные в Automation Hub[246], могут быть использованы в режиме ЗПС без дополнительных настроек.

Программное обеспечение, исполняемые файлы которого не имеют цифровой подписи ПАО Группа Астра или ее партнеров, далее называется сторонним ПО.

Если вы хотите использовать стороннее ПО в собственных образах среды исполнения, следуйте этой инструкции.

Примечание

В этом руководстве рассматривается создание образа среды исполнения на базе образа `aa-minimal-ee`, доступного в Automation Hub.

17.7.2 Этапы разработки

Разработка образа среды исполнения, включающего стороннее ПО и способного работать в режиме ЗПС, состоит из следующих этапов:

1. Подготовка образа со всеми необходимыми пакетами, независимо от наличия цифровых подписей, и его проверка в окружении с выключенным режимом ЗПС.

   В данном руководстве этот этап не рассматривается. Для получения соответствующих инструкций обратитесь к *руководству по Ansible Builder*.

2. Получение ISO-образа с комплектом ключей для подписывания исполняемых файлов.
3. Подписывание исполняемых файлов, не имеющих цифровой подписи для работы в режиме ЗПС.

   В этом руководстве рассматривается подписывание исполняемых файлов формата ELF (Executable and Linkable Format[247]), распространяемых в составе DEB-пакетов и пакетов Python формата `.whl`.

4. Сборка образа среды исполнения с использованием подписанных пакетов. Далее эти шаги представлены более подробно.

17.7.3 Подготовка к работе

Подготовьте окружение к созданию собственного образа среды исполнения, способного работать в режиме ЗПС:

1. Изучите статьи Справочного центра:
   - Порядок создания подписей файлов для режима ЗПС[248];
   - Ограничения программной среды в Astra Linux Special Edition x.8[249].
2. Для ISO-образа с комплектом ключей заполните форму[250] на сайте ПАО Группа Астра.

   ISO-образ содержит следующие файлы:
   - `***_pub.key` – публичный ключ цифровой подписи;
   - `***_secret.gpg` – приватный ключ цифровой подписи;
   - `***_password.txt` – пароль приватного ключа цифровой подписи;

[246] https://hub.astra-automation.ru/
[247] https://ru.wikipedia.org/wiki/Executable_and_Linkable_Format
[248] https://wiki.astralinux.ru/pages/viewpage.action?pageId=61574460
[249] https://wiki.astralinux.ru/pages/viewpage.action?pageId=321820023
[250] https://astra.ru/ready-for-astra/become-tech-partner/

- ***_revoke.rev – сертификат отзыва ключа цифровой подписи.

3. Обновите список доступных пакетов:

```
sudo apt update
```

4. Установите утилиты, необходимые для подписи файлов и работы с архивами:

```
sudo apt install binutils bsign dpkg-dev file unzip zip --yes
```

5. Установите Ansible Builder, следуя *инструкции*.

17.7.4 Создание структуры проекта

Создайте структуру файлов и каталогов проекта:

```
project/
├── files/
│   ├── signed-deb/
│   └── signed-whl/
└── execution-environment.yml
```

Здесь:

- files/signed-deb/ – каталог для размещения подписанных DEB-пакетов;
- files/signed-whl/ – каталог для размещения подписанных пакетов Python формата .whl;
- execution-environment.yml – файл определения среды исполнения.

17.7.5 Подписывание файлов

Чтобы подписать DEB-пакеты и пакеты Python в формате .whl, выполните следующие действия:

1. Убедитесь, что каталог /media/cdrom/ существует и пуст. Если каталог не существует, для его создания выполните команду:

```
sudo mkdir -p /media/cdrom/
```

2. Смонтируйте ISO-образ с ключами в каталог /media/cdrom:

```
sudo mount /path/to/image.iso /media/cdrom -t iso9660 -o ro
```

3. Сохраните пароль приватного ключа в переменной окружения pass:

```
pass=$(cat /media/cdrom/<passfile>)
```

   Здесь <passfile> – название файла с паролем.

4. Загрузите секрет, которым защищен закрытый ключ:

```
echo "$pass" | gpg \
   --batch \
   --yes \
   --pinentry-mode loopback \
   --passphrase-fd 0 \
   --import /media/cdrom/<secret>.gpg
```

   Здесь <secret>.gpg – название файла с секретом.

   Если загрузка секрета выполнена успешно, в терминал выводится набор строк следующего вида:

```
gpg: создан каталог '/home/<user>/.gnupg'
gpg: создан щит с ключами '/home/<user>/.gnupg/pubring.kbx'
gpg: ключ B0F6B01F548C108D: 1 подпись не проверена за отсутствием ключа
gpg: /home/<user>/.gnupg/trustdb.gpg: создана таблица доверия
gpg: ключ B0F6********108D: импортирован открытый ключ "ООО "РусБИТех-Астра" (key␣
↪for signing) <info@astralinux.ru>"
gpg: ключ B0F6********108D: импортирован секретный ключ
gpg: Всего обработано: 1
gpg:                    импортировано: 1
gpg:     прочитано секретных ключей: 1
gpg: импортировано секретных ключей: 1
gpg: абсолютно доверенных ключей не найдено
```

5. Загрузите закрытый ключ:

```
key=$(gpg --with-colons --list-secret-keys | awk -F: '/fpr:/ {print $10; exit}')
```

6. Скопируйте DEB-пакеты в каталог `files/signed-deb/`, а пакеты Python в формате `.whl` в каталог `files/signed-whl`.

Подписывание DEB-пакетов

Чтобы подписать содержимое DEB-пакета, выполните следующие действия:

1. Создайте временный каталог, например:

```
tmp_dir=$(mktemp --directory)
```

2. Распакуйте содержимое пакета в этот каталог:

```
dpkg-deb -R <package>.deb "$tmp_dir"
```

3. Найдите и подпишите все ELF-файлы во временном каталоге:

```
find "$tmp_dir" -exec file {} \; | grep -i -e ELF -e 'shared object' | cut -d":" -
↪f1 | while read file; do
  bsign -s -N --pgoptions "--default-key=$key --passphrase $pass" "$file"
done
```

4. Запакуйте содержимое временного каталога в новый DEB-пакет:

```
dpkg-deb -b "$tmp_dir" <package>_signed.deb
```

5. Удалите временный каталог:

```
rm -rf "$tmp_dir"
```

Подписывание пакетов Python

Чтобы подписать содержимое пакета Python в формате `.whl`, выполните следующие действия:

1. Создайте временный каталог, например:

```
tmp_dir=$(mktemp --directory)
```

2. Распакуйте содержимое пакета во временный каталог:

```
unzip -q <package>.whl -d "$tmp_dir"
```

3. Найдите и подпишите все ELF-файлы во временном каталоге:

```
find "$tmp_dir" -exec file {} \; | grep -i -e ELF -e 'shared object' | cut -d":" -
↪f1 | while read file; do
  bsign -s -N --pgoptions "--default-key=$key --passphrase $pass" "$file"
done
```

4. Перейдите во временный каталог:

```
cd "$tmp_dir"
```

5. Запакуйте содержимое временного каталога в новый пакет `.whl`:

```
zip -r -X --symlinks <package>_signed.whl ./*
```

6. Переместите подписанный пакет в каталог `files/signed-whl/`:

```
mv <package>_signed.whl /path/to/project/files/signed-whl/
```

7. Удалите временный каталог:

```
rm -rf "$tmp_dir"
```

17.7.6 Сборка образа

Для сборки образа с подписанными файлами выполните следующие действия:

1. Файл определения среды исполнения заполните по образцу:

Список 32: `execution-environment.yml`

```
---
version: 3

images:
  base_image:
    name: hub.astra-automation.ru/aa-1.2/aa-minimal-ee:latest

dependencies: {}

additional_build_files:
  - src: files/
    dest: .

additional_build_steps:
  prepend_base:
    - RUN apt update
    - COPY _build/signed-deb/ /opt/deb/
    - COPY _build/signed-whl/ /opt/whl/
    - RUN dpkg -i /opt/deb/*.deb || apt-get install -f -y && dpkg -i /opt/deb/*.deb
    - RUN pip3 install --no-index --find-links=/opt/whl/ /opt/whl/*.whl
```

2. Запустите сборку образа как в следующем примере:

```
ansible-builder build -t signed-ee:1.0.0
```

17.7.7 Проверка работоспособности образа

Для проверки работоспособности собранного образа выполните следующие действия:

1. Если в каталоге `/etc/digsig/keys/` не существует подкаталог `legacy/keys/`, создайте его:

```
sudo mkdir -p /etc/digsig/keys/legacy/keys/
```

2. Скопируйте публичный ключ цифровой подписи в каталог `/etc/digsig/keys/legacy/keys/`:

```
sudo cp /media/cdrom/<pubkey_file> /etc/digsig/keys/legacy/keys/pubkey.gpg
```

3. Включите режим ЗПС:

```
sudo astra-digsig-control enable
```

4. Перезагрузите компьютер.
5. Запустите образ в интерактивном режиме, например:

```
podman run --rm --tty --interactive signed-ee:1.0.0 /bin/bash
```

6. Для проверки работоспособности утилит, установленных из DEB-пакетов, запустите их, например:

```
cowsay "Testing signing"
```

   Режим ЗПС заблокирует запуск, если у ELF-файлов утилиты нет цифровой подписи.

7. Для проверки работоспособности пакетов Python выполните следующие действия:
   1. Запустите интерпретатор:

```
python3
```

   2. Импортируйте модули, установленные из подписанных пакетов Python, например:

```
import pandas
import numpy
```

      Режим ЗПС заблокирует импорт, а интерпретатор Python выведет сообщения об ошибках, если у ELF-файлов модуля нет цифровой подписи.

   3. Завершите работу с интерпретатором:

```
exit()
```

17.8 Справочные данные

Здесь приведены:

- справочные данные по назначению и применению утилит для создания и управления контентом Ansible;
- сведения о версиях компонентов в образах среды исполнения, созданных ПАО Группа Астра.

17.8.1 Ansible Navigator

Утилита Ansible Navigator является универсальным инструментом для разработки, тестирования и применения инфраструктурного кода в виде коллекций и сценариев. Для этого она предоставляет несколько команд с различными аргументами.

Вызов утилиты имеет следующий вид:

```
ansible-navigator {<command>} {<arguments>}
```

Для понимания принципов работы утилиты обратитесь к ее *описанию*.

Настройки Ansible Navigator могут быть заданы следующими способами, в порядке возрастания приоритета:

- конфигурационный файл;
- переменная окружения;
- аргументы CLI.

Общие аргументы

Ansible Navigator принимает аргументы, общие для всех команд, а также специфичные для утилит, которые используются для выполнения отдельных задач.

Аргументы, общие для всех команд:

-h, --help
: Вывод справочной информации о командах и аргументах Ansible Navigator.

--version
: Вывод информации о версии Ansible Navigator.

Основные настройки

Основные настройки управляют поведением самого Ansible Navigator.

ansible-runner-artifact-dir

Путь к каталогу для хранения артефактов, созданных во время запуска `ansible-runner`.

Аргументы CLI: `--rad`, `--ansible-runner-artifact-dir`.

Переменная окружения: `ANSIBLE_NAVIGATOR_ANSIBLE_RUNNER_ARTIFACT_DIR`.

Конфигурационный файл:

```
ansible-navigator:
  ansible-runner:
    artifact-dir:
```

ansible-runner-rotate-artifacts-count

Удаление каталогов с артефактами, кроме последних N запусков `ansible-runner`. При значении `0` не удаляется ни один каталог с артефактами.

Аргументы CLI: `--rac`, `--ansible-runner-rotate-artifacts-count`.

Переменная окружения: `ANSIBLE_NAVIGATOR_ANSIBLE_RUNNER_ROTATE_ARTIFACTS_COUNT`.

Конфигурационный файл:

```
ansible-navigator:
  ansible-runner:
    rotate-artifacts-count:
```

ansible-runner-timeout

Период ожидания в секундах перед принудительным завершением выполнения `ansible-runner`.

Аргументы CLI: `--rt`, `--ansible-runner-timeout`.

Переменная окружения: `ANSIBLE_NAVIGATOR_ANSIBLE_RUNNER_TIMEOUT`.

Конфигурационный файл:

```
ansible-navigator:
  ansible-runner:
    timeout:
```

ansible-runner-write-job-events

Запись событий `job_events` в каталог артефактов:

Возможные значения:

- `true` – запись включена;

- false – запись выключена.

Аргументы CLI: --rwje, --ansible-runner-write-job-events.

Переменная окружения: ANSIBLE_NAVIGATOR_ANSIBLE_RUNNER_WRITE_JOB_EVENTS.

Конфигурационный файл:

```
ansible-navigator:
  ansible-runner:
    job-events:
```

app

Команда, выполняемая при запуске Ansible Navigator по умолчанию.

Возможные значения:

- builder;
- collections;
- config;
- doc;
- exec;
- images;
- inventory;
- lint;
- replay;
- run;
- settings;
- welcome.

Значение по умолчанию: welcome.

Аргументы CLI: позиционные.

Переменная окружения: ANSIBLE_NAVIGATOR_APP.

Конфигурационный файл:

```
ansible-navigator:
  app:
```

cmdline

Дополнительные параметры, передаваемые в команды Ansible (ansible-playbook, ansible-doc и так далее).

Аргументы CLI: позиционные.

Переменная окружения: ANSIBLE_NAVIGATOR_CMDLINE.

Конфигурационный файл:

```
ansible-navigator:
  ansible:
    cmdline:
```

collection-doc-cache-path

Путь к каталогу для хранения кеша документации коллекций.

Значение по умолчанию: `~/.cache/ansible-navigator/collection_doc_cache.db`.

Аргументы CLI: `--cdcp`, `--collection-doc-cache-path`.

Переменная окружения: `ANSIBLE_NAVIGATOR_COLLECTION_DOC_CACHE_PATH`.

Конфигурационный файл:

```
ansible-navigator:
  collection-doc-cache-path:
```

container-engine

Инструмент контейнеризации:

Возможные значения:

- `auto` – тип инструмента контейнеризации выбирается автоматически.

  Примечание

  Если в системе установлены Docker и Podman, будет использоваться Podman.

- `podman` – Podman.
- `docker` – Docker.

Значение по умолчанию: `auto`.

Аргументы CLI: `--ce`, `--container-engine`.

Переменная окружения: `ANSIBLE_NAVIGATOR_CONTAINER_ENGINE`.

Конфигурационный файл:

```
ansible-navigator:
  execution-environment:
    container-engine:
```

container-options

Дополнительные параметры, передаваемые в команду запуска инструмента контейнеризации.

Аргументы CLI: `--co`, `--container-options`.

Переменная окружения: `ANSIBLE_NAVIGATOR_CONTAINER_OPTIONS`.

Конфигурационный файл:

```
ansible-navigator:
  execution-environment:
    container-options:
```

display-color

Использование цветового оформления.

Возможные значения:

- `true` – включено;
- `false` – выключено.

Значение по умолчанию: `true`.

Аргументы CLI: `--dc`, `--display-color`.

Переменная окружения: `NO_COLOR`.

Конфигурационный файл:

```
ansible-navigator:
  color:
    enable:
```

editor-command

Команда запуска текстового редактора.

Значение по умолчанию: `vi +{line_number} {filename}`.

Аргументы CLI: `--ecmd`, `--editor-command`.

Переменная окружения: `ANSIBLE_NAVIGATOR_EDITOR_COMMAND`.

Конфигурационный файл:

```
ansible-navigator:
  editor:
    command:
```

editor-console

Использование редактора в режиме *TUI*.

Возможные значения:

- `true` – текстовый интерфейс;
- `false` – режим работы определяется редактором.

Значение по умолчанию: `true`.

Аргументы CLI: `--econ`, `--editor-console`.

Переменная окружения: `ANSIBLE_NAVIGATOR_EDITOR_CONSOLE`.

Конфигурационный файл:

```
ansible-navigator:
  editor:
    console:
```

execution-environment

Использование среды исполнения.

Возможные значения:

- `true` – разрешено.
- `false` – запрещено.

Значение по умолчанию: `true`.

Аргументы CLI: `--ee`, `--execution-environment`.

Переменная окружения: `ANSIBLE_NAVIGATOR_EXECUTION_ENVIRONMENT`.

Конфигурационный файл:

```
ansible-navigator:
  execution-environment:
    enabled:
```

execution-environment-image

Название образа среды исполнения.

Значение по умолчанию: `hub.astra-automation.ru/aa-1.2/aa-full-ee:latest`.

Аргументы CLI: `--eei`, `--execution-environment-image`.

Переменная окружения: `ANSIBLE_NAVIGATOR_EXECUTION_ENVIRONMENT_IMAGE`.

Конфигурационный файл:

```
ansible-navigator:
  execution-environment:
    image:
```

execution-environment-volume-mounts

Монтирование тома <volume> в точку <mount_point> контейнера при запуске среды исполнения.

Подробности о монтировании томов см. в документации используемой системы контейнеризации:

- Docker[251];
- Podman[252].

Аргументы CLI: `--eev`, `--execution-environment-volume-mounts`.

Переменная окружения: `ANSIBLE_NAVIGATOR_EXECUTION_ENVIRONMENT_VOLUME_MOUNTS`.

Конфигурационный файл:

```
ansible-navigator:
  execution-environment:
    volume-mounts:
```

log-append

Управление записью журнала сообщений в файл.

Возможные значения:

- `true` – для хранения всех записей используется один файл журнала;
- `false` – для каждой сессии создается отдельный файл журнала.

Значение по умолчанию: `true`.

Аргументы CLI: `--la`, `--log-append`.

Переменная окружения: `ANSIBLE_NAVIGATOR_LOG_APPEND`.

Конфигурационный файл:

```
ansible-navigator:
  logging:
    append:
```

log-file

Путь к файлу хранения журнала Ansible Navigator.

Значение по умолчанию: `./ansible-navigator.log`.

Аргументы CLI: `--lf`, `--log-file`.

[251] https://docs.docker.com/engine/storage/volumes/
[252] https://docs.podman.io/en/latest/markdown/podman-volume-mount.1.html

Переменная окружения: ANSIBLE_NAVIGATOR_LOG_FILE.

Конфигурационный файл:

```
ansible-navigator:
  logging:
    file:
```

log-level

Минимальный уровень важности сообщения, необходимый для его записи в журнал Ansible Navigator.

Возможные значения: (в порядке возрастания важности):

- debug;
- info;
- warning;
- error;
- critical.

Значение по умолчанию: warning.

Аргументы CLI: --ll, --log-level.

Переменная окружения: ANSIBLE_NAVIGATOR_LOG_LEVEL.

Конфигурационный файл:

```
ansible-navigator:
  logging:
    level:
```

mode

Режим работы Ansible Navigator.

Возможные значения:

- stdout – использование стандартного потока вывода;
- interactive – интерактивный режим.

Значение по умолчанию: interactive.

Аргументы CLI: -m, --mode.

Переменная окружения: ANSIBLE_NAVIGATOR_MODE.

Конфигурационный файл:

```
ansible-navigator:
  mode:
```

osc4

Использование в терминале цветового оформления в соответствии со стандартом OSC 4.

Возможные значения:

- true – включено;
- false – выключено.

Значение по умолчанию: true.

Аргументы CLI: --osc4, --osc4.

Переменная окружения: ANSIBLE_NAVIGATOR_OSC4.

Конфигурационный файл:

```
ansible-navigator:
  color:
    osc4:
```

pass-environment-variable

Передача существующей переменной окружения в среду исполнения.

Аргументы CLI: --penv, --pass-environment-variable.

Переменная окружения: ANSIBLE_NAVIGATOR_PASS_ENVIRONMENT_VARIABLES.

Конфигурационный файл:

```
ansible-navigator:
  execution-environment:
    environment-variables:
      pass:
```

pull-arguments

Дополнительные параметры для команды получения (pull) образа среды исполнения из реестра, например:

```
--pull-arguments='--tls-verify=false'
```

Аргументы CLI: --pa, --pull-arguments.

Переменная окружения: ANSIBLE_NAVIGATOR_PULL_ARGUMENTS.

Конфигурационный файл:

```
ansible-navigator:
  execution-environment:
    pull:
      arguments:
```

pull-policy

Политика получения образа среды исполнения.

Возможные значения:

- always – всегда загружать образ из реестра.
- missing – загрузить образ из реестра только в том случае, когда нет его локальной копии.
- never – никогда не загружать образ из реестра.
- tag – если значение тега равно latest, образ будет загружен из реестра. Если значение тега отличается от latest, образ будет загружен из реестра только при отсутствии локальной копии.

Значение по умолчанию: tag.

Аргументы CLI: --pp, --pull-policy.

Переменная окружения: ANSIBLE_NAVIGATOR_PULL_POLICY.

Конфигурационный файл:

```
ansible-navigator:
  execution-environment:
    pull:
      policy:
```

set-environment-variable

Передача в среду исполнения переменной окружения <varibable> со значением <value>.

Аргументы CLI: `--senv`, `--set-environment-variable`.

Переменная окружения: `ANSIBLE_NAVIGATOR_SET_ENVIRONMENT_VARIABLES`.

Конфигурационный файл:

```
ansible-navigator:
  execution-environment:
    environment-variables:
      set:
```

time-zone

Используемый часовой пояс.

Возможные значения:

- название часового пояса в формате IANA[253];
- `local`.

Значение по умолчанию: `UTC`.

Аргументы CLI: `--tz`, `--time-zone`.

Переменная окружения: `TZ`.

Конфигурационный файл:

```
ansible-navigator:
  time-zone:
```

ssh-forward-keys

Монтирование каталога `~/.ssh/` в файловую систему контейнера.

Возможные значения:

- `true`;
- `false`.

Значение по умолчанию: `true`.

Аргументы CLI: `--ssh-forward-keys`, `--ssh-forward-keys`.

Переменная окружения: `ANSIBLE_NAVIGATOR_SSH_FORWARD_KEYS`.

Конфигурационный файл:

```
ansible-navigator:
  execution-environment:
    ssh-forwarding:
      keys:
```

[253] https://www.iana.org/time-zones

ssh-forward-config

Монтирование файла `~/.ssh/config` в файловую систему контейнера.

Возможные значения:

- `true`;
- `false`.

Значение по умолчанию: `true`.

Аргументы CLI: `--ssh-forward-config`, `--ssh-forward-config`.

Переменная окружения: `ANSIBLE_NAVIGATOR_SSH_FORWARD_CONFIG`.

Конфигурационный файл:

```
ansible-navigator:
  execution-environment:
    ssh-forwarding:
      config:
```

ssh-forward-agent

Передача переменной окружения `SSH_AUTH_SOCK` в контейнер.

Возможные значения:

- `true`;
- `false`.

Значение по умолчанию: `true`.

Аргументы CLI: `--ssh-forward-agent`, `--ssh-forward-agent`.

Переменная окружения: `ANSIBLE_NAVIGATOR_SSH_FORWARD_AGENT`.

Конфигурационный файл:

```
ansible-navigator:
  execution-environment:
    ssh-forwarding:
      agent:
```

Настройки команд

Настройки команд управляют поведением утилит, используемых при выполнении команд Ansible Navigator.

builder

help-builder

Вывод справки к команде `ansible-builder` при работе в режиме `stdout`.

Возможные значения:

- `True`;
- `False`.

Значение по умолчанию: `False`.

Аргументы CLI: `--hb`, `--help-builder`.

Переменная окружения: `ANSIBLE_NAVIGATOR_HELP_BUILDER`.

Конфигурационный файл:

```
ansible-navigator:
  ansible-builder:
    help:
```

workdir

Путь к каталогу с манифестами `ansible-builder`.

Значение по умолчанию: `.` (текущий каталог).

Аргументы CLI: `--bwd`, `--workdir`.

Переменная окружения: `ANSIBLE_NAVIGATOR_WORKDIR`.

Конфигурационный файл:

```
ansible-navigator:
  ansible-builder:
    workdir:
```

collections

format

Формат для вывода в `stdout`.

Возможные значения:

- `json`;
- `yaml`.

Значение по умолчанию: `yaml`.

Аргументы CLI: `--fmt`, `--format`.

Переменная окружения: `ANSIBLE_NAVIGATOR_FORMAT`.

Конфигурационный файл:

```
ansible-navigator:
  format:
```

config

config

Путь к конфигурационному файлу Ansible.

Аргументы CLI: `-c`, `--config`.

Переменная окружения: `ANSIBLE_CONFIG`.

Конфигурационный файл:

```
ansible-navigator:
  ansible:
    config:
      path:
```

help-config

Вывод справки к команде `ansible-config` при работе в режиме `stdout`.

Возможные значения:

- `True`;

- False.

Значение по умолчанию: False.

Аргументы CLI: --hc, --help-config.

Переменная окружения: ANSIBLE_NAVIGATOR_HELP_CONFIG.

Конфигурационный файл:

```
ansible-navigator:
  ansible:
    config:
      help:
```

doc

help-doc

Вывод справки к команде ansible-doc при работе в режиме stdout.

Возможные значения:

- True;
- False.

Значение по умолчанию: False.

Аргументы CLI: --hd, --help-doc.

Переменная окружения: ANSIBLE_NAVIGATOR_HELP_DOC.

Конфигурационный файл:

```
ansible-navigator:
  ansible:
    doc:
      help:
```

plugin-name

Название расширения.

Аргументы CLI: позиционные.

Переменная окружения: ANSIBLE_NAVIGATOR_PLUGIN_NAME.

Конфигурационный файл:

```
ansible-navigator:
  ansible:
    doc:
      plugin:
        name:
```

plugin-type

Название типа расширения.

Возможные значения:

- become;
- cache;
- callback;
- cliconf;
- connection;

- filter;
- httpapi;
- inventory;
- keyword;
- lookup;
- module;
- netconf;
- role;
- shell;
- strategy;
- test;
- vars.

Значение по умолчанию: module.

Аргументы CLI: -t, --type.

Переменная окружения: ANSIBLE_NAVIGATOR_PLUGIN_TYPE.

Конфигурационный файл:

```
ansible-navigator:
  ansible:
    doc:
      plugin:
        type:
```

exec

exec-command

Команда для запуска в контейнере со средой исполнения.

Значение по умолчанию: /bin/bash.

Аргументы CLI: позиционные.

Переменная окружения: ANSIBLE_NAVIGATOR_EXEC_COMMAND.

Конфигурационный файл:

```
ansible-navigator:
  exec:
    command:
```

exec-shell

Необходимость выполнения команды в оболочке.

Возможные значения:

- True;
- False.

Значение по умолчанию: True.

Аргументы CLI: --exshell, --exec-shell.

Переменная окружения: ANSIBLE_NAVIGATOR_EXEC_SHELL.

Конфигурационный файл:

```
ansible-navigator:
  exec:
    shell:
```

images

format

Формат для вывода в stdout.

Возможные значения:

- json;
- yaml.

Значение по умолчанию: yaml.

Аргументы CLI: --fmt, --format.

Переменная окружения: ANSIBLE_NAVIGATOR_FORMAT.

Конфигурационный файл:

```
ansible-navigator:
  format:
```

images-details

Список разделов, отображаемых при выводе подробной информации об образе среды исполнения.

Возможные значения элементов списка:

- ansible_collections – коллекции Ansible;
- ansible_version – версия Ansible;
- everything – все возможные поля;
- os_release – версия ОС;
- python_packages – пакеты Python;
- python_version – версия интерпретатора Python;
- system_packages – пакеты, установленные с помощью системного менеджера пакетов.

Значение по умолчанию: ['everything'].

Аргументы CLI: -d, --details.

Переменная окружения: ANSIBLE_NAVIGATOR_IMAGES_DETAILS.

Конфигурационный файл:

```
ansible-navigator:
  images:
    details:
```

inventory

help-inventory

Вывод справки к команде ansible-inventory при работе в режиме stdout.

Возможные значения:

- True;

- False.

Значение по умолчанию: False.

Аргументы CLI: --hi, --help-inventory.

Переменная окружения: ANSIBLE_NAVIGATOR_HELP_INVENTORY.

Конфигурационный файл:

```
ansible-navigator:
  ansible:
    inventory:
      help:
```

inventory

Путь к файлу инвентаря или список узлов одной строкой через запятую.

Аргументы CLI: -i, --inventory.

Переменная окружения: ANSIBLE_INVENTORY.

Конфигурационный файл:

```
ansible-navigator:
  ansible:
    inventory:
      entries:
```

inventory-column

Атрибуты узлов для показа в режиме просмотра инвентаря.

Аргументы CLI: --ic, --inventory-column.

Переменная окружения: ANSIBLE_NAVIGATOR_INVENTORY_COLUMNS.

Конфигурационный файл:

```
ansible-navigator:
  inventory-columns:
```

lint

lint-config

Путь к конфигурационному файлу ansible-lint.

Аргументы CLI: --lic, --lint-config.

Переменная окружения: ANSIBLE_LINT_CONFIG.

Конфигурационный файл:

```
ansible-navigator:
  ansible-lint:
    config:
```

lintables

Путь к файлам для обработки с помощью ansible-lint.

Аргументы CLI: позиционные.

Переменная окружения: ANSIBLE_NAVIGATOR_LINTABLES.

Конфигурационный файл:

```
ansible-navigator:
  ansible-lint:
    lintables:
```

replay

playbook-artifact-replay

Путь к артефактам сценариев для воспроизведения.

Аргументы CLI: позиционные.

Переменная окружения: ANSIBLE_NAVIGATOR_PLAYBOOK_ARTIFACT_REPLAY.

Конфигурационный файл:

```
ansible-navigator:
  playbook-artifact:
    replay:
```

run

enable-prompts

Разрешение запроса на ввод пароля при выполнении набора сценариев. Включение этой настройки устанавливает режим stdout и запрещает создание артефактов.

Возможные значения:

- True;
- False.

Значение по умолчанию: False.

Аргументы CLI: --ep, --enable-prompts.

Переменная окружения: ANSIBLE_NAVIGATOR_ENABLE_PROMPTS.

Конфигурационный файл:

```
ansible-navigator:
  enable-prompts:
```

help-playbook

Вывод справки к команде ansible-playbook при работе в режиме stdout.

Возможные значения:

- True;
- False.

Значение по умолчанию: False.

Аргументы CLI: --hp, --help-playbook.

Переменная окружения: ANSIBLE_NAVIGATOR_HELP_PLAYBOOK.

Конфигурационный файл:

```
ansible-navigator:
  ansible:
    playbook:
      help:
```

inventory

Путь к файлу инвентаря или список узлов одной строкой через запятую.

Аргументы CLI: `-i`, `--inventory`.

Переменная окружения: `ANSIBLE_INVENTORY`.

Конфигурационный файл:

```
ansible-navigator:
  ansible:
    inventory:
      entries:
```

inventory-column

Атрибуты узлов для показа в режиме просмотра инвентаря.

Аргументы CLI: `--ic`, `--inventory-column`.

Переменная окружения: `ANSIBLE_NAVIGATOR_INVENTORY_COLUMNS`.

Конфигурационный файл:

```
ansible-navigator:
  inventory-columns:
```

playbook

Путь к файлу набора сценариев.

Аргументы CLI: позиционные.

Переменная окружения: `ANSIBLE_NAVIGATOR_PLAYBOOK`.

Конфигурационный файл:

```
ansible-navigator:
  ansible:
    playbook:
      path:
```

playbook-artifact-enable

Включение или выключение создания артефактов для выполненных наборов сценариев.

Примечание

Эта настройка несовместима с режимом `stdout`, требующим пользовательский ввод.

Возможные значения:

- `True`;
- `False`.

Значение по умолчанию: `True`.

Аргументы CLI: `--pae`, `--playbook-artifact-enable`.

Переменная окружения: `ANSIBLE_NAVIGATOR_PLAYBOOK_ARTIFACT_ENABLE`.

Конфигурационный файл:

```
ansible-navigator:
  playbook-artifact:
    enable:
```

playbook-artifact-save-as

Шаблон названия файлов артефактов, создаваемых для выполненных наборов сценариев.

Доступны следующие подстановочные символы:

- {playbook_dir} – каталог с наборами сценариев;
- {playbook_name} – название файла набора сценариев;
- {playbook_status} – статус выполнения набора сценариев;
- {time_stamp} – дата и время создания артефакта.

Значение по умолчанию: {playbook_dir}/{playbook_name}-artifact-{time_stamp}.json.

Аргументы CLI: --pas, --playbook-artifact-save-as.

Переменная окружения: ANSIBLE_NAVIGATOR_PLAYBOOK_ARTIFACT_SAVE_AS.

Конфигурационный файл:

```
ansible-navigator:
  playbook-artifact:
    save-as:
```

settings

settings-effective

Необходимость отображения действующих настроек.

При расчете значений сочетаются:

- значения по умолчанию;
- параметры CLI;
- переменные окружения;
- настройки из конфигурационного файла.

Значение по умолчанию: False.

Аргументы CLI: --se, --effective.

Переменная окружения: ANSIBLE_NAVIGATOR_SETTINGS_EFFECTIVE.

Конфигурационный файл:

```
ansible-navigator:
  settings:
    effective:
```

settings-sample

Генерация примера конфигурационного файла.

Значение по умолчанию: False.

Аргументы CLI: --gs, --sample.

Переменная окружения: ANSIBLE_NAVIGATOR_SETTINGS_SAMPLE.

Конфигурационный файл:

```
ansible-navigator:
  settings:
    sample:
```

settings-schema

Генерация схемы конфигурационного файла.

Возможное значение: `json`.

Значение по умолчанию: `json`.

Аргументы CLI: `--ss`, `--schema`.

Переменная окружения: `ANSIBLE_NAVIGATOR_SETTINGS_SCHEMA`.

Конфигурационный файл:

```
ansible-navigator:
  settings:
    schema:
```

settings-sources

Отображение источника каждой действующей настройки.

Значение по умолчанию: `False`.

Аргументы CLI: `--so`, `--sources`.

Переменная окружения: `ANSIBLE_NAVIGATOR_SETTINGS_SOURCES`.

Конфигурационный файл:

```
ansible-navigator:
  settings:
    sources:
```

17.8.2 Ansible Builder

Утилита Ansible Builder является универсальным инструментом для создания образов среды исполнения (EE (Execution Environment, среда исполнения)). Для этого она предоставляет несколько команд с различными аргументами, описание которых приводится далее.

Вызов утилиты имеет следующий вид:

```
ansible-builder <command> [<arguments>]
```

Здесь:

- <command> – команда;
- <arguments> – аргументы команды.

Возможные команды:

- *build* – создание образа контейнера.
- *create* – создание файла спецификации образа.
- *introspect* – поиск в каталоге проекта сведений о коллекциях и вывод информации об зависимостях.

build

Аргументы команды:

-h, **--help**
Справочная информация о команде.

-t TAG [TAG ...], **--tag** TAG [TAG ...]
Название создаваемого образа (по умолчанию – ansible-execution-env:latest).

--container-runtime {podman,docker}
Система управления контейнерами:

- docker – Docker;
- podman – Podman.

Значение по умолчанию: podman.

--build-arg BUILD_ARGS
Переменные, которые могут быть использованы во время сборки:

- ANSIBLE_GALAXY_CLI_COLLECTION_OPTS – позволяет пользователю передать параметр –pre для команды установки коллекций (ansible-galaxy collections install), чтобы разрешить установку предварительных выпусков коллекций.
- ANSIBLE_GALAXY_CLI_ROLE_OPTS – позволяет пользователю передавать любые параметры для команды установки ролей, такие как –no-deps.
- EE_BASE_IMAGE – указывает родительский образ для среды исполнения.
- EE_BUILDER_IMAGE – указывает образ, используемый для компиляции типовых задач.
- PKGMGR_PRESERVE_CACHE – контролирует частоту очистки кеша менеджера пакетов в процессе сборки образа. Если это значение не задано, кеш очищается наиболее часто. Если задано значение always , кеш никогда не очищается. Любое другое значение заставляет очищать кеш только после установки системных зависимостей на заключительном этапе сборки образа.

--no-cache
Отключение кеширования во время сборки образа.

--prune-images
Удаление неиспользуемых образов после создания образа.

--container-policy {system,ignore_all,signature_required}
Политика проверки образа контейнера:

- system – использование системной политики или подписи со встроенными путями к связкам ключей;
- ignore_all – игнорирование всех подписей;
- signature_required – использование подписей, указанных с помощью параметра --container-keyring.

> **Важно**
>
> Аргумент поддерживается только при использовании Podman.

--container-keyring CONTAINER_KEYRING
Путь к хранилищу контейнера, которое может использоваться для хранения и управления учетными данными контейнера, а также для обеспечения безопасного доступа к защищенным ресурсам.

--squash {new,all,off}
Объединение слоев во время сборки, что позволяет уменьшить размер конечного образа:

- new – существовавшие ранее слои не изменяются, а все новые слои объединяются в один;
- all – все слои образа объединяются в один;
- off – объединение слоев отключено.

Значение по умолчанию: off.

Важно

Аргумент поддерживается только при использовании Podman.

-f FILENAME, **--file** FILENAME

Путь к файлу определения среды исполнения.

Значение по умолчанию: execution-environment.(yml|yaml).

-c BUILD_CONTEXT, **--context** BUILD_CONTEXT

Каталог, который будет использоваться для хранения контекста сборки.

Значение по умолчанию: context.

--output-filename {Containerfile,Dockerfile}

Название файла для записи определения образа, зависит от --container-runtime:

- Containerfile – Podman;
- Dockerfile – Docker.

--galaxy-keyring GALAXY_KEYRING

Путь к хранилищу для проверки подписи коллекции при загрузке из Galaxy. Проверка отключена, если она не настроена.

--galaxy-ignore-signature-status-codes GALAXY_IGNORE_SIGNATURE_STATUS_CODES

Позволяет игнорировать определенные коды состояния подписи при работе с Galaxy. Может быть указано несколько раз.

--galaxy-required-valid-signature-count GALAXY_REQUIRED_VALID_SIGNATURE_COUNT

Количество действительных подписей, необходимых для успешной загрузки коллекции с Ansible Galaxy.

-v [VERBOSITY], **--verbosity** [VERBOSITY]

Детализация вывода. Добавление нескольких -v увеличит детализацию. Максимальное значение -vvv. Также принимаются целочисленные значения, например, -v3 или --verbosity 3.

Значение по умолчанию: 2.

create

Аргументы команды:

- -h, --help;
- -f FILENAME, --file FILENAME;
- -c BUILD_CONTEXT, --context BUILD_CONTEXT;
- --output-filename {Containerfile,Dockerfile};
- --galaxy-keyring GALAXY_KEYRING;
- --galaxy-ignore-signature-status-codes GALAXY_IGNORE_SIGNATURE_STATUS_CODES;
- --galaxy-required-valid-signature-count GALAXY_REQUIRED_VALID_SIGNATURE_COUNT;
- -v [VERBOSITY], --verbosity [VERBOSITY].

Подробное описание аргументов приведено в описании команды *build*.

introspect

Аргументы команды:

folder
: Путь к каталогу с коллекциям Ansible. Внутри указанного каталога должен существовать подкаталог `ansible_collections`.

-h, --help
: Справочная информация о работе команды.

--sanitize
: Очистка зависимостей или данных, которые могут быть связаны с проектом. Например, удаление непроверенных или ненадежных зависимостей, чтобы гарантировать, что результирующий список остается безопасным и актуальным.

--user-pip USER_PIP
: Путь к файлу `requirements.txt`.

--user-bindep USER_BINDEP
: Путь к файлу `bindep.txt`.

--write-pip WRITE_PIP
: Путь к файлу, в который будет записан список зависимостей Python.

--write-bindep WRITE_BINDEP
: Путь к файлу, в который будет записан список системных зависимостей.

-v [VERBOSITY], **--verbosity** [VERBOSITY]
: Детализация вывода. Добавление нескольких `-v` увеличит детализацию. Максимальное значение `-vvv`. Также принимаются целочисленные значения, например, `-v3` или `--verbosity 3`.

  Значение по умолчанию: 2.

17.8.3 Ansible Creator

Утилита Ansible Creator упрощает создание базовой структуры коллекций и сценариев Ansible. Для этого она предоставляет несколько команд с различными аргументами, описание которых приводится далее.

Вызов утилиты имеет следующий вид:

```
ansible-creator [-h] [--version] <command> [<arguments>]
```

Здесь:

- <command> – команда;
- <arguments> – аргументы команды.

Возможные аргументы утилиты:

-h, --help
: Вывод справочной информации о командах и аргументах Ansible Creator.

--version
: Вывод информации о версии Ansible Creator.

-v, --verbose
: Увеличение уровня детализации вывода.

  Значение по умолчанию: 2.

Команды

При запуске Ansible Creator можно использовать следующие команды:

- *add* – добавление содержимого в уже существующий проект;
- *init* – инициализация коллекции или набора сценариев.

add

Важно

Эта часть документации находится в стадии разработки.

init

Эта команда создает структуру файлов и каталогов, необходимую для начала работы над коллекцией или набором сценариев.

Примечание

Предполагается, что наборы сценариев будут распространяться в составе коллекции.

Синтаксис команды:

```
ansible-creator init [collection|playbook] [-h] [--na] [--lf LOG_FILE] [--ll {notset,
↪debug,info,warning,error,critical}] [--la {true,false}] [--json] [-v] [-f] [-o] [-no]
↪<namespace>.<collection> [<path>]
```

Здесь:

- <namespace> – название пространства имен;
- <collection> – название коллекции;
- <path> – путь к каталогу, в котором Ansible Creator должен разместить файлы и каталоги коллекции.

  Значение по умолчанию: `./` (текущий каталог).

Подробное описание аргументов команды см. *ниже*.

Описание аргументов команд

Команды Ansible Creator поддерживают следующие аргументы:

`--json`
: Вывод сообщений в формате JSON:

  - `true` – включен;
  - `false` – выключен.

  Значение по умолчанию: `false`.

`--la`, `--log-append` `[true|false]`
: Вывод сообщений в файл:

  - `true` – включен;
  - `false` – выключен.

  Значение по умолчанию: `true`.

--lf, **--log-file** <file>
: Путь к файлу для вывода сообщений.

  Значение по умолчанию: ~/ansible-creator.log.

--ll, **--log-level** <level>
: Уровень подробности сообщений в порядке убывания:

  1. notset;
  2. debug;
  3. info;
  4. warning;
  5. error;
  6. critical.

  Значение по умолчанию: notset.

--na, **--no-ansi** [true|false]
: Запрет использования кодов ANSI для подсветки цветов в терминале:

  - true – включен (цветовое оформление не используется);
  - false – выключен.

  Значение по умолчанию: false.

-f, **--force**
: При выполнении команды init Ansible Creator проверяет существование целевого каталога. Если каталог не существует или пуст, Ansible Creator выполняет создание необходимых файлов и каталогов. В противном случае Ansible Creator выводит в терминал запрос на подтверждение переинициализации.

  Эта опция выключает вывод запроса на подтверждение переинициализации каталога.

  Значение по умолчанию: false (для переинициализации требуется подтверждение пользователя).

-h, **--help**
: Вывод справочной информации об использовании Ansible Creator или его команд.

-no, **--no-overwrite**
: Эта опция запрещает перезапись файлов и каталогов при выполнении операций.

  Значение по умолчанию: false (перезапись включена).

-o, **--overwrite**
: Эта опция разрешает перезапись файлов и каталогов при выполнении операций.

  Значение по умолчанию: false (перезапись выключена).

-v, **--verbosity**
: Увеличение уровня детализации сообщений, выводимых в терминал.

  Эта опция обладает свойством аддитивностью – ее можно использовать до трех раз, например:

  ```
  ansible-creator -vvv ...
  ```

  Значение по умолчанию: 0.

17.8.4 Ansible Lint

Утилита Ansible Lint является универсальным инструментом для проверки качества и стиля кода в сценариях, ролях и коллекциях Ansible. Для этого она предоставляет различные аргументы, описание которых приводится далее.

Вызов утилиты имеет следующий вид:

```
ansible-lint [-h] [-P | -L | -T] [-f {brief,full,md,json,codeclimate,quiet,pep8,sarif}]␣
↪[--sarif-file SARIF_FILE] [-q]
             [--profile {min,basic,moderate,safety,shared,production}] [-p] [--project-
↪dir PROJECT_DIR] [-r RULESDIR]
             [-R] [-s] [--fix [WRITE_LIST]] [--show-relpath] [-t TAGS] [-v] [-x SKIP_
↪LIST] [--generate-ignore] [-w WARN_LIST]
             [--enable-list ENABLE_LIST] [--nocolor] [--force-color] [--exclude EXCLUDE_
↪PATHS [EXCLUDE_PATHS ...]]
             [-c CONFIG_FILE] [-i IGNORE_FILE] [--offline] [--version] [lintables ...]
```

Для понимания принципов работы утилиты обратитесь к ее *описанию*.

Аргументы команды

При запуске Ansible Lint можно использовать следующие аргументы:

`lintables`
: Позиционный аргумент, идущий в самом конце и определяющий список файлов (список путей), подлежащих проверке.

`-h`, `--help`
: Вывод справочной информации.

`-P`, `--list-profiles`
: Вывод списка всех профилей. Форматирование вывода не поддерживается.

`-L`, `--list-rules`
: Вывод списка всех правил. Возможные значения:

  - `brief` – краткое описание правил (по умолчанию);
  - `full` – полное описание правил;
  - `md` – формат Markdown.

`-T`, `--list-tags`
: Вывод всех тегов и правил, к которым они относятся. Чтобы включить правила с тегом `opt-in`, необходимо повысить уровень детализации с помощью флага `-v`.

`-f`, `--format`
: Формат вывода:

  - `brief` – краткий;
  - `codeclimate` – совместимый с Code Climate;
  - `full` – полный с выводом детальной информацией о каждой найденной проблеме;
  - `json` – JSON;
  - `md` – Markdown;
  - pep8 – соответствующий стандарту **PEP 8**[254];
  - `quiet`– минимально возможное количество информации, то есть только сообщения об ошибках;
  - `sarif` – SARIF.

  Значение по умолчанию отсутствует.

[254] https://peps.python.org/pep-0008/

--sarif-file
Файл для вывода в формате SARIF.

-q
Уменьшение уровня детализации вывода. Для минимального уровня укажите `-qq`.

--profile
Профиль проверки:

- `min` – минимальные проверки, чтобы Ansible мог загрузить контент;
- `basic` – стандартные правила форматирования и синтаксиса;
- `moderate` – рекомендации по читаемости и поддержке;
- `safety` – правила безопасности для избежания использования опасных модулей;
- `shared` – профиль для подготовки к публикации кода;
- `production` – наивысший уровень строгости.

-p, --parseable
Вывод результата в формате, удобном для разбора, эквивалентен `-f pep8`.

--project-dir
Каталог проекта или репозитория. По умолчанию – каталог, в котором расположен конфигурационный файл `.ansible-lint`.

-r, --rules-dir
Каталоги с пользовательскими правилами. Указанные правила заменяют собой встроенные. Чтобы использовать и встроенные правила, и пользовательские, используйте аргумент `--rules-dir` совместно с аргументом `-R`.

-R
Сохранение встроенных правил при использовании аргумента `-r`.

-s, --strict
Возвращение ненулевого кода выхода для предупреждений, а не только для ошибок.

---fix
Разрешение автоматического исправления проблем, включая форматирование YAML. Можно ограничить список правил, для которых разрешены исправления, указав `all`, `none`, список идентификаторов или тегов.

--show-relpath
Вывод относительного пути к текущему каталогу.

-t, --tags
Использование только тех правил, идентификаторы или теги которых совпадают с указанными.

-v
Повышение уровня детализации вывода. Для максимального уровня укажите `-vv`.

-x, --skip-list
Использование только тех правил, идентификаторы или теги которых не совпадают с указанными, например, `--skip-list=name`.

--generate-ignore
Генерация файла `.ansible-lint-ignore`, который содержит список файлов и правил, которые должны быть пропущены при проверке. Каждая строка содержит путь к файлу и через пробел название или идентификатор правила.

-w, --warn-list
Показ предупреждений только для указанных правил, если это не переопределено в конфигурационном файле. Значение по умолчанию: `experimental, jinja[spacing], fqcn[deep]`.

--enable-list
: Включение дополнительных правил. Позволяет указать идентификаторы или теги дополнительных правил, которые должны быть включены при проверке.

--nocolor
: Отключение цветного вывода (эквивалентно `NO_COLOR=1`).

--force-color
: Принудительное включение цветного вывода (эквивалентно `FORCE_COLOR=1`).

--exclude
: Путь к файлу или каталогу, который следует пропустить (повторяемый параметр).

-c, --config-file
: Конфигурационный файл. По умолчанию ищет файлы `.ansible-lint`, `config/ansible-lint.yml`, или `.config/ansible-lint.yaml`.

-i, --ignore-file
: Файл игнорирования. В файле игнорирования содержится список файлов и правил, которые должны быть пропущены при проверке. Каждая строка содержит путь к файлу и через пробел идентификатор правила. По умолчанию ищет файлы `.ansible-lint-ignore` или `.config/ansible-lint-ignore.txt`.

--offline
: Отключение установки зависимостей из `requirements.yml` и обновления схем.

--version
: Вывод информации о версии Ansible Lint.

Переменные окружения

При запуске Ansible Lint можно использовать следующие переменные окружения:

ANSIBLE_LINT_CUSTOM_RULESDIR
: Путь к каталогам с пользовательскими правилами.

ANSIBLE_LINT_IGNORE_FILE
: Переопределение имени файла игнорирования.

  Значение по умолчанию: `.ansible-lint-ignore`.

ANSIBLE_LINT_WRITE_TMP
: Сохранение исправлений во временные файлы, не изменяя оригиналы. Созданные временные файлы используются для проверки.

ANSIBLE_LINT_SKIP_SCHEMA_UPDATE
: Отключение обновления схем.

ANSIBLE_LINT_NODEPS
: Отключение установки зависимостей и проверок, которые могут завершиться неудачей при отсутствии модулей.

Файл настроек

В файле настроек Ansible Lint можно задать следующие параметры:

profile
: Профиль проверки. Возможные значения параметра приведены в описании аргумента `--profile`. При значении `null`, используется конфигурация, заданная другими параметрами.

  Пример заполнения:

```
profile: null
```

sarif_file

Файл для вывода в формате SARIF.

Пример заполнения:

```
sarif_file: result.sarif
```

exclude_paths

Путь к файлу или каталогу, который следует пропустить (повторяемый параметр).

Пример заполнения:

```
exclude_paths:
  - .cache/
  - test/fixtures/formatting-before/
  - test/fixtures/formatting-prettier/
```

parseable

Включение форматирования, удобного для разбора.

Пример заполнения:

```
parseable: true
```

quiet

Понижение уровня детализации вывода.

Пример заполнения:

```
quiet: true
```

strict

Указание обрабатывать предупреждения как ошибки.

Пример заполнения:

```
strict: true
```

verbosity

Уровень подробности вывода. Если установлено значение 1, вывод будет наименее подробным. Для более детализированного вывода используйте более высокие значения.

Пример заполнения:

```
verbosity: 1
```

mock_modules

Список модулей, которые Ansible Lint должен имитировать для прохождения синтаксической проверки, если они отсутствуют.

Пример заполнения:

```
mock_modules:
  - zuul_return
  - fake_namespace.fake_collection.fake_module
  - fake_namespace.fake_collection.fake_module.fake_submodule
```

mock_roles

Список ролей, которые Ansible Lint должен имитировать для прохождения синтаксической проверки, если они отсутствуют.

Пример заполнения:

```
mock_roles:
  - mocked_role
  - author.role_name
  - fake_namespace.fake_collection.fake_role
```

loop_var_prefix

Определение префиксов для переменных цикла, используемых в ролях. Префикс должен начинаться с __ или названия роли, что предотвращает конфликт названий переменных.

Пример заполнения:

```
loop_var_prefix: "^(__|{role}_)"
```

var_naming_pattern

Регулярное выражение, определяющее допустимый шаблон для названий переменных. В данном примере название переменной должно начинаться с буквы или подчеркивания и содержать только строчные буквы, цифры и подчеркивания:

```
var_naming_pattern: "^[a-z_][a-z0-9_]*$"
```

use_default_rules

Указание необходимости использования встроенных правил Ansible Lint.

Пример заполнения:

```
use_default_rules: true
```

rulesdir

Список путей к каталогам с пользовательскими правилами.

Пример заполнения:

```
rulesdir:
  - ./rule/directory/
```

skip_list

Список тегов правил, которые должны быть проигнорированы. Чтобы правило было пропущено, достаточно совпадения одного тега из списка.

Пример заполнения:

```
skip_list:
  - skip_this_tag
```

enable_list

Список дополнительных правил.

Пример заполнения:

```
enable_list:
  - args
  - empty-string-compare
  - no-log-password
  - no-same-owner
  - name[prefix]
  - galaxy-version-incorrect
  - yaml
```

tags

Ограничение проверки только правилами с указанными тегами. Чтобы правило было выбрано, достаточно, чтобы оно содержало один из указанных тегов.

Пример заполнения:

```
tags:
  - jinja[spacing]
```

warn_list

Список тегов для правил, неудачную проверку которых нужно рассматривать как предупреждение, а не ошибку. Чтобы правило попало в предупреждения, достаточно совпадения одного тега из списка.

Пример заполнения:

```
warn_list:
  - skip_this_tag
  - experimental
```

write_list

Автоматическое исправление кода для соответствия указанным правилам. Возможные значения:

- `all` – разрешает все исправления;
- none – отключает исправления;
- список тегов.

Пример заполнения:

```
write_list:
  - all
```

offline

Отключение установки зависимостей и обновления схем.

Пример заполнения:

```
offline: true
```

extra_vars

Значения обязательных переменных Ansible, используемых в сценариях или ролях, чтобы избежать ошибок анализа, возникающих из-за их отсутствия.

Пример заполнения:

```
extra_vars:
  foo: bar
  multiline_string_variable: |
    line1
    line2
  complex_variable: ":{;\t$()"
```

skip_action_validation

Включение принудительной проверки действий с помощью задач. Обычно это не требуется, потому что проверка синтаксиса покрывает это.

Пример заполнения:

```
skip_action_validation: false
```

kinds

Добавление пользовательских шаблонов для определения типа файла.

Пример заполнения:

```
kinds:
  - yaml: "**/*.yaml-too"
```

only_builtins_allow_collections

Список сторонних коллекций, использование которых разрешено при включенном правиле only_builtins.

Пример заполнения:

```
only_builtins_allow_collections:
  - example_ns.example_collection
```

only_builtins_allow_modules

Список сторонних модулей, использование которых разрешено при включенном правиле only_builtins.

Пример заполнения:

```
only_builtins_allow_modules:
  - example_module
```

task_name_prefix

Определение префикса, который будет добавлен к названиям задач.

Пример заполнения:

```
task_name_prefix: "{stem} | "
```

max_block_depth

Ограничение максимальной вложенности блоков.

Пример заполнения:

```
max_block_depth: 20
```

supported_ansible_also

Добавление поддержки для версий Ansible, которые могут быть не указаны как поддерживаемые по умолчанию.

Пример заполнения:

```
supported_ansible_also:
  - "2.14"
```

17.8.5 Ansible Molecule

Утилита *Ansible Molecule* является универсальным инструментом для разработки и тестирования ролей Ansible. Для этого она предоставляет несколько команд с различными аргументами, описание которых приводится далее.

Вызов утилиты имеет следующий вид:

```
molecule [<options>] <command> [<arguments>]
```

Здесь:

- <options> – аргументы утилиты;
- <command> – команда;
- <arguments> – аргументы команды.

Возможные аргументы утилиты:

--help

Вывод справочной информации о командах и аргументах Ansible Molecule.

--version

Вывод информации о версии Ansible Molecule.

--debug, --no-debug
: Включение или отключение режима отладки. По умолчанию отключен.

-v, --verbose
: Увеличение уровня детализации Ansible.

  Значение по умолчанию: `0`.

--c, --base-config TEXT
: Путь к базовой конфигурации (можно указать несколько раз). Если указан, Ansible Molecule выполнит в указанном порядке слияние настроек и слияние внутри каждого сценария. По умолчанию Ansible Molecule ищет файл `.config/molecule/config.yml` в текущем каталоге, и если не найдет, будет искать в домашнем каталоге пользователя.

-e, --env-file TEXT
: Имя файла с переменными, используемыми при обработке *файла настроек*. Значение по умолчанию: `.env.yml`.

Команды

При запуске Ansible Molecule можно использовать следующие команды.

check

Эта команда позволяет увидеть, что будет сделано, не внося изменения в тестовую среду. Она приводит к выполнению следующих команд:

1. *destroy*;
2. *dependency*;
3. *create*;
4. *prepare*;
5. *converge*.

Синтаксис команды:

```
molecule check [-s, --parallel, --help]
```

Подробное описание аргументов команды см. *ниже*.

cleanup

Эта команда выполняет очистку любых изменений, внесенных в тестовую среду во время выполнения тестов. Обычно используют для возвращения системы в исходное состояние. Синтаксис команды:

```
molecule cleanup [-s, --help]
```

Подробное описание аргументов команды см. *ниже*.

converge

Эта команда настраивает тестовые среды, выполняя следующие команды:

1. *dependency*;
2. *create*;
3. *prepare*;
4. *converge*.

Синтаксис команды:

```
molecule converge [-s, --help]
```

Подробное описание аргументов команды см. *ниже*.

create

Эта команда создает тестовые среды. Синтаксис команды:

```
molecule create [-s, -d, --help]
```

Подробное описание аргументов команды см. *ниже*.

dependency

Эта команда выполняет установку необходимых зависимостей перед выполнением тестирования. Синтаксис команды:

```
molecule dependency [-s, --help]
```

Подробное описание аргументов команды см. *ниже*.

destroy

Эта команда удаляет тестовые среды. Синтаксис команды:

```
molecule destroy [-s, -d, --all, --parallel, --help]
```

Подробное описание аргументов команды см. *ниже*.

drivers

Эта команда отображает список доступных драйверов. Синтаксис команды:

```
molecule drivers [-f, --help]
```

Аргументы команды:

-f, **--format** [simple|plain]
: Формат вывода. Значение по умолчанию: `simple`.

idempotence

Эта команда выполняет настройку тестовых сред и проверку идемпотентности. Синтаксис команды:

```
molecule idempotence [-s, --help]
```

Подробное описание аргументов команды см. *ниже*.

init

Эта команда создает структуру каталогов и файлов, необходимых для работы Ansible Molecule с новой ролью или сценарием. Синтаксис команды:

```
molecule init [--help, scenario] [--dependency-name, -d, --provisioner-name]
```

Подробное описание аргументов команды см. *ниже*.

list

Эта команда показывает статус тестовых сред. Синтаксис команды:

```
molecule list [-s, -f, --help]
```

Аргументы команды:

-f, **--format** [simple|plain]
: Формат вывода. Значение по умолчанию: simple.

Подробное описание аргументов команды см. *ниже*.

login

Эта команда позволяет войти в тестовую среду. Синтаксис команды:

```
molecule login [-h, -s, --help]
```

Подробное описание аргументов команды см. *ниже*.

matrix

Эта команда отображает матрицу шагов, используемых для тестирования. Синтаксис команды:

```
molecule matrix [-s, --help]
```

Подробное описание аргументов команды см. *ниже*.

prepare

Эта команда подготавливает тестовые среды к выполнению тестов. Синтаксис команды:

```
molecule prepare [-s, -d, -f --help]
```

Аргументы команды:

-f, **--force**, **--no-force**
: Включить или отключить режим принудительного выполнения. По умолчанию выключен.

Подробное описание аргументов команды см. *ниже*.

reset

Эта команда возвращает временные каталоги Ansible Molecule в исходное состояние. Синтаксис команды:

```
molecule reset [-s, --help]
```

Подробное описание аргументов команды см. *ниже*.

side-effect

Эта команда позволяет выполнить действия, которые не входят в основное применение роли. Синтаксис команды:

```
molecule side-effect [-s, --help]
```

Подробное описание аргументов команды см. *ниже*.

syntax

Эта команда проверяет синтаксис файлов роли. Синтаксис команды:

```
molecule syntax [-s, --help]
```

Подробное описание аргументов команды см. *ниже*.

test

Эта команда запускает полный цикл тестирования. Синтаксис команды:

```
molecule test [-s, -p, -d, --all, --destroy, --parallel, --help]
```

Подробное описание аргументов команды см. *ниже*.

verify

Эта команда выполняет проверку состояния тестовых сред после тестирования. Синтаксис команды:

```
molecule verify [-s, --help]
```

Подробное описание аргументов команды см. *ниже*.

Описание аргументов команд

Команды Ansible Molecule поддерживают следующие аргументы:

--all / --no-all
: Выполнение команды для всех сценариев. По умолчанию выключено (--no-all).

-d, --driver-name [default|molecule_brest|molecule_yandexcloud|podman]
: Название используемого драйвера. Значение по умолчанию: default.

--destroy [always|never]
: Политика удаления тестовых сред после завершения тестов:
  - always – удалять после каждого теста (по умолчанию);
  - never – не удалять.

-h, --host TEXT
: Узел для подключения.

-p, --platform-name TEXT
: Название платформы для тестирования.

  Значение по умолчанию: None.

--parallel / --no-parallel
: Включение или отключение параллельного режима. По умолчанию отключен.

-s, --scenario-name TEXT
: Название используемого сценария.

  Значение по умолчанию: default.

scenario
: Название инициализированного сценария.

  Значение по умолчанию: default.

  Возможные дополнительные аргументы:

--dependency-name [galaxy]
: Название используемой зависимости.

  Значение по умолчанию: galaxy.

-d, --driver-name [default|molecule_brest|molecule_yandexcloud|podman]
: Название используемого драйвера.

  Значение по умолчанию: default.

--provisioner-name [ansible]
: Название используемого *исполнителя сценариев*.

  Значение по умолчанию: ansible.

Файл настроек

Для настройки Ansible Molecule используется файл molecule.yml. В нем описывается, как и где будет выполнятся тестирование ролей. Он может содержать следующие параметры:

- драйвер;
- зависимости;
- исполнитель сценариев;
- платформы;
- средства проверки;
- сценарии.

Драйвер

Драйвер указывает какой инструмент утилита будет использовать для создания тестовой среды. Драйвер может быть изменен в командной строке. Утилита запоминает последний использованный драйвер и будет использовать его для всех последующих команд или до тех пор, пока не будут удалены тестовые среды.

Пример заполнения:

```
driver:
  options:
    managed: False
    login_cmd_template: "podman exec -ti {instance} bash"
    ansible_connection_options:
      ansible_connection: podman
```

Возможные параметры:

name
: Название используемого драйвера.

options
: Дополнительные параметры для настройки драйвера:

  managed
  : Указывает, должна ли утилита управлять созданием и уничтожением тестовой среды:

    - True – утилита управляет тестовой средой (по умолчанию);
    - False – управление передается внешнему инструменту.

  login_cmd_template
  : Шаблон команды для входа в тестовую среду.

ansible_connection_options

Настройки соединения, которые будут использоваться для взаимодействия с тестовой средой.

Возможные опции:

ansible_connection

Тип соединения.

Зависимости

Для указания пути к файлу зависимостей используются параметры `role-file` и `requirements-file`. Если необходимо указать путь к файлу ролей, то используется параметр `role-file`. Если необходимо установить коллекции, то используйте параметр `requirements-file`.

Пример заполнения:

```
dependency:
  name: galaxy
  options:
    requirements-file: requirements.yml
```

Возможные параметры:

name

Тип зависимости:

- `galaxy` – для загрузки зависимостей из Ansible Galaxy.
- `shell` – для выполнения произвольных shell-команд.

enabled

Указывает, следует ли включить управление зависимостями:

- `True` (по умолчанию);
- `False`.

options

Дополнительные параметры для настройки зависимостей:

ignore-certs

Указывает, следует ли игнорировать ошибки, связанные с сертификатами SSL при загрузке зависимостей:

- `True`;
- `False`.

ignore-errors

Указывает, следует ли игнорировать ошибки, возникающие при загрузке зависимостей:

- `True`;
- `False`.

role-file

Указывает файл, содержащий список ролей, которые необходимо загрузить.

requirements-file

Указывает файл, содержащий список коллекций Ansible, которые нужно установить.

command

Команда, которую нужно выполнить для управления зависимостями. Только для типа зависимости `shell`.

Исполнитель сценариев

Ansible является исполнителем по умолчанию. Утилита управляет жизненным циклом тестовых сред. Однако пользователь может предоставить сценарии для выполнения этапов. Для этого в параметре playbooks необходимо указать путь к файлу, содержащему сценарий выполнения этапа.

Пример заполнения:

```
provisioner:
  name: ansible
  playbooks:
    create: create.yml
    converge: converge.yml
    destroy: destroy.yml
```

Платформы

Платформы определяют тестовые среды и группы, к которым они относятся. Можно использовать несколько тестовых сред.

Пример заполнения:

```
platforms:
  - name: molecule-astra
    image: hub.astra-automation.ru/aa-1.2/aa-minimal-ee
```

Возможные параметры:

name
: Название тестовой платформы.

image
: Образ контейнера, который будет использоваться для платформы.

privileged
: Указывает, нужно ли запускать контейнер в привилегированном режиме.

groups
: Группы, к которым принадлежит платформа.

Средства проверки

Утилита производит тестирование с помощью настраиваемых средств проверки. Ansible является средством проверки по умолчанию.

Пример заполнения:

```
verifier:
  name: testinfra
```

Возможные параметры:

name
: Название средства проверки:

enabled
: Указывает, нужно ли включить тестирование:

  - True;
  - False.

directory
: Путь к тестовому каталогу.

Сценарии

Сценарий определяет этапы, которые будут выполняться в рамках тестирования. Сценарий по умолчанию – `default`.

Пример заполнения:

```
scenario:
  test_sequence:
    - dependency
    - cleanup
    - destroy
    - syntax
    - create
    - prepare
    - converge
    - idempotence
    - side_effect
    - verify
    - cleanup
```

17.8.6 Ansible Test

Важно

Эта часть документации находится в стадии разработки.

17.8.7 Tox Ansible

Утилита Tox Ansible предназначена для упрощения тестирования коллекций Ansible.

Вызов утилиты имеет следующий вид:

```
tox <command> [<arguments>]
```

Здесь:

- <command> – команда;
- <arguments> – аргументы команды.

Для понимания принципов работы утилиты обратитесь к ее *описанию*.

Команды

При запуске Tox Ansible можно использовать следующие команды.

run

Эта основная команда для запуска тестовых окружений.

Синтаксис команды:

```
tox run [-h, --colored, --exit-and-dump-after, -c, --workdir,
        --root, --runner, -v, -q, --result-json, --hashseed, --discover,
        --list-dependencies, --no-list-dependencies, -e, -m, -f, --skip-env,
        -s, -n, -b, --installpkg, --develop, --no-recreate-pkg,
        --skip-pkg-install, --version, --no-provision,
        --no-recreate-provision, -r, -x]
```

Подробное описание аргументов команды см. *ниже*.

run-parallel

Эта команда запускает несколько тестовых окружений одновременно, что ускоряет выполнение тестов.

Синтаксис команды:

```
tox run-parallel [-h, --colored,, --exit-and-dump-after, -c,
                  --workdir, --root, --runner, -v, -q, --result-json,
                  --hashseed, --discover,--list-dependencies,
                  --no-list-dependencies, -e, -m, -f, --skip-env, -s, -n, -b,
                  --installpkg, --develop, --no-recreate-pkg,
                  --skip-pkg-install, -p, -o, --parallel-no-spinner,
                  --version, --no-provision, --no-recreate-provision, -r, -x]
```

Подробное описание аргументов команды см. *ниже*.

depends

Эта команда используется для визуализации зависимостей между окружениями. Для ее использования в файле настроек должны быть указаны зависимости.

Синтаксис команды:

```
tox depends [-h, --colored, --exit-and-dump-after, -c,
             --workdir, --root, --runner, -v, -q, --result-json,
             --hashseed, --discover, --list-dependencies,
             --no-list-dependencies, --version, --no-provision,
             --no-recreate-provision, -r, -x]
```

Подробное описание аргументов команды см. *ниже*.

list

Эта команда выводит все доступные окружения, определенные в файле настроек.

Синтаксис команды:

```
tox list [-h, --colored, --exit-and-dump-after, -c, --workdir,
          --root, --runner, -v, -q, --result-json, --hashseed, --discover,
          --list-dependencies, --no-list-dependencies, --no-desc, -m, -f,
          --skip-env, -d, --version, --no-provision, --no-recreate-provision,
          -r, -x]
```

Подробное описание аргументов команды см. *ниже*.

devenv

Эта команда создает окружение разработки и включает специфичные параметры для этого процесса.

Синтаксис команды:

```
tox devenv [-h, --colored, --exit-and-dump-after, -c,
            --workdir, --root, --runner, -v, -q, --result-json,
            --hashseed, --discover, --list-dependencies,
            --no-list-dependencies, -e, --skip-env, --no-recreate-pkg,
            --version, --no-provision, --no-recreate-provision, -r, -x]
```

Подробное описание аргументов команды см. *ниже*.

config

Эта команда показывает настройки Tox.

Синтаксис команды:

```
tox config [-h, --colored, --exit-and-dump-after, -c,
           --workdir, --root, --runner, -v, -q, --result-json,
           --hashseed, --discover, --list-dependencies,
           --no-list-dependencies, -k, --core, -e, -m, -f, --skip-env,
           -s, --develop, --no-recreate-pkg, --version, --no-provision,
           --no-recreate-provision, -r, -x]
```

Подробное описание аргументов команды см. *ниже*.

quickstart

Эта команда создает минимальный файл настроек.

Синтаксис команды:

```
tox quickstart [-h, --colored, --exit-and-dump-after, -c,
               --workdir, --root, --runner, -v, -q, --result-json,
               --hashseed, --discover, --list-dependencies,
               --no-list-dependencies, --version, --no-provision,
               --no-recreate-provision, -r, -x] [root]
```

Подробное описание аргументов команды см. *ниже*.

[root]
: Путь к каталогу, в котором необходимо создать файл настроек.

exec

Эта команда позволяет запустить любую команду внутри виртуального окружения.

Синтаксис команды:

```
tox exec [-h, --colored, --exit-and-dump-after, -c, --workdir,
         --root, --runner, -v, -q, --result-json, --hashseed, --discover,
         --list-dependencies, --no-list-dependencies, -e, --skip-env, -s, -n,
         -b, --installpkg, --develop, --no-recreate-pkg, --skip-pkg-install,
         --version, --no-provision, --no-recreate-provision, -r, -x]
```

Подробное описание аргументов команды см. *ниже*.

Аргументы команды

При запуске Tox Ansible можно использовать следующие аргументы:

-h
: Вывод справочной информации.

--colored {yes,no}
: Использование цветного вывода:

  - yes – включить цветной вывод (по умолчанию);
  - no – отключить цветной вывод.

--exit-and-dump-after <seconds>
: Остановка выполнения тестов через определенное количество секунд <seconds>. При значении 0 (по умолчанию) выполнение тестов не останавливается до их завершения.

-c file
: Путь к файлу настроек.

--workdir dir
: Рабочий каталог.

  Значение по умолчанию: путь к каталогу с файлом настроек.

--root dir
: Корневой каталог проекта.

  Значение по умолчанию: путь к каталогу с файлом настроек.

--runner {virtualenv}
: Механизм создания виртуальных окружений.

-v
: Повышение уровня детализации вывода.

-q
: Уменьшение уровня детализации вывода.

--result-json path
: Сохранение результатов выполнения в JSON-файл.

--hashseed SEED
: Установка фиксированного значения для переменной PYTHONHASHSEED.

--discover path [path ...]
: Интерпретаторы Python, которые необходимо использовать при тестировании.

--list-dependencies
: Вывод списка зависимостей, установленных в окружении.

--no-list-dependencies
: Запрет вывода списка зависимостей, установленных в окружении.

-s [v], **--skip-missing-interpreters** [v]
: Поведение утилиты при отсутствии в системе интерпретатора Python, который указан в файле настроек:

  - true – пропустить тестовые окружения, для которых не найден соответствующий интерпретатор Python, и продолжить выполнение без фиксации ошибки;
  - false – завершить работу с сообщением об ошибке;
  - config – использовать настройку skip_missing_interpreters из файла настроек (по умолчанию).

-n, **--notest**
: Не запускать тестовые команды.

-b, **--pkg-only**, **--sdistonly**
: Создание дистрибутивного пакета проекта установки в тестовое окружение.

--installpkg INSTALL_PKG
: Дистрибутивный пакет, который необходимо установить в тестовое окружение.

--develop
: Установка дистрибутивного пакета в режиме разработки.

--no-recreate-pkg
: Запрет пересоздания окружения, в котором собирается дистрибутивный пакет.

--skip-pkg-install
: Не устанавливать дистрибутив в тестовом окружении.

--version
: Вывод информации о версии Tox Ansible.

--no-provision [REQ_JSON]
: Отключение стадии подготовки. Если указан путь к файлу, Tox Ansible не будет создавать виртуальные окружения, но запишет метаданные в формате JSON в указанный файл.

--no-recreate-provision
: Отмена действия аргумента `-r`.

-r
: Пересоздание виртуальных окружений при каждом запуске.

-x OVERRIDE
: Переопределение настроек для конкретного окружения.

--matrix-scope {all,sanity,integration,unit}
: Определение тестов, которые будут включены в матрицу для GitHub Actions:
  - `all` – все тесты;
  - `sanity` – тесты для проверки работоспособности;
  - `integration` – интеграционные тесты;
  - `unit` – модульные тесты.

--gh-matrix
: Генерация матрицы для GitHub Actions.

--ansible
: Включение поддержки тестирования Ansible.

-e ENV
: Окружение для проведения тестирования.

-m label [label ...]
: Использование окружений с определенными метками.

-f factor [factor ...]
: Использование окружений с определенными факторами (частями названий).

--skip-env <re>
: Исключение из тестирования окружений, названия которых совпадают с заданным регулярным выражением <re>.

-d
: Вывод только окружений по умолчанию.

--no-desc
: Вывод окружений без их описания.

-p VAL
: Количество параллельных процессов при выполнении тестов.

-o, **--parallel-live**
: Вывод результатов тестов в реальном времени.

--parallel-no-spinner
: Отключение анимации загрузки.

-k key [key ...]
: Вывод только указанных параметров конфигурации.

--core
: Вывод базовых параметров конфигурации.

Файл настроек

Для настройки Tox Ansible используется файл `tox-ansible.ini`. В нем указываются параметры окружений и используемых средств тестирования.

Основные настройки

`[tox]` это основная секция файла настроек, в ней указываются следующие параметры:

envlist

Список окружений, которые будут созданы и запущены. Для названий окружений рекомендуется использовать следующий шаблон:

```
<type>-py<VERSION>-ansible<VERSION>
```

Здесь:

- <type> – тип теста;
- py<VERSION> – версия Python, которая будет использоваться в окружении;
- ansible<VERSION> – версия Ansible или Ansible Core, которая будет установлена.

Пример заполнения:

```
[tox]
envlist =
    unit-py37-ansible2.9,
    sanity-py37-ansible2.10
```

skip-missing-interpreters

Поведение утилиты при отсутствии в системе интерпретатора Python, который указан в файле настроек:

- `true` – пропустить тестовые окружения, для которых не найден соответствующий интерпретатор Python, и продолжить выполнение без фиксации ошибки;
- `false` – завершить работу с ошибкой.

Пример заполнения:

```
[tox]
skip_missing_interpreters = true
```

Настройка окружений

Настройки окружений в Tox Ansible делятся на две категории:

- глобальные – применяются ко всем окружениям и задаются в секции `[testenv]`;
- индивидуальные – задаются для каждого тестового окружения отдельно в секциях вида `[testenv:<env_name>]`.

Поддерживаются следующие параметры:

allowlist_externals

Список разрешенных системных команд.

Пример заполнения:

```
allowlist_externals = echo
```

basepython:

Версия Python, которую необходимо использовать.

Пример заполнения:

```
[testenv]
basepython =
    py37: python3.7
```

В данном примере окружения с префиксом py37 будут использовать Python 3.7.

deps

Зависимости, которые необходимо установить.

Пример заполнения:

```
[testenv]
deps =
    ansible2.9: ansible==2.9.*
```

В данном примере Python-пакет `ansible` версии `2.9.*` будет установлен в окружениях, названия которых содержат строку `ansible2.9`.

commands

Команды, которые необходимо выполнить в окружении.

Пример заполнения:

```
[testenv]
commands =
    echo "Тип теста: {envname}, версия Python: {basepython}, версия Ansible: {deps}"
```

В результате выполнения данной команды будет выведена информация об окружении.

17.8.8 Ansible Pytest

Ansible Pytest предоставляет вспомогательный код, позволяющий упростить написание и поддержку тестов.

ansible_adhoc

Вспомогательный код `ansible_adhoc` управляет объектом `HostManager`, позволяющим взаимодействовать с узлами инвентаря.

Основные возможности:

- Инициализация инвентарного списка.
- Управление параметрами инвентарного списка.

  По умолчанию вспомогательный код использует параметры, переданные в аргументах CLI при запуске `pytest`, но также позволяет настраивать параметры инвентаря в коде теста.
- Вызов модулей и команд Ansible.

  Объект `HostManager` предоставляет методы, позволяющие вызывать модули Ansible для выполнения команд на отдельных узлах и группах узлов.

ansible_facts

Вспомогательный код `ansible_facts` предоставляет факты Ansible в формате JSON для узлов, указанных в инвентаре. К фактам относятся сведения об оборудовании, версии ОС и ее настройках, доменное имя узла, время его работы с момента последней загрузки и тому подобное. Факты могут быть полезны при принятии решений о выполнении или пропуске определенных тестов, например, на определенных платформах или в определенных конфигурациях.

ansible_module

Вспомогательный код ansible_module предназначен для вызова модулей Ansible внутри тестов без дополнительной настройки инвентарного списка или подключения. Главное отличие от вспомогательного кода ansible_adhoc в том, что используются параметры, указанные при запуске pytest.

localhost

Вспомогательный код localhost предназначен для выполнения модулей Ansible на том же узле, где запускается pytest. Он предоставляет объект ModuleDispatcher, который не требует настройки подключения или инвентарного списка.

17.8.9 Ansible Sign

Утилита Ansible Sign является универсальным инструментом для создания и проверки цифровой подписи содержимого Ansible. Для этого она предоставляет различные аргументы, описание которых приводится далее.

Вызов утилиты имеет следующий вид:

```
ansible-sign [-h] [--version] [--debug] [--nocolor] CONTENT_TYPE
```

Для понимания принципов работы утилиты обратитесь к ее *описанию*.

17.8.10 Экспорт коллекций из образа

Для быстрого экспорта набора коллекций, проверенных или сертифицированных для использования в Astra Automation, в состав образа среды исполнения aa-full-ee входят следующие компоненты:

- набор коллекций;
- утилита export_collections.py для загрузки этих коллекций в приватный реестр.

Утилита принимает две команды:

- list – просмотр доступных коллекций;
- publish – публикация коллекций.

Синтаксис вызова утилиты выглядит следующим образом:

```
podman run aa-full-ee:latest python3 export_collections.py [arguments] <command>␣
↪[command_arguments]
```

Здесь:

- [arguments] – общие аргументы;
- <command> – команда;
- [command_arguments] – аргументы команды.

Общие аргументы

Утилита поддерживает следующие аргументы, доступные для использования с любой из команд:

-d <dir>, **--collections-dir** <dir>

Путь к каталогу, в котором хранятся коллекции Ansible.

Значение аргумента также может быть передано через переменную окружения COLLECTIONS_DIR.

Значение по умолчанию: /usr/share/ansible/collections/ansible_collections/.

-h, **--help**

Вывод справки.

Команды

Утилита поддерживает следующие команды:

list
: Вывод списка доступных для публикации коллекций.

publish
: Публикация выбранных коллекций в реестр коллекций.

  Поддерживаемые аргументы:

  - `--all` – публикация всех доступных коллекций.
  - `--autocreate` – автоматическое создание пространства имен (namespace). Если аргумент не задан, то пространство имен должно быть создано вручную заранее. В противном случае работа утилиты завершится с ошибкой.
  - `--collection <namespace.collection>` – название одной коллекции для публикации.
  - `--namespace <namespace>` – пространство имен для публикации всех содержащихся в нем коллекций.
  - `-s <url>`, `--galaxy-url <url>` – IP-адрес или доменное имя реестра коллекций. Значение аргумента также может быть передано через переменную окружения `ANSIBLE_GALAXY_SERVER_GALAXY_URL`.

    Обязательный аргумент, если переменная окружения не задана.
  - `-t <token>`, `--validation-token <token>` – токен доступа к API реестра коллекций. Значение аргумента также может быть передано через переменную окружения `ANSIBLE_GALAXY_SERVER_VALIDATED_TOKEN`.

    Обязательный аргумент, если переменная окружения не задана.

Примеры

Для изучения возможностей утилиты ознакомьтесь с приведенными ниже примерами.

Список доступных коллекций

Следующий пример демонстрирует вывод списка коллекций из стандартного каталога `/usr/share/ansible/collections/ansible_collections/`:

```
podman run aa-full-ee:latest python3 export_collections.py list
```

Список коллекций из пользовательского каталога

Следующий пример демонстрирует вывод списка коллекций из указанного каталога:

```
podman run aa-full-ee:latest python3 export_collections.py -d /data/collections list
```

Публикация одной коллекции

Следующий пример демонстрирует публикацию одной коллекции в Private Automation Hub:

```
podman run \
    -e ANSIBLE_GALAXY_SERVER_GALAXY_URL=hub.example.org \
    -e ANSIBLE_GALAXY_SERVER_VALIDATED_TOKEN=TOKEN123 \
    aa-full-ee:latest \
    python3 export_collections.py publish --collection astra.ald_pro
```

Публикация всех коллекций в пространстве имен

Следующий пример демонстрирует публикацию всех коллекций в указанном пространстве имен:

```
podman run \
    -e ANSIBLE_GALAXY_SERVER_GALAXY_URL=hub.example.org \
    -e ANSIBLE_GALAXY_SERVER_VALIDATED_TOKEN=TOKEN123 \
    aa-full-ee:latest \
    python3 export_collections.py publish --namespace astra
```

Публикация всех доступных коллекций

Следующий пример демонстрирует публикацию всех доступных коллекций в Private Automation Hub:

```
podman run \
    -e ANSIBLE_GALAXY_SERVER_GALAXY_URL=hub.example.org \
    -e ANSIBLE_GALAXY_SERVER_VALIDATED_TOKEN=TOKEN123 \
    aa-full-ee:latest \
    python3 export_collections.py publish --all
```

17.8.11 Образы служебных контейнеров

Образы, созданные ПАО Группа Астра, доступны для загрузки из Automation Hub[255].

Образы aa-minimal-ee

1.0.4

Пакеты APT

Пакет	Версия	Описание
adduser	3.134+astra.se10	add and remove users a
ansible-core	2.15.10-1	Configuration managem
apt	2.6.1+ci202312061551+astra2+b1	commandline package m
apt-transport-https	2.6.1+ci202312061551+astra2+b1	transitional package for
astra-archive-keyring	2023.12	GnuPG archive keys of t
astra-version	8.1.41+v1.8.1.12	Update Astra version
base-files	12.4+deb12u1.astra6	Debian base system mis
base-passwd	3.6.1+b1	Debian base system ma
bash	5.2.15-2.astra.se1+b2	GNU Bourne Again SHell
bsdutils	1:2.38.1-5+deb12u1+astra13+b1	basic utilities from 4.4BS
ca-certificates	20230311+b2	Common CA certificates
coreutils	9.1-1+ci202402121122+astra2+b1	GNU core utilities
curl	7.88.1-10+deb12u5+b2	command line tool for tr
dash	0.5.12-2.astra1	POSIX-compliant shell
debconf	1.5.82+b1	Debian configuration ma
debianutils	5.7-0.5~deb12u1+b1	Miscellaneous utilities sp
diffutils	1:3.8-4+b1	File comparison utilities
dirmngr	2.2.40-1.1.astra6+b2	GNU privacy guard - net
dpkg	1.21.22.astra.se3+b2	Debian package manage
e2fsprogs	1.47.0-2+b1	ext2/ext3/ext4 file syste
expect	5.45.4-2+ci202403211247+astra1+b1	Automates interactive a
findutils	4.9.0-4+b1	utilities for finding files–
gawk	1:5.2.1-2+b1	GNU awk, a pattern scan
gcc-12-base:amd64	12.2.0-14.astra3+b1	GCC, the GNU Compiler
gnupg	2.2.40-1.1.astra6+b2	GNU privacy guard - a fr

[255] https://hub.astra-automation.ru/

Таблица 1 – продолжение с предыдущей с

Пакет	Версия	Описание
gnupg-l10n	2.2.40-1.1.astra6+b2	GNU privacy guard - loc
gnupg-utils	2.2.40-1.1.astra6+b2	GNU privacy guard - util
gpg	2.2.40-1.1.astra6+b2	GNU Privacy Guard – mi
gpg-agent	2.2.40-1.1.astra6+b2	GNU privacy guard - cry
gpg-wks-client	2.2.40-1.1.astra6+b2	GNU privacy guard - We
gpg-wks-server	2.2.40-1.1.astra6+b2	GNU privacy guard - We
gpgconf	2.2.40-1.1.astra6+b2	GNU privacy guard - cor
gpgsm	2.2.40-1.1.astra6+b2	GNU privacy guard - S/M
gpgv	2.2.40-1.1.astra6+b2	GNU privacy guard - sig
grep	3.8-5+b1	GNU grep, egrep and fg
gzip	1.12-1+b1	GNU compression utilitie
hostname	3.23+nmu1+b1	utility to set/show the h
ieee-data	20220827.1	OUI and IAB listings
init-system-helpers	1.65.2+b1	helper tools for all init sy
libacl1:amd64	2.3.1-3+b1	access control list - shar
libapt-pkg6.0:amd64	2.6.1+ci202312061551+astra2+b1	package management r
libassuan0:amd64	2.5.5-5+b1	IPC library for the GnuP
libattr1:amd64	1:2.5.1-4+ci202403112058+astra3+b1	extended attribute hand
libaudit-common	1:3.0.9-1+ci202401241435+astra3+b2	Dynamic library for secu
libaudit1:amd64	1:3.0.9-1+ci202401241435+astra3+b2	Dynamic library for secu
libblkid1:amd64	2.38.1-5+deb12u1+astra13+b1	block device ID library
libbrotli1:amd64	1.0.9-2+b1	library implementing br
libbsd0:amd64	0.11.7-2+b1	utility functions from BS
libbz2-1.0:amd64	1.0.8-5+b1	high-quality block-sortin
libc-bin	2.36-9+deb12u7+ci202405171200+astra5+b1	GNU C Library: Binaries
libc6:amd64	2.36-9+deb12u7+ci202405171200+astra5+b1	GNU C Library: Shared li
libcap-ng0:amd64	0.8.3-1+b2	alternate POSIX capabili
libcap2:amd64	1:2.66-4+b1	POSIX 1003.1e capabilit
libcbor0.8:amd64	0.8.0-2+b1	library for parsing and g
libcom-err2:amd64	1.47.0-2+b1	common error descriptio
libcrypt1:amd64	1:4.4.33-2+b1	libcrypt shared library
libcurl4:amd64	7.88.1-10+deb12u5+b2	easy-to-use client-side U
libdb5.3:amd64	5.3.28+dfsg2-1+ci202311141605+astra1+b1	Berkeley v5.3 Database
libdebconfclient0:amd64	0.270astra1+b1	Debian Configuration Ma
libedit2:amd64	3.1-20221030-2+b1	BSD editline and history
libexpat1:amd64	2.5.0-1+ci202405221303+astra2	XML parsing C library - r
libext2fs2:amd64	1.47.0-2+b1	ext2/ext3/ext4 file syste
libffi8:amd64	3.4.4-1+b1	Foreign Function Interfa
libfido2-1:amd64	1.12.0-2+b2	library for generating ar
libfile-find-rule-perl	0.34-3+b1	module to search for file
libgcc-s1:amd64	12.2.0-14.astra3+b1	GCC support library
libgcrypt20:amd64	1.10.1-3.astra2+b1	LGPL Crypto library - run
libgdbm-compat4:amd64	1.23-3+b1	GNU dbm database rout
libgdbm6:amd64	1.23-3+b1	GNU dbm database rout
libgmp10:amd64	2:6.2.1+dfsg1-1.1+b1	Multiprecision arithmeti
libgnutls30:amd64	3.7.9-2+deb12u2+b1	GNU TLS library - main r
libgost	2.0.2-5+ci4+b1	This package contains th
libgpg-error0:amd64	1.46-1+b1	GnuPG development run
libgssapi-krb5-2:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerberos runtime lib
libhogweed6:amd64	3.8.1-2+b1	low level cryptographic
libidn2-0:amd64	2.3.3-1+b1	Internationalized domai
libk5crypto3:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerberos runtime lib
libkeyutils1:amd64	1.6.3-2+b1	Linux Key Management
libkrb5-3:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerberos runtime lib
libkrb5support0:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerberos runtime lib
libksba8:amd64	1.6.3-2+b1	X.509 and CMS support
libldap-2.5-0:amd64	2.5.13+dfsg-5+ci202405061914+astra2	OpenLDAP libraries

Таблица 1 – продолжение с предыдущей с

Пакет	Версия	Описание
liblocale-gettext-perl	1.07-5+b1	module using libc functi
liblz4-1:amd64	1.9.4-1+b1	Fast LZ compression alg
liblzma5:amd64	5.4.1-0.2+b1	XZ-format compression
libmd0:amd64	1.0.4-2+b1	message digest function
libmount1:amd64	2.38.1-5+deb12u1+astra13+b1	device mounting library
libmpfr6:amd64	4.2.0-1+b1	multiple precision floati
libncursesw6:amd64	6.4-4+b1	shared libraries for term
libnettle8:amd64	3.8.1-2+b1	low level cryptographic
libnghttp2-14:amd64	1.52.0-1+deb12u1+ci1+b1	library implementing HT
libnpth0:amd64	1.6-3+b1	replacement for GNU Pt
libnsl2:amd64	1.3.0-2+b1	Public client interface fo
libnumber-compare-perl	0.03-3+b1	module for performing n
libp11-kit0:amd64	0.24.1-2+b1	library for loading and c
libpam-modules:amd64	1.5.2-6+deb12u1.astra.se21+b1	Pluggable Authenticatio
libpam-modules-bin	1.5.2-6+deb12u1.astra.se21+b1	Pluggable Authenticatio
libpam-runtime	1.5.2-6+deb12u1.astra.se21+b1	Runtime support for the
libpam0g:amd64	1.5.2-6+deb12u1.astra.se21+b1	Pluggable Authenticatio
libparsec-base3	3.9+ci132	base library for PARSEC
libparsec-cap3	3.9+ci132	capabilities library for PA
libpcre2-8-0:amd64	10.42-1+b1	New Perl Compatible Re
libpdp	3.9+ci132	library for PARSEC DP se
libperl5.36:amd64	5.36.0-7+deb12u1+ci202404171352+astra2	shared Perl library
libproc2-0:amd64	2:4.0.2-3+b1	library for accessing pro
libpsl5:amd64	0.21.2-1+b1	Library for Public Suffix
libpython3-stdlib:amd64	3.11.2-1+b1	interactive high-level ob
libpython3.11-minimal:amd64	3.11.2-6astra4+b1	Minimal subset of the Py
libpython3.11-stdlib:amd64	3.11.2-6astra4+b1	Interactive high-level ob
libreadline8:amd64	8.2-1.3+b1	GNU readline and histor
librtmp1:amd64	2.4+20151223.gitfa8646d.1-2+b1	toolkit for RTMP streams
libsasl2-2:amd64	2.1.28+dfsg-10+b2	Cyrus SASL - authentica
libsasl2-modules-db:amd64	2.1.28+dfsg-10+b2	Cyrus SASL - pluggable
libseccomp2:amd64	2.5.4-1+b2	high level interface to Li
libselinux1:amd64	3.4-1+b1	SELinux runtime shared
libsemanage-common	3.4-1+b1	Common files for SELinu
libsemanage2:amd64	3.4-1+b1	SELinux policy manager
libsepol2:amd64	3.4-2.1+b1	SELinux library for mani
libsigsegv2:amd64	2.14-1+b1	Library for handling pag
libsmartcols1:amd64	2.38.1-5+deb12u1+astra13+b1	smart column output ali
libsodium23:amd64	1.0.18-1+b1	Network communication
libsqlite3-0:amd64	3.40.1-2+ci202405281356+astra2	SQLite 3 shared library
libss2:amd64	1.47.0-2+b1	command-line interface
libssh2-1:amd64	1.10.0-3+b2	SSH2 client-side library
libssl3:amd64	3.2.0-2-astra5+ci1	Secure Sockets Layer to
libstdc++6:amd64	12.2.0-14.astra3+b1	GNU Standard C++ Libr
libsystemd0:amd64	252.17-1~deb12u1astra.se1+ci13+b1	systemd utility library
libtasn1-6:amd64	4.19.0-2+b1	Manage ASN.1 structure
libtcl8.6:amd64	8.6.13+dfsg-2+b1	Tcl (the Tool Command L
libtext-charwidth-perl:amd64	0.04-11+b1	get display widths of ch
libtext-glob-perl	0.11-3+b1	Perl module for matchin
libtext-iconv-perl:amd64	1.7-8+b1	module to convert betw
libtext-wrapi18n-perl	0.06-10+b1	internationalized substit
libtinfo6:amd64	6.4-4+b1	shared low-level terminf
libtirpc-common	1.3.3+ds-1+b1	transport-independent R
libtirpc3:amd64	1.3.3+ds-1+b1	transport-independent R
libudev1:amd64	252.17-1~deb12u1astra.se1+ci13+b1	libudev shared library
libunistring2:amd64	1.0-2+b1	Unicode string library fo
libuuid1:amd64	2.38.1-5+deb12u1+astra13+b1	Universally Unique ID lib

Таблица 1 – продолжение с предыдущей с

Пакет	Версия	Описание
libxxhash0:amd64	0.8.1-1+b1	shared library for xxhas
libyaml-0-2:amd64	0.2.5-1+b1	Fast YAML 1.1 parser an
libzstd1:amd64	1.5.4+dfsg2-5+b1	fast lossless compressio
login	1:4.13+dfsg1-1.astra.se10+b1	system login tools
logsave	1.47.0-2+b1	save the output of a cor
lsb-base	11.6	transitional package for
mawk	1.3.4.20200120-3.1+b1	Pattern scanning and te
media-types	10.0.0	List of standard media t
mount	2.38.1-5+deb12u1+astra13+b1	tools for mounting and r
ncurses-base	6.4-4+b1	basic terminal type defi
ncurses-bin	6.4-4+b1	terminal-related prograr
netbase	6.4	Basic TCP/IP networking
openssh-client	1:9.6p1-2~deb10u1astra8se5	secure shell (SSH) client
openssl	3.2.0-2-astra5+ci1	Secure Sockets Layer to
passwd	1:4.13+dfsg1-1.astra.se10+b1	change and administer p
perl	5.36.0-7+deb12u1+ci202404171352+astra2	Larry Wall’s Practical Ex
perl-base	5.36.0-7+deb12u1+ci202404171352+astra2	minimal Perl system
perl-modules-5.36	5.36.0-7+deb12u1+ci202404171352+astra2	Core Perl modules
pinentry-curses	1.2.1-1+b1	curses-based PIN or pas
procps	2:4.0.2-3+b1	/proc file system utilities
python3	3.11.2-1+b1	interactive high-level ob
python3-bcrypt	3.2.2-1+b1	password hashing librar
python3-certifi	2022.9.24-1	root certificates for valid
python3-cffi-backend:amd64	1.15.1-5+b1	Foreign Function Interfa
python3-chardet	5.1.0+dfsg-2	Universal Character Enc
python3-charset-normalizer	3.0.1-2	charset, encoding and la
python3-cryptography	38.0.4-3+ci202405171208+astra3+b1	Python library exposing
python3-daemon	2.3.2-1	library for making a Unix
python3-distutils	3.11.2-3+b1	distutils package for Pyt
python3-dnspython	2.3.0-1+b1	DNS toolkit for Python 3
python3-httplib2	0.20.4-3	comprehensive HTTP cli
python3-idna	3.3-1	Python IDNA2008 (RFC 5
python3-jinja2	3.1.2-1	small but fast and easy
python3-lib2to3	3.11.2-3+b1	Interactive high-level ob
python3-lockfile	1:0.12.2-2.2	file locking library for Py
python3-markupsafe	2.1.2-1+b1	HTML/XHTML/XML string
python3-minimal	3.11.2-1+b1	minimal subset of the Py
python3-nacl	1.5.0-2+b1	Python bindings to libso
python3-netaddr	0.8.0-2	manipulation of various
python3-ntlm-auth	1.4.0-2	NTLM low-level Python l
python3-packaging	23.0-1	core utilities for python3
python3-paramiko	2.12.0-2+b1	Make ssh v2 connection
python3-pexpect	4.8.0-4	Python 3 module for aut
python3-pip	23.0.1+dfsg-1	Python package installe
python3-pkg-resources	66.1.1-1	Package Discovery and
python3-ptyprocess	0.7.0-5	Run a subprocess in a ps
python3-pyparsing	3.0.9-1	alternative to creating a
python3-requests	2.28.1+dfsg-1+b1	elegant and simple HTTP
python3-requests-ntlm	1.1.0-3+b1	Adds support for NTLM a
python3-resolvelib	0.9.0-2	module to resolve abstra
python3-setuptools	66.1.1-1	Python3 Distutils Enhan
python3-six	1.16.0-4	Python 2 and 3 compati
python3-urllib3	1.26.12-1+ci202406111901+astra2	HTTP library with thread
python3-wheel	0.38.4-2	built-package format for
python3-winrm	0.3.0-4+deb12u1	Python 3 library for Win
python3-xmltodict	0.13.0-1	Makes working with XML
python3-yaml	6.0-3+b1	YAML parser and emitte

Таблица 1 – продолжение с предыдущей с

Пакет	Версия	Описание
python3.11	3.11.2-6astra4+b1	Interactive high-level ob
python3.11-ansible-runner	2.4.0+aa1.1.0	Consistent Ansible Pytho
python3.11-minimal	3.11.2-6astra4+b1	Minimal subset of the Py
python3.11-pypsrp	0.8.1+aa1.1.0	PowerShell Remoting Pr
python3.11-pyspnego	0.11.0+aa1.1.0	Windows Negotiate Auth
python3.11-requests	2.32.3+aa1.1.0	Python HTTP for Human
python3.11-urllib3	2.2.3+aa1.1.0	HTTP library with thread
readline-common	8.2-1.3+b1	GNU readline and histor
sed	4.9-1+b1	GNU stream editor for fi
sensible-utils	0.0.17+nmu1	Utilities for sensible alte
sshpass	1.09-1+b1	Non-interactive ssh pass
sysvinit-utils	3.06-4+b1	System-V-like utilities
tar	1.34+dfsg-1.2+deb12u1+b1	GNU version of the tar a
tcl-expect:amd64	5.45.4-2+ci202403211247+astra1+b1	Automates interactive a
tcl8.6	8.6.13+dfsg-2+b1	Tcl (the Tool Command L
tzdata	2024a-4	time zone and daylight-s
usrmerge	37~deb12u1+b1	Convert the system to th
util-linux	2.38.1-5+deb12u1+astra13+b1	miscellaneous system u
util-linux-extra	2.38.1-5+deb12u1+astra13+b1	interactive login tools
zlib1g:amd64	1:1.2.13.dfsg-1+b1	compression library - ru

Пакеты Python

Пакет	Версия
ansible-core	2.15.10
ansible-runner	2.4.0
bcrypt	3.2.2
certifi	2022.9.24
chardet	5.1.0
charset-normalizer	3.0.1
cryptography	38.0.4
dnspython	2.3.0
dumb-init	1.2.5
httplib2	0.20.4
idna	3.3
Jinja2	3.1.2
lockfile	0.12.2
MarkupSafe	2.1.2
netaddr	0.8.0
ntlm-auth	1.4.0
packaging	23.0
paramiko	2.12.0
pexpect	4.8.0
ptyprocess	0.7.0
PyNaCl	1.5.0
pyparsing	3.0.9
pypsrp @ file:///docker_share/dist/pypsrp-0.8.1-py3-none-any.whl	
pyspnego	0.11.0
python-daemon	2.3.2
pywinrm	0.3.0
PyYAML	6.0
requests	2.32.3
requests-ntlm	1.1.0
resolvelib	0.9.0
six	1.16.0

продолжается на следующей странице

Таблица 2 – продолжение с предыдущей страницы

Пакет	Версия
urllib3	2.2.3
xmltodict	0.13.0

Образы aa-full-ee

0.3.0

Пакеты APT

Пакет	Версия	Описание
adduser	3.134+astra.se10	add and re
ansible-compat	24.9.1+aa1.1.0	Ansible co
ansible-core	2.15.10-1	Configurati
ansible-lint	24.9.2+aa1.1.0	Checks pla
ansible-sign	0.1.1+aa1.1.0	Ansible co
apt	2.6.1+ci202312061551+astra2+b1	commandli
apt-transport-https	2.6.1+ci202312061551+astra2+b1	transitional
astra-archive-keyring	2023.12	GnuPG arc
astra-version	8.1.41+v1.8.1.12	Update Ast
base-files	12.4+deb12u1.astra6	Debian bas
base-passwd	3.6.1+b1	Debian bas
bash	5.2.15-2.astra.se1+b2	GNU Bourn
binutils	2.40-2+b1	GNU assem
binutils-common:amd64	2.40-2+b1	Common fi
binutils-x86-64-linux-gnu	2.40-2+b1	GNU binary
bsdutils	1:2.38.1-5+deb12u1+astra13+b1	basic utiliti
ca-certificates	20230311+b2	Common C
coreutils	9.1-1+ci202402121122+astra2+b1	GNU core u
cpp	4:12.2.0-3+b1	GNU C prep
cpp-12	12.2.0-14.astra3+b1	GNU C prep
curl	7.88.1-10+deb12u5+b2	command l
dash	0.5.12-2.astra1	POSIX-com
debconf	1.5.82+b1	Debian con
debianutils	5.7-0.5~deb12u1+b1	Miscellaneo
diffutils	1:3.8-4+b1	File compar
dirmngr	2.2.40-1.1.astra6+b2	GNU privac
dpkg	1.21.22.astra.se3+b2	Debian pac
e2fsprogs	1.47.0-2+b1	ext2/ext3/e
expect	5.45.4-2+ci202403211247+astra1+b1	Automates
findutils	4.9.0-4+b1	utilities for
gawk	1:5.2.1-2+b1	GNU awk, a
gcc	4:12.2.0-3+b1	GNU C com
gcc-12	12.2.0-14.astra3+b1	GNU C com
gcc-12-base:amd64	12.2.0-14.astra3+b1	GCC, the G
git	1:2.39.2-1.1+ci3	fast, scalab
git-man	1:2.39.2-1.1+ci3	fast, scalab
gnupg	2.2.40-1.1.astra6+b2	GNU privac
gnupg-l10n	2.2.40-1.1.astra6+b2	GNU privac
gnupg-utils	2.2.40-1.1.astra6+b2	GNU privac
gpg	2.2.40-1.1.astra6+b2	GNU Privac
gpg-agent	2.2.40-1.1.astra6+b2	GNU privac
gpg-wks-client	2.2.40-1.1.astra6+b2	GNU privac
gpg-wks-server	2.2.40-1.1.astra6+b2	GNU privac
gpgconf	2.2.40-1.1.astra6+b2	GNU privac
gpgsm	2.2.40-1.1.astra6+b2	GNU privac
gpgv	2.2.40-1.1.astra6+b2	GNU privac

Таблица 3 – продолж

Пакет	Версия	Описание
grep	3.8-5+b1	GNU grep,
gzip	1.12-1+b1	GNU compr
hostname	3.23+nmu1+b1	utility to se
ieee-data	20220827.1	OUI and IAB
init-system-helpers	1.65.2+b1	helper tools
libacl1:amd64	2.3.1-3+b1	access cont
libapt-pkg6.0:amd64	2.6.1+ci202312061551+astra2+b1	package ma
libasan8:amd64	12.2.0-14.astra3+b1	AddressSan
libassuan0:amd64	2.5.5-5+b1	IPC library f
libatomic1:amd64	12.2.0-14.astra3+b1	support libr
libattr1:amd64	1:2.5.1-4+ci202403112058+astra3+b1	extended a
libaudit-common	1:3.0.9-1+ci202401241435+astra3+b2	Dynamic lib
libaudit1:amd64	1:3.0.9-1+ci202401241435+astra3+b2	Dynamic lib
libbinutils:amd64	2.40-2+b1	GNU binary
libblkid1:amd64	2.38.1-5+deb12u1+astra13+b1	block devic
libbrotli1:amd64	1.0.9-2+b1	library impl
libbsd0:amd64	0.11.7-2+b1	utility funct
libbz2-1.0:amd64	1.0.8-5+b1	high-quality
libc-bin	2.36-9+deb12u7+ci202405171200+astra5+b1	GNU C Libra
libc-dev-bin	2.36-9+deb12u7+ci202405171200+astra5+b1	GNU C Libra
libc6:amd64	2.36-9+deb12u7+ci202405171200+astra5+b1	GNU C Libra
libc6-dev:amd64	2.36-9+deb12u7+ci202405171200+astra5+b1	GNU C Libra
libcap-ng0:amd64	0.8.3-1+b2	alternate P
libcap2:amd64	1:2.66-4+b1	POSIX 1003
libcbor0.8:amd64	0.8.0-2+b1	library for p
libcc1-0:amd64	12.2.0-14.astra3+b1	GCC cc1 plu
libcom-err2:amd64	1.47.0-2+b1	common er
libcrypt-dev:amd64	1:4.4.33-2+b1	libcrypt dev
libcrypt1:amd64	1:4.4.33-2+b1	libcrypt sha
libctf-nobfd0:amd64	2.40-2+b1	Compact C
libctf0:amd64	2.40-2+b1	Compact C
libcurl3-gnutls:amd64	7.88.1-10+deb12u5+b2	easy-to-use
libcurl4:amd64	7.88.1-10+deb12u5+b2	easy-to-use
libdb5.3:amd64	5.3.28+dfsg2-1+ci202311141605+astra1+b1	Berkeley v5
libdebconfclient0:amd64	0.270astra1+b1	Debian Con
libedit2:amd64	3.1-20221030-2+b1	BSD editlin
liberror-perl	0.17029-2+b1	Perl module
libexpat1:amd64	2.5.0-1+ci202405221303+astra2	XML parsin
libexpat1-dev:amd64	2.5.0-1+ci202405221303+astra2	XML parsin
libext2fs2:amd64	1.47.0-2+b1	ext2/ext3/e
libffi8:amd64	3.4.4-1+b1	Foreign Fun
libfido2-1:amd64	1.12.0-2+b2	library for g
libfile-find-rule-perl	0.34-3+b1	module to s
libgcc-12-dev:amd64	12.2.0-14.astra3+b1	GCC suppo
libgcc-s1:amd64	12.2.0-14.astra3+b1	GCC suppo
libgcrypt20:amd64	1.10.1-3.astra2+b1	LGPL Crypt
libgdbm-compat4:amd64	1.23-3+b1	GNU dbm d
libgdbm6:amd64	1.23-3+b1	GNU dbm d
libgmp10:amd64	2:6.2.1+dfsg1-1.1+b1	Multiprecisi
libgnutls30:amd64	3.7.9-2+deb12u2+b1	GNU TLS lib
libgomp1:amd64	12.2.0-14.astra3+b1	GCC OpenM
libgost	2.0.2-5+ci4+b1	This packag
libgpg-error0:amd64	1.46-1+b1	GnuPG dev
libgprofng0:amd64	2.40-2+b1	GNU Next G
libgssapi-krb5-2:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerber
libhogweed6:amd64	3.8.1-2+b1	low level cr
libidn2-0:amd64	2.3.3-1+b1	Internation

Таблица 3 – продолж

Пакет	Версия	Описание
libisl23:amd64	0.25-1.1+b1	manipulatin
libitm1:amd64	12.2.0-14.astra3+b1	GNU Transa
libjansson4:amd64	2.14-2+b1	C library fo
libjs-jquery	3.6.1+dfsg+~3.5.14-1+b1	JavaScript l
libjs-sphinxdoc	5.3.0-4+b1	JavaScript s
libjs-underscore	1.13.4~dfsg+~1.11.4-3	JavaScript's
libk5crypto3:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerber
libkeyutils1:amd64	1.6.3-2+b1	Linux Key M
libkrb5-3:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerber
libkrb5support0:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerber
libksba8:amd64	1.6.3-2+b1	X.509 and C
libldap-2.5-0:amd64	2.5.13+dfsg-5+ci202405061914+astra2	OpenLDAP
liblocale-gettext-perl	1.07-5+b1	module usi
liblsan0:amd64	12.2.0-14.astra3+b1	LeakSanitiz
liblz4-1:amd64	1.9.4-1+b1	Fast LZ com
liblzma5:amd64	5.4.1-0.2+b1	XZ-format c
libmd0:amd64	1.0.4-2+b1	message di
libmount1:amd64	2.38.1-5+deb12u1+astra13+b1	device mou
libmpc3:amd64	1.3.1-1+b1	multiple pr
libmpfr6:amd64	4.2.0-1+b1	multiple pr
libncursesw6:amd64	6.4-4+b1	shared libra
libnettle8:amd64	3.8.1-2+b1	low level cr
libnghttp2-14:amd64	1.52.0-1+deb12u1+ci1+b1	library impl
libnpth0:amd64	1.6-3+b1	replacemen
libnsl-dev:amd64	1.3.0-2+b1	libnsl devel
libnsl2:amd64	1.3.0-2+b1	Public clien
libnumber-compare-perl	0.03-3+b1	module for
libp11-kit0:amd64	0.24.1-2+b1	library for l
libpam-modules:amd64	1.5.2-6+deb12u1.astra.se21+b1	Pluggable A
libpam-modules-bin	1.5.2-6+deb12u1.astra.se21+b1	Pluggable A
libpam-runtime	1.5.2-6+deb12u1.astra.se21+b1	Runtime su
libpam0g:amd64	1.5.2-6+deb12u1.astra.se21+b1	Pluggable A
libparsec-base3	3.9+ci132	base library
libparsec-cap3	3.9+ci132	capabilities
libpcre2-8-0:amd64	10.42-1+b1	New Perl Co
libpdp	3.9+ci132	library for P
libperl5.36:amd64	5.36.0-7+deb12u1+ci202404171352+astra2	shared Perl
libpq-dev	15.6-astra.se2+b1	header files
libpq5:amd64	15.6-astra.se2+b1	PostgreSQL
libproc2-0:amd64	2:4.0.2-3+b1	library for a
libprotobuf32:amd64	3.21.12-3+b1	protocol bu
libpsl5:amd64	0.21.2-1+b1	Library for
libpython3-dev:amd64	3.11.2-1+b1	header files
libpython3-stdlib:amd64	3.11.2-1+b1	interactive
libpython3.11:amd64	3.11.2-6astra4+b1	Shared Pyth
libpython3.11-dev:amd64	3.11.2-6astra4+b1	Header files
libpython3.11-minimal:amd64	3.11.2-6astra4+b1	Minimal sub
libpython3.11-stdlib:amd64	3.11.2-6astra4+b1	Interactive
libquadmath0:amd64	12.2.0-14.astra3+b1	GCC Quad-
libreadline8:amd64	8.2-1.3+b1	GNU readlin
librtmp1:amd64	2.4+20151223.gitfa8646d.1-2+b1	toolkit for R
libsasl2-2:amd64	2.1.28+dfsg-10+b2	Cyrus SASL
libsasl2-modules-db:amd64	2.1.28+dfsg-10+b2	Cyrus SASL
libseccomp2:amd64	2.5.4-1+b2	high level i
libselinux1:amd64	3.4-1+b1	SELinux run
libsemanage-common	3.4-1+b1	Common fil
libsemanage2:amd64	3.4-1+b1	SELinux po

Таблица 3 – продолж

Пакет	Версия	Описание
libsepol2:amd64	3.4-2.1+b1	SELinux libr
libsigsegv2:amd64	2.14-1+b1	Library for
libsmartcols1:amd64	2.38.1-5+deb12u1+astra13+b1	smart colur
libsodium23:amd64	1.0.18-1+b1	Network co
libsqlite3-0:amd64	3.40.1-2+ci202405281356+astra2	SQLite 3 sh
libss2:amd64	1.47.0-2+b1	command-l
libssh2-1:amd64	1.10.0-3+b2	SSH2 client
libssl-dev:amd64	3.2.0-2-astra5+ci1	Secure Soc
libssl3:amd64	3.2.0-2-astra5+ci1	Secure Soc
libstdc++6:amd64	12.2.0-14.astra3+b1	GNU Standa
libsystemd0:amd64	252.17-1~deb12u1astra.se1+ci13+b1	systemd ut
libtasn1-6:amd64	4.19.0-2+b1	Manage AS
libtcl8.6:amd64	8.6.13+dfsg-2+b1	Tcl (the Too
libtext-charwidth-perl:amd64	0.04-11+b1	get display
libtext-glob-perl	0.11-3+b1	Perl module
libtext-iconv-perl:amd64	1.7-8+b1	module to c
libtext-wrapi18n-perl	0.06-10+b1	internationa
libtinfo6:amd64	6.4-4+b1	shared low-
libtirpc-common	1.3.3+ds-1+b1	transport-in
libtirpc-dev:amd64	1.3.3+ds-1+b1	transport-in
libtirpc3:amd64	1.3.3+ds-1+b1	transport-in
libtsan2:amd64	12.2.0-14.astra3+b1	ThreadSani
libubsan1:amd64	12.2.0-14.astra3+b1	UBSan – un
libudev1:amd64	252.17-1~deb12u1astra.se1+ci13+b1	libudev sha
libunistring2:amd64	1.0-2+b1	Unicode str
libuuid1:amd64	2.38.1-5+deb12u1+astra13+b1	Universally
libxxhash0:amd64	0.8.1-1+b1	shared libra
libyaml-0-2:amd64	0.2.5-1+b1	Fast YAML 1
libzstd1:amd64	1.5.4+dfsg2-5+b1	fast lossles
linux-libc-dev:amd64	5.4.0-54astra7+ci98	latest linux
linux-libc-dev-6.1.90-1:amd64	6.1.90-1.astra2+ci15	Linux Kerne
login	1:4.13+dfsg1-1.astra.se10+b1	system logi
logsave	1.47.0-2+b1	save the ou
lsb-base	11.6	transitional
mawk	1.3.4.20200120-3.1+b1	Pattern sca
media-types	10.0.0	List of stan
mount	2.38.1-5+deb12u1+astra13+b1	tools for mo
ncurses-base	6.4-4+b1	basic termi
ncurses-bin	6.4-4+b1	terminal-re
netbase	6.4	Basic TCP/I
openssh-client	1:9.6p1-2~deb10u1astra8se5	secure shel
openssl	3.2.0-2-astra5+ci1	Secure Soc
passwd	1:4.13+dfsg1-1.astra.se10+b1	change and
perl	5.36.0-7+deb12u1+ci202404171352+astra2	Larry Wall's
perl-base	5.36.0-7+deb12u1+ci202404171352+astra2	minimal Pe
perl-modules-5.36	5.36.0-7+deb12u1+ci202404171352+astra2	Core Perl m
pinentry-curses	1.2.1-1+b1	curses-base
procps	2:4.0.2-3+b1	/proc file sy
python3	3.11.2-1+b1	interactive
python3-attr	22.2.0-1	Attributes w
python3-bcrypt	3.2.2-1+b1	password h
python3-certifi	2022.9.24-1	root certific
python3-cffi	1.15.1-5+b1	Foreign Fun
python3-cffi-backend:amd64	1.15.1-5+b1	Foreign Fun
python3-chardet	5.1.0+dfsg-2	Universal C
python3-charset-normalizer	3.0.1-2	charset, en
python3-click	8.1.3-2	Wrapper ar

Таблица 3 – продолж

Пакет	Версия	Описание
python3-colorama	0.4.6-2	Cross-platfo
python3-cryptography	38.0.4-3+ci202405171208+astra3+b1	Python libra
python3-daemon	2.3.2-1	library for r
python3-dev	3.11.2-1+b1	header files
python3-distlib	0.3.6-1	low-level co
python3-distutils	3.11.2-3+b1	distutils pa
python3-dnspython	2.3.0-1+b1	DNS toolkit
python3-filelock	3.9.0-1	platform in
python3-httplib2	0.20.4-3	comprehen
python3-idna	3.3-1	Python IDN
python3-importlib-metadata	4.12.0-1	library to a
python3-jinja2	3.1.2-1	small but fa
python3-jsonschema	4.10.3-1	An(other) i
python3-lib2to3	3.11.2-3+b1	Interactive
python3-lockfile	1:0.12.2-2.2	file locking
python3-markdown-it	2.1.0-5	Python port
python3-markupsafe	2.1.2-1+b1	HTML/XHTM
python3-mdurl	0.1.2-1	Python port
python3-minimal	3.11.2-1+b1	minimal su
python3-more-itertools	8.10.0-2	library with
python3-mypy-extensions	0.4.3-4	Experiment
python3-nacl	1.5.0-2+b1	Python bind
python3-netaddr	0.8.0-2	manipulatio
python3-ntlm-auth	1.4.0-2	NTLM low-l
python3-openssl	23.0.0-1+b1	Python 3 w
python3-packaging	23.0-1	core utilitie
python3-paramiko	2.12.0-2+b1	Make ssh v
python3-pathspec	0.11.0-1	utility librar
python3-pexpect	4.8.0-4	Python 3 m
python3-pip	23.0.1+dfsg-1	Python pac
python3-pkg-resources	66.1.1-1	Package Di
python3-platformdirs	2.6.0-1	determining
python3-ply	3.11-5+b1	Lex and Yac
python3-protobuf	3.21.12-3+b1	Python 3 bi
python3-ptyprocess	0.7.0-5	Run a subp
python3-pycparser	2.21-1	C parser in
python3-pygments	2.14.0+dfsg-1	syntax high
python3-pyparsing	3.0.9-1	alternative
python3-pyrsistent:amd64	0.18.1-1+b1	persistent/f
python3-requests	2.28.1+dfsg-1+b1	elegant and
python3-requests-ntlm	1.1.0-3+b1	Adds suppo
python3-resolvelib	0.9.0-2	module to r
python3-rich	13.3.1-1	render rich
python3-ruamel.yaml.clib:amd64	0.2.7-1+b1	C version o
python3-setuptools	66.1.1-1	Python3 Di
python3-six	1.16.0-4	Python 2 ar
python3-urllib3	1.26.12-1+ci202406111901+astra2	HTTP library
python3-wheel	0.38.4-2	built-packa
python3-winrm	0.3.0-4+deb12u1	Python 3 lib
python3-xmltodict	0.13.0-1	Makes work
python3-yaml	6.0-3+b1	YAML parse
python3-zipp	1.0.0-6	pathlib-com
python3.11	3.11.2-6astra4+b1	Interactive
python3.11-ansible-runner	2.4.0+aa1.1.0	Consistent
python3.11-backports.ssl-match-hostname	3.7.0.1+aa1.1.0	The ssl.mat
python3.11-black	24.10.0+aa1.1.0	The uncom
python3.11-bracex	2.5.post1+aa1.1.0	Bash style

Таблица 3 – продолж

Пакет	Версия	Описание
python3.11-cryptography	42.0.8+aa1.1.0	cryptograph
python3.11-dev	3.11.2-6astra4+b1	Header files
python3.11-googleapis-common-protos	1.65.0+aa1.1.0	Common pr
python3.11-grpcio	1.67.1+aa1.1.0	HTTP/2-bas
python3.11-minimal	3.11.2-6astra4+b1	Minimal sub
python3.11-pika	1.3.2+aa1.1.0	Pika Python
python3.11-protobuf	4.25.5+aa1.1.0	Protocol Bu
python3.11-pyjwt	2.9.0+aa1.1.0	JSON Web T
python3.11-pypsrp	0.8.1+aa1.1.0	PowerShell
python3.11-pyspnego	0.11.0+aa1.1.0	Windows N
python3.11-python-gnupg	0.5.2+aa1.1.0	A wrapper f
python3.11-requests	2.32.3+aa1.1.0	Python HTT
python3.11-ruamel.yaml	0.18.6+aa1.1.0	ruamel.yam
python3.11-subprocess-tee	0.4.2+aa1.1.0	subprocess
python3.11-urllib3	2.2.3+aa1.1.0	HTTP librar
python3.11-wcmatch	10.0+aa1.1.0	Wildcard/gl
python3.11-yamllint	1.35.1+aa1.1.0	A linter for
python3.11-yandexcloud	0.324.0+aa1.1.0	The Yandex
readline-common	8.2-1.3+b1	GNU readlin
rpcsvc-proto	1.4.3-1+b1	RPC protoc
sed	4.9-1+b1	GNU stream
sensible-utils	0.0.17+nmu1	Utilities for
sshpass	1.09-1+b1	Non-interac
sysvinit-utils	3.06-4+b1	System-V-li
tar	1.34+dfsg-1.2+deb12u1+b1	GNU versio
tcl-expect:amd64	5.45.4-2+ci202403211247+astra1+b1	Automates
tcl8.6	8.6.13+dfsg-2+b1	Tcl (the Too
tzdata	2024a-4	time zone a
usrmerge	37~deb12u1+b1	Convert the
util-linux	2.38.1-5+deb12u1+astra13+b1	miscellaneo
util-linux-extra	2.38.1-5+deb12u1+astra13+b1	interactive
zlib1g:amd64	1:1.2.13.dfsg-1+b1	compressio
zlib1g-dev:amd64	1:1.2.13.dfsg-1+b1	compressio

Пакеты Python

Пакет
ansible-compat
ansible-core
ansible-lint @ file:///docker_share/dist/ansible_lint-24.9.2-py3-none-any.whl
ansible-runner
ansible-sign @ file:///docker_share/dist/ansible_sign-0.1.1-py3-none-any.whl
attrs
backports.ssl_match_hostname @ file:///docker_share/dist/backports.ssl_match_hostname-3.7.0.1-py2.py3
bcrypt
black
bracex
certifi
cffi
chardet
charset-normalizer
click
colorama
cryptography
distlib

продолжается

Таблица 4 – продолжение с предыдущей страницы

Пакет
dnspython
docker
dumb-init
filelock
googleapis-common-protos
grpcio
httplib2
idna
importlib-metadata
Jinja2
jsonschema
lockfile
markdown-it-py
MarkupSafe
mdurl
more-itertools
mypy-extensions
netaddr
ntlm-auth
packaging
paramiko
pathspec
pexpect
pika @ file:///docker_share/dist/pika-1.3.2-py3-none-any.whl
platformdirs
ply
protobuf
psycopg2-binary
ptyprocess
pycparser
Pygments
PyJWT
PyNaCl
pyOpenSSL
pyparsing
pypsrp @ file:///docker_share/dist/pypsrp-0.8.1-py3-none-any.whl
pyrsistent
pyspnego
python-daemon
python-gnupg
pywinrm
PyYAML
requests
requests-ntlm
resolvelib
rich
ruamel.yaml
ruamel.yaml.clib
six
subprocess-tee
textfsm
ttp
urllib3
wcmatch
xmltodict
yamllint
yandexcloud

продолжается

Таблица 4 – продолжение с предыдущей страницы

Пакет
zipp

Коллекции Ansible

Коллекция	Версия
ansible.posix	1.5.4
ansible.utils	3.0.0
astra.aa_controller	0.14.2
astra.ald_pro	3.0.0
astra.astralinux	0.5.0
astra.brest	4.0.3
astra.ceph	3.0.3
astra.chrony	0.2.1
astra.cups	1.3.4
astra.dcimanager	0.2.1
astra.dhcp	2.0.2
astra.docker	4.0.1
astra.etcd	0.2.1
astra.freeipa	0.4.1
astra.grafana	2.0.1
astra.haproxy	2.0.1
astra.hardening	0.3.1
astra.iscsi	0.4.2
astra.keepalived	0.3.0
astra.keycloak	2.0.2
astra.memcached	3.0.1
astra.nfs	0.4.1
astra.nginx	1.8.4
astra.ocfs2	0.2.1
astra.patroni	0.2.1
astra.pgbouncer	0.2.1
astra.postgresql	3.0.1
astra.prometheus	2.1.1
astra.rabbitmq	2.0.1
astra.repo_mirror	0.2.1
astra.rubackup	0.7.1
astra.rupost	0.16.1
astra.termidesk	0.10.2
astra.yandexcloud	0.4.0
community.docker	4.1.0
community.general	7.5.0
community.library_inventory_filtering_v1	1.0.2
community.postgresql	2.3.2
community.rabbitmq	1.2.3
freeipa.ansible_freeipa	1.10.0

0.2.0

Пакеты APT

Пакет	Версия	Описание
adduser	3.134+astra.se10	add and re
ansible-compat	24.9.1+aa1.1.0	Ansible co

Таблица 6 – продолж

Пакет	Версия	Описание
ansible-core	2.15.10-1	Configurati
ansible-lint	24.9.2+aa1.1.0	Checks play
ansible-sign	0.1.1+aa1.1.0	Ansible con
apt	2.6.1+ci202312061551+astra2+b1	commandli
apt-transport-https	2.6.1+ci202312061551+astra2+b1	transitional
astra-archive-keyring	2023.12	GnuPG arch
astra-version	8.1.41+v1.8.1.12	Update Ast
base-files	12.4+deb12u1.astra6	Debian bas
base-passwd	3.6.1+b1	Debian bas
bash	5.2.15-2.astra.se1+b2	GNU Bourn
binutils	2.40-2+b1	GNU assem
binutils-common:amd64	2.40-2+b1	Common fil
binutils-x86-64-linux-gnu	2.40-2+b1	GNU binary
bsdutils	1:2.38.1-5+deb12u1+astra13+b1	basic utiliti
ca-certificates	20230311+b2	Common CA
coreutils	9.1-1+ci202402121122+astra2+b1	GNU core u
cpp	4:12.2.0-3+b1	GNU C prep
cpp-12	12.2.0-14.astra3+b1	GNU C prep
curl	7.88.1-10+deb12u5+b2	command l
dash	0.5.12-2.astra1	POSIX-comp
debconf	1.5.82+b1	Debian con
debianutils	5.7-0.5~deb12u1+b1	Miscellaneo
diffutils	1:3.8-4+b1	File compar
dirmngr	2.2.40-1.1.astra6+b2	GNU privac
dpkg	1.21.22.astra.se3+b2	Debian pac
e2fsprogs	1.47.0-2+b1	ext2/ext3/e
expect	5.45.4-2+ci202403211247+astra1+b1	Automates
findutils	4.9.0-4+b1	utilities for
gawk	1:5.2.1-2+b1	GNU awk, a
gcc	4:12.2.0-3+b1	GNU C com
gcc-12	12.2.0-14.astra3+b1	GNU C com
gcc-12-base:amd64	12.2.0-14.astra3+b1	GCC, the G
git	1:2.39.2-1.1+ci3	fast, scalab
git-man	1:2.39.2-1.1+ci3	fast, scalab
gnupg	2.2.40-1.1.astra6+b2	GNU privac
gnupg-l10n	2.2.40-1.1.astra6+b2	GNU privac
gnupg-utils	2.2.40-1.1.astra6+b2	GNU privac
gpg	2.2.40-1.1.astra6+b2	GNU Privac
gpg-agent	2.2.40-1.1.astra6+b2	GNU privac
gpg-wks-client	2.2.40-1.1.astra6+b2	GNU privac
gpg-wks-server	2.2.40-1.1.astra6+b2	GNU privac
gpgconf	2.2.40-1.1.astra6+b2	GNU privac
gpgsm	2.2.40-1.1.astra6+b2	GNU privac
gpgv	2.2.40-1.1.astra6+b2	GNU privac
grep	3.8-5+b1	GNU grep,
gzip	1.12-1+b1	GNU compr
hostname	3.23+nmu1+b1	utility to se
ieee-data	20220827.1	OUI and IAB
init-system-helpers	1.65.2+b1	helper tools
libacl1:amd64	2.3.1-3+b1	access cont
libapt-pkg6.0:amd64	2.6.1+ci202312061551+astra2+b1	package ma
libasan8:amd64	12.2.0-14.astra3+b1	AddressSan
libassuan0:amd64	2.5.5-5+b1	IPC library f
libatomic1:amd64	12.2.0-14.astra3+b1	support libr
libattr1:amd64	1:2.5.1-4+ci202403112058+astra3+b1	extended a
libaudit-common	1:3.0.9-1+ci202401241435+astra3+b2	Dynamic lib
libaudit1:amd64	1:3.0.9-1+ci202401241435+astra3+b2	Dynamic lib

Таблица 6 – продолж

Пакет	Версия	Описание
libbinutils:amd64	2.40-2+b1	GNU binary
libblkid1:amd64	2.38.1-5+deb12u1+astra13+b1	block devic
libbrotli1:amd64	1.0.9-2+b1	library impl
libbsd0:amd64	0.11.7-2+b1	utility funct
libbz2-1.0:amd64	1.0.8-5+b1	high-quality
libc-bin	2.36-9+deb12u7+ci202405171200+astra5+b1	GNU C Libra
libc-dev-bin	2.36-9+deb12u7+ci202405171200+astra5+b1	GNU C Libra
libc6:amd64	2.36-9+deb12u7+ci202405171200+astra5+b1	GNU C Libra
libc6-dev:amd64	2.36-9+deb12u7+ci202405171200+astra5+b1	GNU C Libra
libcap-ng0:amd64	0.8.3-1+b2	alternate P
libcap2:amd64	1:2.66-4+b1	POSIX 1003
libcbor0.8:amd64	0.8.0-2+b1	library for p
libcc1-0:amd64	12.2.0-14.astra3+b1	GCC cc1 plu
libcom-err2:amd64	1.47.0-2+b1	common er
libcrypt-dev:amd64	1:4.4.33-2+b1	libcrypt dev
libcrypt1:amd64	1:4.4.33-2+b1	libcrypt sha
libctf-nobfd0:amd64	2.40-2+b1	Compact C
libctf0:amd64	2.40-2+b1	Compact C
libcurl3-gnutls:amd64	7.88.1-10+deb12u5+b2	easy-to-use
libcurl4:amd64	7.88.1-10+deb12u5+b2	easy-to-use
libdb5.3:amd64	5.3.28+dfsg2-1+ci202311141605+astra1+b1	Berkeley v5
libdebconfclient0:amd64	0.270astra1+b1	Debian Con
libedit2:amd64	3.1-20221030-2+b1	BSD editlin
liberror-perl	0.17029-2+b1	Perl module
libexpat1:amd64	2.5.0-1+ci202405221303+astra2	XML parsin
libexpat1-dev:amd64	2.5.0-1+ci202405221303+astra2	XML parsin
libext2fs2:amd64	1.47.0-2+b1	ext2/ext3/e
libffi8:amd64	3.4.4-1+b1	Foreign Fun
libfido2-1:amd64	1.12.0-2+b2	library for g
libfile-find-rule-perl	0.34-3+b1	module to s
libgcc-12-dev:amd64	12.2.0-14.astra3+b1	GCC suppo
libgcc-s1:amd64	12.2.0-14.astra3+b1	GCC suppo
libgcrypt20:amd64	1.10.1-3.astra2+b1	LGPL Crypt
libgdbm-compat4:amd64	1.23-3+b1	GNU dbm d
libgdbm6:amd64	1.23-3+b1	GNU dbm d
libgmp10:amd64	2:6.2.1+dfsg1-1.1+b1	Multiprecisi
libgnutls30:amd64	3.7.9-2+deb12u2+b1	GNU TLS lib
libgomp1:amd64	12.2.0-14.astra3+b1	GCC OpenM
libgost	2.0.2-5+ci4+b1	This packag
libgpg-error0:amd64	1.46-1+b1	GnuPG dev
libgprofng0:amd64	2.40-2+b1	GNU Next G
libgssapi-krb5-2:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerberc
libhogweed6:amd64	3.8.1-2+b1	low level cr
libidn2-0:amd64	2.3.3-1+b1	Internationa
libisl23:amd64	0.25-1.1+b1	manipulatin
libitm1:amd64	12.2.0-14.astra3+b1	GNU Transa
libjansson4:amd64	2.14-2+b1	C library fo
libjs-jquery	3.6.1+dfsg+~3.5.14-1+b1	JavaScript l
libjs-sphinxdoc	5.3.0-4+b1	JavaScript s
libjs-underscore	1.13.4~dfsg+~1.11.4-3	JavaScript's
libk5crypto3:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerberc
libkeyutils1:amd64	1.6.3-2+b1	Linux Key M
libkrb5-3:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerberc
libkrb5support0:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerberc
libksba8:amd64	1.6.3-2+b1	X.509 and C
libldap-2.5-0:amd64	2.5.13+dfsg-5+ci202405061914+astra2	OpenLDAP
liblocale-gettext-perl	1.07-5+b1	module usi

Таблица 6 – продолж

Пакет	Версия	Описание
liblsan0:amd64	12.2.0-14.astra3+b1	LeakSanitiz
liblz4-1:amd64	1.9.4-1+b1	Fast LZ com
liblzma5:amd64	5.4.1-0.2+b1	XZ-format c
libmd0:amd64	1.0.4-2+b1	message di
libmount1:amd64	2.38.1-5+deb12u1+astra13+b1	device mou
libmpc3:amd64	1.3.1-1+b1	multiple pr
libmpfr6:amd64	4.2.0-1+b1	multiple pr
libncursesw6:amd64	6.4-4+b1	shared libra
libnettle8:amd64	3.8.1-2+b1	low level cr
libnghttp2-14:amd64	1.52.0-1+deb12u1+ci1+b1	library impl
libnpth0:amd64	1.6-3+b1	replacemen
libnsl-dev:amd64	1.3.0-2+b1	libnsl devel
libnsl2:amd64	1.3.0-2+b1	Public clien
libnumber-compare-perl	0.03-3+b1	module for
libp11-kit0:amd64	0.24.1-2+b1	library for l
libpam-modules:amd64	1.5.2-6+deb12u1.astra.se21+b1	Pluggable A
libpam-modules-bin	1.5.2-6+deb12u1.astra.se21+b1	Pluggable A
libpam-runtime	1.5.2-6+deb12u1.astra.se21+b1	Runtime su
libpam0g:amd64	1.5.2-6+deb12u1.astra.se21+b1	Pluggable A
libparsec-base3	3.9+ci132	base library
libparsec-cap3	3.9+ci132	capabilities
libpcre2-8-0:amd64	10.42-1+b1	New Perl Co
libpdp	3.9+ci132	library for P
libperl5.36:amd64	5.36.0-7+deb12u1+ci202404171352+astra2	shared Perl
libpq-dev	15.6-astra.se2+b1	header files
libpq5:amd64	15.6-astra.se2+b1	PostgreSQL
libproc2-0:amd64	2:4.0.2-3+b1	library for a
libprotobuf32:amd64	3.21.12-3+b1	protocol bu
libpsl5:amd64	0.21.2-1+b1	Library for
libpython3-dev:amd64	3.11.2-1+b1	header files
libpython3-stdlib:amd64	3.11.2-1+b1	interactive
libpython3.11:amd64	3.11.2-6astra4+b1	Shared Pyth
libpython3.11-dev:amd64	3.11.2-6astra4+b1	Header files
libpython3.11-minimal:amd64	3.11.2-6astra4+b1	Minimal sub
libpython3.11-stdlib:amd64	3.11.2-6astra4+b1	Interactive
libquadmath0:amd64	12.2.0-14.astra3+b1	GCC Quad-p
libreadline8:amd64	8.2-1.3+b1	GNU readlin
librtmp1:amd64	2.4+20151223.gitfa8646d.1-2+b1	toolkit for R
libsasl2-2:amd64	2.1.28+dfsg-10+b2	Cyrus SASL
libsasl2-modules-db:amd64	2.1.28+dfsg-10+b2	Cyrus SASL
libseccomp2:amd64	2.5.4-1+b2	high level i
libselinux1:amd64	3.4-1+b1	SELinux run
libsemanage-common	3.4-1+b1	Common fil
libsemanage2:amd64	3.4-1+b1	SELinux po
libsepol2:amd64	3.4-2.1+b1	SELinux libr
libsigsegv2:amd64	2.14-1+b1	Library for
libsmartcols1:amd64	2.38.1-5+deb12u1+astra13+b1	smart colum
libsodium23:amd64	1.0.18-1+b1	Network co
libsqlite3-0:amd64	3.40.1-2+ci202405281356+astra2	SQLite 3 sh
libss2:amd64	1.47.0-2+b1	command-l
libssh2-1:amd64	1.10.0-3+b2	SSH2 client
libssl-dev:amd64	3.2.0-2-astra5+ci1	Secure Soc
libssl3:amd64	3.2.0-2-astra5+ci1	Secure Soc
libstdc++6:amd64	12.2.0-14.astra3+b1	GNU Standa
libsystemd0:amd64	252.17-1~deb12u1astra.se1+ci13+b1	systemd ut
libtasn1-6:amd64	4.19.0-2+b1	Manage AS
libtcl8.6:amd64	8.6.13+dfsg-2+b1	Tcl (the Too

Таблица 6 – продолж

Пакет	Версия	Описание
libtext-charwidth-perl:amd64	0.04-11+b1	get display
libtext-glob-perl	0.11-3+b1	Perl module
libtext-iconv-perl:amd64	1.7-8+b1	module to c
libtext-wrapi18n-perl	0.06-10+b1	internationa
libtinfo6:amd64	6.4-4+b1	shared low-
libtirpc-common	1.3.3+ds-1+b1	transport-in
libtirpc-dev:amd64	1.3.3+ds-1+b1	transport-in
libtirpc3:amd64	1.3.3+ds-1+b1	transport-in
libtsan2:amd64	12.2.0-14.astra3+b1	ThreadSani
libubsan1:amd64	12.2.0-14.astra3+b1	UBSan – un
libudev1:amd64	252.17-1~deb12u1astra.se1+ci13+b1	libudev sha
libunistring2:amd64	1.0-2+b1	Unicode str
libuuid1:amd64	2.38.1-5+deb12u1+astra13+b1	Universally
libxxhash0:amd64	0.8.1-1+b1	shared libra
libyaml-0-2:amd64	0.2.5-1+b1	Fast YAML 1
libzstd1:amd64	1.5.4+dfsg2-5+b1	fast lossles
linux-libc-dev:amd64	5.4.0-54astra7+ci98	latest linux
linux-libc-dev-6.1.90-1:amd64	6.1.90-1.astra2+ci15	Linux Kerne
login	1:4.13+dfsg1-1.astra.se10+b1	system logi
logsave	1.47.0-2+b1	save the ou
lsb-base	11.6	transitional
mawk	1.3.4.20200120-3.1+b1	Pattern sca
media-types	10.0.0	List of stan
mount	2.38.1-5+deb12u1+astra13+b1	tools for mo
ncurses-base	6.4-4+b1	basic termi
ncurses-bin	6.4-4+b1	terminal-re
netbase	6.4	Basic TCP/I
openssh-client	1:9.6p1-2~deb10u1astra8se5	secure shel
openssl	3.2.0-2-astra5+ci1	Secure Soc
passwd	1:4.13+dfsg1-1.astra.se10+b1	change and
perl	5.36.0-7+deb12u1+ci202404171352+astra2	Larry Wall's
perl-base	5.36.0-7+deb12u1+ci202404171352+astra2	minimal Pe
perl-modules-5.36	5.36.0-7+deb12u1+ci202404171352+astra2	Core Perl m
pinentry-curses	1.2.1-1+b1	curses-base
procps	2:4.0.2-3+b1	/proc file sy
python3	3.11.2-1+b1	interactive
python3-attr	22.2.0-1	Attributes w
python3-bcrypt	3.2.2-1+b1	password h
python3-certifi	2022.9.24-1	root certific
python3-cffi	1.15.1-5+b1	Foreign Fun
python3-cffi-backend:amd64	1.15.1-5+b1	Foreign Fun
python3-chardet	5.1.0+dfsg-2	Universal C
python3-charset-normalizer	3.0.1-2	charset, en
python3-click	8.1.3-2	Wrapper ar
python3-colorama	0.4.6-2	Cross-platfo
python3-cryptography	38.0.4-3+ci202405171208+astra3+b1	Python libra
python3-daemon	2.3.2-1	library for r
python3-dev	3.11.2-1+b1	header files
python3-distlib	0.3.6-1	low-level co
python3-distutils	3.11.2-3+b1	distutils pa
python3-dnspython	2.3.0-1+b1	DNS toolkit
python3-filelock	3.9.0-1	platform in
python3-httplib2	0.20.4-3	comprehen
python3-idna	3.3-1	Python IDN
python3-importlib-metadata	4.12.0-1	library to a
python3-jinja2	3.1.2-1	small but fa
python3-jsonschema	4.10.3-1	An(other) i

Таблица 6 – продолж

Пакет	Версия	Описание
python3-lib2to3	3.11.2-3+b1	Interactive
python3-lockfile	1:0.12.2-2.2	file locking
python3-markdown-it	2.1.0-5	Python port
python3-markupsafe	2.1.2-1+b1	HTML/XHTM
python3-mdurl	0.1.2-1	Python port
python3-minimal	3.11.2-1+b1	minimal su
python3-more-itertools	8.10.0-2	library with
python3-mypy-extensions	0.4.3-4	Experiment
python3-nacl	1.5.0-2+b1	Python bind
python3-netaddr	0.8.0-2	manipulatio
python3-ntlm-auth	1.4.0-2	NTLM low-l
python3-openssl	23.0.0-1+b1	Python 3 w
python3-packaging	23.0-1	core utilitie
python3-paramiko	2.12.0-2+b1	Make ssh v
python3-pathspec	0.11.0-1	utility librar
python3-pexpect	4.8.0-4	Python 3 m
python3-pip	23.0.1+dfsg-1	Python pac
python3-pkg-resources	66.1.1-1	Package Di
python3-platformdirs	2.6.0-1	determining
python3-ply	3.11-5+b1	Lex and Yac
python3-protobuf	3.21.12-3+b1	Python 3 bi
python3-ptyprocess	0.7.0-5	Run a subp
python3-pycparser	2.21-1	C parser in
python3-pygments	2.14.0+dfsg-1	syntax high
python3-pyparsing	3.0.9-1	alternative
python3-pyrsistent:amd64	0.18.1-1+b1	persistent/f
python3-requests	2.28.1+dfsg-1+b1	elegant and
python3-requests-ntlm	1.1.0-3+b1	Adds suppo
python3-resolvelib	0.9.0-2	module to r
python3-rich	13.3.1-1	render rich
python3-ruamel.yaml.clib:amd64	0.2.7-1+b1	C version o
python3-setuptools	66.1.1-1	Python3 Dis
python3-six	1.16.0-4	Python 2 an
python3-urllib3	1.26.12-1+ci202406111901+astra2	HTTP library
python3-wheel	0.38.4-2	built-packa
python3-winrm	0.3.0-4+deb12u1	Python 3 lib
python3-xmltodict	0.13.0-1	Makes work
python3-yaml	6.0-3+b1	YAML parse
python3-zipp	1.0.0-6	pathlib-com
python3.11	3.11.2-6astra4+b1	Interactive
python3.11-ansible-runner	2.4.0+aa1.1.0	Consistent
python3.11-backports.ssl-match-hostname	3.7.0.1+aa1.1.0	The ssl.mat
python3.11-black	24.10.0+aa1.1.0	The uncom
python3.11-bracex	2.5.post1+aa1.1.0	Bash style
python3.11-cryptography	42.0.8+aa1.1.0	cryptograph
python3.11-dev	3.11.2-6astra4+b1	Header files
python3.11-googleapis-common-protos	1.65.0+aa1.1.0	Common pr
python3.11-grpcio	1.67.1+aa1.1.0	HTTP/2-bas
python3.11-minimal	3.11.2-6astra4+b1	Minimal sub
python3.11-pika	1.3.2+aa1.1.0	Pika Python
python3.11-protobuf	4.25.5+aa1.1.0	Protocol Bu
python3.11-pyjwt	2.9.0+aa1.1.0	JSON Web T
python3.11-pypsrp	0.8.1+aa1.1.0	PowerShell
python3.11-pyspnego	0.11.0+aa1.1.0	Windows N
python3.11-python-gnupg	0.5.2+aa1.1.0	A wrapper f
python3.11-requests	2.32.3+aa1.1.0	Python HTT
python3.11-ruamel.yaml	0.18.6+aa1.1.0	ruamel.yam

Таблица 6 – продолж

Пакет	Версия	Описание
python3.11-subprocess-tee	0.4.2+aa1.1.0	subprocess
python3.11-urllib3	2.2.3+aa1.1.0	HTTP librar
python3.11-wcmatch	10.0+aa1.1.0	Wildcard/gl
python3.11-yamllint	1.35.1+aa1.1.0	A linter for
python3.11-yandexcloud	0.324.0+aa1.1.0	The Yandex
readline-common	8.2-1.3+b1	GNU readli
rpcsvc-proto	1.4.3-1+b1	RPC protoc
sed	4.9-1+b1	GNU strean
sensible-utils	0.0.17+nmu1	Utilities for
sshpass	1.09-1+b1	Non-interac
sysvinit-utils	3.06-4+b1	System-V-li
tar	1.34+dfsg-1.2+deb12u1+b1	GNU versio
tcl-expect:amd64	5.45.4-2+ci202403211247+astra1+b1	Automates
tcl8.6	8.6.13+dfsg-2+b1	Tcl (the Too
tzdata	2024a-4	time zone a
usrmerge	37~deb12u1+b1	Convert the
util-linux	2.38.1-5+deb12u1+astra13+b1	miscellaneo
util-linux-extra	2.38.1-5+deb12u1+astra13+b1	interactive
zlib1g:amd64	1:1.2.13.dfsg-1+b1	compressio
zlib1g-dev:amd64	1:1.2.13.dfsg-1+b1	compressio

Пакеты Python

Пакет
ansible-compat @ file:///docker_share/dist/ansible_compat-24.9.1-py3-none-any.whl
ansible-core
ansible-lint @ file:///docker_share/dist/ansible_lint-24.9.2-py3-none-any.whl
ansible-runner
ansible-sign @ file:///docker_share/dist/ansible_sign-0.1.1-py3-none-any.whl
attrs
backports.ssl_match_hostname @ file:///docker_share/dist/backports.ssl_match_hostname-3.7.0.1-py2.py3
bcrypt
black
bracex
certifi
cffi
cfgv
chardet
charset-normalizer
click
colorama
cryptography
Cython
distlib
dnspython
docker
dumb-init
filelock
googleapis-common-protos
grpcio
httplib2
identify
idna
importlib-metadata
Jinja2

продолжается

Таблица 7 – продолжение с предыдущей страницы

Пакет
jmespath
jsonschema
lockfile
markdown-it-py
MarkupSafe
mdurl
more-itertools
mypy-extensions
netaddr
nodeenv
ntlm-auth
packaging
paramiko
pathspec
pexpect
pika @ file:///docker_share/dist/pika-1.3.2-py3-none-any.whl
platformdirs
ply
pre_commit
protobuf
psycopg2-binary
ptyprocess
pycparser
Pygments
PyJWT
PyNaCl
pyOpenSSL
pyparsing
pypsrp @ file:///docker_share/dist/pypsrp-0.8.1-py3-none-any.whl
pyrsistent
pyspnego
python-daemon
python-gnupg
pywinrm
PyYAML
requests
requests-ntlm
resolvelib
rich
ruamel.yaml
ruamel.yaml.clib
six
subprocess-tee
textfsm
ttp
urllib3
virtualenv
wcmatch
xmltodict
yamllint
yandexcloud
zipp

Коллекции Ansible

Коллекция	Версия
ansible.posix	1.5.4
ansible.utils	3.0.0
astra.aa_controller	0.14.0
astra.ald_pro	3.0.0
astra.astralinux	0.5.0
astra.brest	4.0.3
astra.ceph	3.0.3
astra.chrony	0.2.1
astra.cups	1.3.4
astra.dcimanager	0.2.1
astra.dhcp	2.0.2
astra.docker	4.0.1
astra.etcd	0.2.1
astra.freeipa	0.4.1
astra.grafana	2.0.1
astra.haproxy	2.0.1
astra.hardening	0.3.1
astra.iscsi	0.4.2
astra.keepalived	0.3.0
astra.keycloak	2.0.2
astra.memcached	3.0.1
astra.nfs	0.4.1
astra.nginx	1.8.4
astra.ocfs2	0.2.1
astra.patroni	0.2.1
astra.pgbouncer	0.2.1
astra.postgresql	3.0.1
astra.prometheus	2.1.1
astra.rabbitmq	2.0.1
astra.repo_mirror	0.2.1
astra.rubackup	0.7.1
astra.rupost	0.16.1
astra.termidesk	0.10.2
astra.yandexcloud	0.4.0
community.docker	4.1.0
community.general	7.5.0
community.library_inventory_filtering_v1	1.0.2
community.postgresql	2.3.2
community.rabbitmq	1.2.3
freeipa.ansible_freeipa	1.10.0

Образы `aa-control-ee`

`0.1.1`

Пакеты APT

Пакет	Версия	Описание
adduser	3.134+astra.se10	add and rem
ansible-core	2.15.10-1	Configuratio
ansible-sign	0.1.1+aa1.1.0	Ansible cont
apt	2.6.1+ci202312061551+astra2+b1	commandlin
apt-transport-https	2.6.1+ci202312061551+astra2+b1	transitional
astra-archive-keyring	2023.12	GnuPG archi

Пакет	Версия	Описание
astra-version	8.1.41+v1.8.1.12	Update Astr
base-files	12.4+deb12u1.astra6	Debian base
base-passwd	3.6.1+b1	Debian base
bash	5.2.15-2.astra.se1+b2	GNU Bourne
binutils	2.40-2+b1	GNU assemb
binutils-common:amd64	2.40-2+b1	Common file
binutils-x86-64-linux-gnu	2.40-2+b1	GNU binary
bsdutils	1:2.38.1-5+deb12u1+astra13+b1	basic utilitie
ca-certificates	20230311+b2	Common CA
coreutils	9.1-1+ci202402121122+astra2+b1	GNU core ut
cpp	4:12.2.0-3+b1	GNU C prepr
cpp-12	12.2.0-14.astra3+b1	GNU C prepr
curl	7.88.1-10+deb12u5+b2	command li
dash	0.5.12-2.astra1	POSIX-comp
debconf	1.5.82+b1	Debian conf
debianutils	5.7-0.5~deb12u1+b1	Miscellaneou
diffutils	1:3.8-4+b1	File compari
dirmngr	2.2.40-1.1.astra6+b2	GNU privacy
dpkg	1.21.22.astra.se3+b2	Debian pack
e2fsprogs	1.47.0-2+b1	ext2/ext3/ex
expect	5.45.4-2+ci202403211247+astra1+b1	Automates i
findutils	4.9.0-4+b1	utilities for f
fontconfig-config	2.14.2-4ubuntu1+b1	generic font
fonts-noto-core	20201225-1	No Tofu font
gawk	1:5.2.1-2+b1	GNU awk, a
gcc	4:12.2.0-3+b1	GNU C comp
gcc-12	12.2.0-14.astra3+b1	GNU C comp
gcc-12-base:amd64	12.2.0-14.astra3+b1	GCC, the GN
git	1:2.39.2-1.1+ci3	fast, scalabl
git-man	1:2.39.2-1.1+ci3	fast, scalabl
gnupg	2.2.40-1.1.astra6+b2	GNU privacy
gnupg-l10n	2.2.40-1.1.astra6+b2	GNU privacy
gnupg-utils	2.2.40-1.1.astra6+b2	GNU privacy
gpg	2.2.40-1.1.astra6+b2	GNU Privacy
gpg-agent	2.2.40-1.1.astra6+b2	GNU privacy
gpg-wks-client	2.2.40-1.1.astra6+b2	GNU privacy
gpg-wks-server	2.2.40-1.1.astra6+b2	GNU privacy
gpgconf	2.2.40-1.1.astra6+b2	GNU privacy
gpgsm	2.2.40-1.1.astra6+b2	GNU privacy
gpgv	2.2.40-1.1.astra6+b2	GNU privacy
grep	3.8-5+b1	GNU grep, e
gzip	1.12-1+b1	GNU compre
hostname	3.23+nmu1+b1	utility to set
icu-devtools	72.1-3+b1	Developmen
ieee-data	20220827.1	OUI and IAB
init-system-helpers	1.65.2+b1	helper tools
javascript-common	11+nmu1+b1	Base suppor
less	590-2.1~deb12u2	pager progra
libabsl20220623:amd64	20220623.1-1+b1	extensions t
libacl1:amd64	2.3.1-3+b1	access contr
libaom3:amd64	3.6.0-1+ci202406111403+astra3	AV1 Video C
libapt-pkg6.0:amd64	2.6.1+ci202312061551+astra2+b1	package ma
libasan8:amd64	12.2.0-14.astra3+b1	AddressSani
libassuan0:amd64	2.5.5-5+b1	IPC library fo
libatomic1:amd64	12.2.0-14.astra3+b1	support libra
libattr1:amd64	1:2.5.1-4+ci202403112058+astra3+b1	extended at
libaudit-common	1:3.0.9-1+ci202401241435+astra3+b2	Dynamic libr

Пакет	Версия	Описание
libaudit1:amd64	1:3.0.9-1+ci202401241435+astra3+b2	Dynamic lib
libavif15:amd64	0.11.1-1+b1	Library for h
libbinutils:amd64	2.40-2+b1	GNU binary
libblkid1:amd64	2.38.1-5+deb12u1+astra13+b1	block device
libbrotli1:amd64	1.0.9-2+b1	library imple
libbsd0:amd64	0.11.7-2+b1	utility functi
libbz2-1.0:amd64	1.0.8-5+b1	high-quality
libc-bin	2.36-9+deb12u7+ci202405171200+astra5+b1	GNU C Libra
libc-dev-bin	2.36-9+deb12u7+ci202405171200+astra5+b1	GNU C Libra
libc-devtools	2.36-9+deb12u7+ci202405171200+astra5+b1	GNU C Libra
libc6:amd64	2.36-9+deb12u7+ci202405171200+astra5+b1	GNU C Libra
libc6-dev:amd64	2.36-9+deb12u7+ci202405171200+astra5+b1	GNU C Libra
libcap-ng0:amd64	0.8.3-1+b2	alternate PO
libcap2:amd64	1:2.66-4+b1	POSIX 1003.
libcbor0.8:amd64	0.8.0-2+b1	library for pa
libcc1-0:amd64	12.2.0-14.astra3+b1	GCC cc1 plu
libcom-err2:amd64	1.47.0-2+b1	common err
libcrypt-dev:amd64	1:4.4.33-2+b1	libcrypt dev
libcrypt1:amd64	1:4.4.33-2+b1	libcrypt shar
libctf-nobfd0:amd64	2.40-2+b1	Compact C T
libctf0:amd64	2.40-2+b1	Compact C T
libcurl3-gnutls:amd64	7.88.1-10+deb12u5+b2	easy-to-use
libcurl4:amd64	7.88.1-10+deb12u5+b2	easy-to-use
libdav1d7:amd64	1.4.1-1	fast and sma
libdb5.3:amd64	5.3.28+dfsg2-1+ci202311141605+astra1+b1	Berkeley v5.
libde265-0:amd64	1.0.11-1+deb12u2+b1	Open H.265
libdebconfclient0:amd64	0.270astra1+b1	Debian Conf
libdeflate0:amd64	1.14-1+b1	fast, whole-b
libedit2:amd64	3.1-20221030-2+b1	BSD editline
liberror-perl	0.17029-2+b1	Perl module
libexpat1:amd64	2.5.0-1+ci202405221303+astra2	XML parsing
libexpat1-dev:amd64	2.5.0-1+ci202405221303+astra2	XML parsing
libext2fs2:amd64	1.47.0-2+b1	ext2/ext3/ex
libffi8:amd64	3.4.4-1+b1	Foreign Func
libfido2-1:amd64	1.12.0-2+b2	library for g
libfile-find-rule-perl	0.34-3+b1	module to s
libfontconfig1:amd64	2.14.2-4ubuntu1+b1	generic font
libfreetype6:amd64	2.12.1+dfsg-5+b1	FreeType 2 f
libgav1-1:amd64	0.18.0-1+b1	AV1 decoder
libgcc-12-dev:amd64	12.2.0-14.astra3+b1	GCC support
libgcc-s1:amd64	12.2.0-14.astra3+b1	GCC support
libgcrypt20:amd64	1.10.1-3.astra2+b1	LGPL Crypto
libgd3:amd64	2.3.3-9+b2	GD Graphics
libgdbm-compat4:amd64	1.23-3+b1	GNU dbm da
libgdbm6:amd64	1.23-3+b1	GNU dbm da
libgmp10:amd64	2:6.2.1+dfsg1-1.1+b1	Multiprecisio
libgnutls30:amd64	3.7.9-2+deb12u2+b1	GNU TLS libr
libgomp1:amd64	12.2.0-14.astra3+b1	GCC OpenM
libgost	2.0.2-5+ci4+b1	This packag
libgpg-error0:amd64	1.46-1+b1	GnuPG deve
libgprofng0:amd64	2.40-2+b1	GNU Next G
libgssapi-krb5-2:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerbero
libheif1:amd64	1.15.1-1+ci2	ISO/IEC 230
libhogweed6:amd64	3.8.1-2+b1	low level cry
libicu-dev:amd64	72.1-3+b1	Developmen
libicu72:amd64	72.1-3+b1	Internationa
libidn2-0:amd64	2.3.3-1+b1	Internationa

Пакет	Версия	Описание
libisl23:amd64	0.25-1.1+b1	manipulatin
libitm1:amd64	12.2.0-14.astra3+b1	GNU Transac
libjansson4:amd64	2.14-2+b1	C library for
libjbig0:amd64	2.1-6.1+b1	JBIGkit libra
libjpeg62-turbo:amd64	1:2.1.5-2+b1	libjpeg-turbo
libjs-jquery	3.6.1+dfsg+~3.5.14-1+b1	JavaScript li
libjs-sphinxdoc	5.3.0-4+b1	JavaScript su
libjs-underscore	1.13.4~dfsg+~1.11.4-3	JavaScript's
libk5crypto3:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerbero
libkeyutils1:amd64	1.6.3-2+b1	Linux Key M
libkrb5-3:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerbero
libkrb5support0:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerbero
libksba8:amd64	1.6.3-2+b1	X.509 and C
libldap-2.5-0:amd64	2.5.13+dfsg-5+ci202405061914+astra2	OpenLDAP li
liblerc4:amd64	4.0.0+ds-2+b1	Limited Erro
liblocale-gettext-perl	1.07-5+b1	module usin
liblsan0:amd64	12.2.0-14.astra3+b1	LeakSanitize
liblz4-1:amd64	1.9.4-1+b1	Fast LZ com
liblzma5:amd64	5.4.1-0.2+b1	XZ-format c
libmd0:amd64	1.0.4-2+b1	message dig
libmount1:amd64	2.38.1-5+deb12u1+astra13+b1	device mou
libmpc3:amd64	1.3.1-1+b1	multiple pre
libmpfr6:amd64	4.2.0-1+b1	multiple pre
libncursesw6:amd64	6.4-4+b1	shared libra
libnettle8:amd64	3.8.1-2+b1	low level cry
libnghttp2-14:amd64	1.52.0-1+deb12u1+ci1+b1	library imple
libnpth0:amd64	1.6-3+b1	replacement
libnsl-dev:amd64	1.3.0-2+b1	libnsl develo
libnsl2:amd64	1.3.0-2+b1	Public client
libnuma1:amd64	2.0.16-1+b1	Libraries for
libnumber-compare-perl	0.03-3+b1	module for p
libp11-kit0:amd64	0.24.1-2+b1	library for lo
libpam-modules:amd64	1.5.2-6+deb12u1.astra.se21+b1	Pluggable A
libpam-modules-bin	1.5.2-6+deb12u1.astra.se21+b1	Pluggable A
libpam-runtime	1.5.2-6+deb12u1.astra.se21+b1	Runtime sup
libpam0g:amd64	1.5.2-6+deb12u1.astra.se21+b1	Pluggable A
libparsec-base3	3.9+ci132	base library
libparsec-cap3	3.9+ci132	capabilities
libpcre2-8-0:amd64	10.42-1+b1	New Perl Co
libpdp	3.9+ci132	library for PA
libperl5.36:amd64	5.36.0-7+deb12u1+ci202404171352+astra2	shared Perl
libpng16-16:amd64	1.6.39-2+b1	PNG library
libproc2-0:amd64	2:4.0.2-3+b1	library for ac
libpsl5:amd64	0.21.2-1+b1	Library for P
libpython3-dev:amd64	3.11.2-1+b1	header files
libpython3-stdlib:amd64	3.11.2-1+b1	interactive h
libpython3.11:amd64	3.11.2-6astra4+b1	Shared Pyth
libpython3.11-dev:amd64	3.11.2-6astra4+b1	Header files
libpython3.11-minimal:amd64	3.11.2-6astra4+b1	Minimal sub
libpython3.11-stdlib:amd64	3.11.2-6astra4+b1	Interactive h
libquadmath0:amd64	12.2.0-14.astra3+b1	GCC Quad-P
librav1e0:amd64	0.7.1-2+b1	Fastest and
libreadline8:amd64	8.2-1.3+b1	GNU readlin
librtmp1:amd64	2.4+20151223.gitfa8646d.1-2+b1	toolkit for RT
libsasl2-2:amd64	2.1.28+dfsg-10+b2	Cyrus SASL
libsasl2-modules-db:amd64	2.1.28+dfsg-10+b2	Cyrus SASL
libseccomp2:amd64	2.5.4-1+b2	high level in

Пакет	Версия	Описание
libselinux1:amd64	3.4-1+b1	SELinux run
libsemanage-common	3.4-1+b1	Common file
libsemanage2:amd64	3.4-1+b1	SELinux poli
libsepol2:amd64	3.4-2.1+b1	SELinux libr
libsigsegv2:amd64	2.14-1+b1	Library for h
libsmartcols1:amd64	2.38.1-5+deb12u1+astra13+b1	smart colum
libsodium23:amd64	1.0.18-1+b1	Network cor
libsqlite3-0:amd64	3.40.1-2+ci202405281356+astra2	SQLite 3 sha
libss2:amd64	1.47.0-2+b1	command-li
libssh2-1:amd64	1.10.0-3+b2	SSH2 client-
libssl3:amd64	3.2.0-2-astra5+ci1	Secure Sock
libstdc++6:amd64	12.2.0-14.astra3+b1	GNU Standa
libsvtav1enc1:amd64	1.4.1+dfsg-1+b1	Scalable Vid
libsystemd0:amd64	252.17-1~deb12u1astra.se1+ci13+b1	systemd util
libtasn1-6:amd64	4.19.0-2+b1	Manage ASN
libtcl8.6:amd64	8.6.13+dfsg-2+b1	Tcl (the Tool
libtext-charwidth-perl:amd64	0.04-11+b1	get display w
libtext-glob-perl	0.11-3+b1	Perl module
libtext-iconv-perl:amd64	1.7-8+b1	module to c
libtext-wrapi18n-perl	0.06-10+b1	internationa
libtiff6:amd64	4.5.0-6+deb12u1+ci202405171501+astra2	Tag Image F
libtinfo6:amd64	6.4-4+b1	shared low-l
libtirpc-common	1.3.3+ds-1+b1	transport-in
libtirpc-dev:amd64	1.3.3+ds-1+b1	transport-in
libtirpc3:amd64	1.3.3+ds-1+b1	transport-in
libtsan2:amd64	12.2.0-14.astra3+b1	ThreadSanit
libubsan1:amd64	12.2.0-14.astra3+b1	UBSan – und
libudev1:amd64	252.17-1~deb12u1astra.se1+ci13+b1	libudev shar
libunistring2:amd64	1.0-2+b1	Unicode stri
libuuid1:amd64	2.38.1-5+deb12u1+astra13+b1	Universally U
libwebp7:amd64	1.2.4-0.2+deb12u1+b1	Lossy compr
libx11-6:amd64	2:1.8.7-1astra3+b1	X11 client-si
libx11-data	2:1.8.7-1astra3+b1	X11 client-si
libx265-199:amd64	3.5-2+b1	H.265/HEVC
libxau6:amd64	1:1.0.9-1+b1	X11 authoris
libxcb1:amd64	1.15-1astra.se3+b1	X C Binding
libxdmcp6:amd64	1:1.1.2-3+b1	X11 Display
libxml2:amd64	2.9.14+dfsg-1.3~deb12u1+ci202405141738+astra2	GNOME XML
libxml2-dev:amd64	2.9.14+dfsg-1.3~deb12u1+ci202405141738+astra2	GNOME XML
libxpm4:amd64	1:3.5.17-1+b1	X11 pixmap
libxslt1.1:amd64	1.1.35-1+b1	XSLT 1.0 pro
libxxhash0:amd64	0.8.1-1+b1	shared libra
libyaml-0-2:amd64	0.2.5-1+b1	Fast YAML 1
libyuv0:amd64	0.0~git20230123.b2528b0-1+b1	Library for Y
libzstd1:amd64	1.5.4+dfsg2-5+b1	fast lossless
linux-libc-dev:amd64	5.4.0-54astra7+ci98	latest linux-l
linux-libc-dev-6.1.90-1:amd64	6.1.90-1.astra2+ci15	Linux Kernel
login	1:4.13+dfsg1-1.astra.se10+b1	system login
logsave	1.47.0-2+b1	save the out
lsb-base	11.6	transitional
manpages	6.03-2	Manual page
manpages-dev	6.03-2	Manual page
mawk	1.3.4.20200120-3.1+b1	Pattern scan
media-types	10.0.0	List of stand
mount	2.38.1-5+deb12u1+astra13+b1	tools for mo
ncurses-base	6.4-4+b1	basic termin
ncurses-bin	6.4-4+b1	terminal-rela

Пакет	Версия	Описание
netbase	6.4	Basic TCP/IP
nsx-policy-python-sdk	3.1.5.0.0+aa1.1.0	NSX Policy P
nsx-python-sdk	3.1.5.0.0+aa1.1.0	NSX Python
nsx-vmc-aws-integration-python-sdk	3.1.5.0.0+aa1.1.0	NSX VMC AW
nsx-vmc-policy-python-sdk	3.1.5.0.0+aa1.1.0	NSX VMC Po
openssh-client	1:9.6p1-2~deb10u1astra8se5	secure shell
openssl	3.2.0-2-astra5+ci1	Secure Sock
passwd	1:4.13+dfsg1-1.astra.se10+b1	change and
patch	2.7.6-7+b1	Apply a diff
perl	5.36.0-7+deb12u1+ci202404171352+astra2	Larry Wall’s
perl-base	5.36.0-7+deb12u1+ci202404171352+astra2	minimal Per
perl-modules-5.36	5.36.0-7+deb12u1+ci202404171352+astra2	Core Perl mo
pinentry-curses	1.2.1-1+b1	curses-base
procps	2:4.0.2-3+b1	/proc file sys
python3	3.11.2-1+b1	interactive h
python3-appdirs	1.4.4-3	determining
python3-argcomplete	2.0.0-1	bash tab co
python3-bcrypt	3.2.2-1+b1	password ha
python3-blinker	1.5-1	Fast, simple
python3-boto3	1.26.27+dfsg-1	Python inter
python3-botocore	1.29.27+repack-1+b1	Low-level, d
python3-cachetools	5.2.0-1	extensible m
python3-certifi	2022.9.24-1	root certifica
python3-cffi	1.15.1-5+b1	Foreign Func
python3-cffi-backend:amd64	1.15.1-5+b1	Foreign Func
python3-chardet	5.1.0+dfsg-2	Universal Ch
python3-charset-normalizer	3.0.1-2	charset, enc
python3-cryptography	38.0.4-3+ci202405171208+astra3+b1	Python libra
python3-daemon	2.3.2-1	library for m
python3-dateutil	2.8.2-2+b1	powerful ext
python3-decorator	5.1.1-3	simplify usa
python3-deprecation	2.1.0-2	Library to ha
python3-dev	3.11.2-1+b1	header files
python3-distlib	0.3.6-1	low-level co
python3-distutils	3.11.2-3+b1	distutils pac
python3-dnspython	2.3.0-1+b1	DNS toolkit f
python3-dogpile.cache	1.1.8-2	caching fron
python3-google-auth	1.5.1-3+b1	Google Auth
python3-httplib2	0.20.4-3	comprehens
python3-idna	3.3-1	Python IDNA
python3-importlib-metadata	4.12.0-1	library to ac
python3-iso8601	1.0.2-1	Python mod
python3-jinja2	3.1.2-1	small but fa
python3-jmespath	1.0.1-1	JSON Matchi
python3-json-pointer	2.3-2	resolve JSON
python3-jsonpatch	1.32-2	library to ap
python3-jwt	2.6.0-1+b1	Python 3 im
python3-keystoneauth1	5.0.0-2	authenticati
python3-kubernetes	22.6.0-2+b1	Kubernetes
python3-lib2to3	3.11.2-3+b1	Interactive h
python3-lockfile	1:0.12.2-2.2	file locking l
python3-lxml:amd64	4.9.2-1+b1	pythonic bin
python3-mako	1.2.4+ds-1	fast and ligh
python3-markupsafe	2.1.2-1+b1	HTML/XHTM
python3-minimal	3.11.2-1+b1	minimal sub
python3-more-itertools	8.10.0-2	library with
python3-munch	2.5.0-2	dot-accessib

Пакет	Версия	Описание
python3-nacl	1.5.0-2+b1	Python bind
python3-netaddr	0.8.0-2	manipulatio
python3-netifaces:amd64	0.11.0-2+b1	portable net
python3-ntlm-auth	1.4.0-2	NTLM low-le
python3-oauthlib	3.2.2-1+b1	generic, spe
python3-openstacksdk	0.101.0-2+b1	SDK for buil
python3-os-service-types	1.7.0-2	lib for consu
python3-packaging	23.0-1	core utilities
python3-paramiko	2.12.0-2+b1	Make ssh v2
python3-pbr	5.10.0-2+b1	inject useful
python3-pexpect	4.8.0-4	Python 3 mo
python3-pip	23.0.1+dfsg-1	Python pack
python3-pkg-resources	66.1.1-1	Package Dis
python3-ply	3.11-5+b1	Lex and Yac
python3-ptyprocess	0.7.0-5	Run a subpr
python3-pyasn1	0.4.8-3	ASN.1 librar
python3-pyasn1-modules	0.2.8-1	Collection of
python3-pycparser	2.21-1	C parser in P
python3-pyparsing	3.0.9-1	alternative t
python3-requests	2.28.1+dfsg-1+b1	elegant and
python3-requests-ntlm	1.1.0-3+b1	Adds suppor
python3-requests-oauthlib	1.3.0+ds-1	module prov
python3-requestsexceptions	1.4.0-3+b1	import exce
python3-resolvelib	0.9.0-2	module to re
python3-rsa	4.8-1	Pure-Python
python3-s3transfer	0.6.0-1	Amazon S3 T
python3-setuptools	66.1.1-1	Python3 Dis
python3-six	1.16.0-4	Python 2 an
python3-stevedore	4.0.2-2	manage dyn
python3-urllib3	1.26.12-1+ci202406111901+astra2	HTTP library
python3-websocket	1.2.3-1	WebSocket c
python3-wheel	0.38.4-2	built-packag
python3-winrm	0.3.0-4+deb12u1	Python 3 lib
python3-xmltodict	0.13.0-1	Makes worki
python3-yaml	6.0-3+b1	YAML parser
python3-zipp	1.0.0-6	pathlib-com
python3.11	3.11.2-6astra4+b1	Interactive h
python3.11-ansible-runner	2.4.0+aa1.1.0	Consistent A
python3.11-cryptography	42.0.8+aa1.1.0	cryptograph
python3.11-dev	3.11.2-6astra4+b1	Header files
python3.11-minimal	3.11.2-6astra4+b1	Minimal sub
python3.11-pyopenssl	24.3.0+aa1.1.0	Python wrap
python3.11-pypsrp	0.8.1+aa1.1.0	PowerShell R
python3.11-pyspnego	0.11.0+aa1.1.0	Windows Ne
python3.11-python-gnupg	0.5.2+aa1.1.0	A wrapper fo
python3.11-pyvmomi	8.0.3.0.1+aa1.1.0	VMware vSp
python3.11-requests	2.32.3+aa1.1.0	Python HTTP
python3.11-urllib3	2.2.3+aa1.1.0	HTTP library
readline-common	8.2-1.3+b1	GNU readlin
rpcsvc-proto	1.4.3-1+b1	RPC protoco
sed	4.9-1+b1	GNU stream
sensible-utils	0.0.17+nmu1	Utilities for s
sshpass	1.09-1+b1	Non-interact
sysvinit-utils	3.06-4+b1	System-V-lik
tar	1.34+dfsg-1.2+deb12u1+b1	GNU version
tcl-expect:amd64	5.45.4-2+ci202403211247+astra1+b1	Automates i
tcl8.6	8.6.13+dfsg-2+b1	Tcl (the Tool

Пакет	Версия	Описание
tofu	1.6.2	OpenTofu is
tzdata	2024a-4	time zone a
usrmerge	37~deb12u1+b1	Convert the
util-linux	2.38.1-5+deb12u1+astra13+b1	miscellaneo
util-linux-extra	2.38.1-5+deb12u1+astra13+b1	interactive l
vapi-common-client	2.34.0+aa1.1.0	vAPI Commo
vapi-runtime	2.34.0+aa1.1.0	vAPI Runtim
vmware-vapi-common-client	2.52.0+aa1.1.0	VMware vAP
vmware-vapi-runtime	2.52.0+aa1.1.0	VMware vAP
vmware-vcenter	8.0.3.0+aa1.1.0	Client library
vmwarecloud-aws	1.64.1+aa1.1.0	Client bindi
vmwarecloud-draas	1.23.1+aa1.1.0	Client bindi
vsphere-automation-sdk	1.87.0+aa1.1.0	VMware vSp
zlib1g:amd64	1:1.2.13.dfsg-1+b1	compressio
zlib1g-dev:amd64	1:1.2.13.dfsg-1+b1	compressio

Пакеты Python

Пакет
adal
aiohappyeyeballs
aiohttp
aiosignal
ansible-core
ansible-runner
ansible-sign @ file:///docker_share/dist/ansible_sign-0.1.1-py3-none-any.whl
anyio
appdirs
applicationinsights
argcomplete
attrs
awxkit
azure-cli-core
azure-cli-telemetry
azure-common
azure-containerregistry
azure-core
azure-identity
azure-iot-hub
azure-keyvault
azure-keyvault-certificates
azure-keyvault-keys
azure-keyvault-secrets
azure-mgmt-apimanagement
azure-mgmt-authorization
azure-mgmt-automation
azure-mgmt-batch
azure-mgmt-cdn
azure-mgmt-compute
azure-mgmt-containerinstance
azure-mgmt-containerregistry
azure-mgmt-containerservice
azure-mgmt-core
azure-mgmt-cosmosdb
azure-mgmt-datafactory

Таблица 10 – продолжение

Пакет
azure-mgmt-devtestlabs
azure-mgmt-dns
azure-mgmt-eventhub
azure-mgmt-hdinsight
azure-mgmt-iothub
azure-mgmt-keyvault
azure-mgmt-loganalytics
azure-mgmt-managedservices
azure-mgmt-managementgroups
azure-mgmt-marketplaceordering
azure-mgmt-monitor
azure-mgmt-network
azure-mgmt-notificationhubs
azure-mgmt-nspkg
azure-mgmt-privatedns
azure-mgmt-rdbms
azure-mgmt-recoveryservices
azure-mgmt-recoveryservicesbackup
azure-mgmt-redis
azure-mgmt-resource
azure-mgmt-search
azure-mgmt-servicebus
azure-mgmt-sql
azure-mgmt-storage
azure-mgmt-trafficmanager
azure-mgmt-web
azure-nspkg
azure-storage-blob
bcrypt
blinker
boto3
botocore
cachetools
certifi
cffi
chardet
charset-normalizer
cryptography
decorator
Deprecated
deprecation
distlib
distro
dnspython
dogpile.cache
dumb-init
frozenlist
google-api-core
googleapis-common-protos
google-auth
google-cloud-core
google-cloud-storage
google-crc32c
google-resumable-media
h11
h2
hpack

Таблица 10 – продолжение

Пакет
httpcore
httplib2
httpx
humanfriendly
hyperframe
idna
importlib_metadata
iso8601
isodate
Jinja2
jmespath
jsonpatch
jsonpointer
jsonschema
jsonschema-specifications
keystoneauth1
knack
kubernetes
lockfile
lxml
Mako
MarkupSafe
microsoft-kiota-abstractions
microsoft-kiota-authentication-azure
microsoft-kiota-http
microsoft-kiota-serialization-form
microsoft-kiota-serialization-json
microsoft-kiota-serialization-multipart
microsoft-kiota-serialization-text
more-itertools
msal
msal-extensions
msgraph-core
msgraph-sdk
msrest
msrestazure
multidict
munch
netaddr
netifaces
nsx_policy_python_sdk @ file://localhost//tmp/pip-req-build-3o3nnbi1/lib/nsx-policy-python-sdk/nsx_policy_
nsx_python_sdk @ file://localhost//tmp/pip-req-build-3o3nnbi1/lib/nsx-python-sdk/nsx_python_sdk-4.2.0-py
nsx-vmc-aws-integration-python-sdk @ file://localhost//tmp/pip-req-build-3o3nnbi1/lib/nsx-vmc-aws-integr
nsx-vmc-policy-python-sdk @ file://localhost//tmp/pip-req-build-3o3nnbi1/lib/nsx-vmc-policy-python-sdk/ns
ntlm-auth
oauthlib
openstacksdk
opentelemetry-api
opentelemetry-sdk
opentelemetry-semantic-conventions
os-service-types
ovirt-engine-sdk-python
ovirt-imageio
packaging
paramiko
pbr
pendulum

Таблица 10 – продолжение

Пакет
pexpect
pkginfo
platformdirs
ply
portalocker
propcache
protobuf
proto-plus
psutil
ptyprocess
pyasn1
pyasn1-modules
pycparser
pycurl
Pygments
PyJWT
PyNaCl
pyOpenSSL
pyparsing
pypsrp @ file:///docker_share/dist/pypsrp-0.8.1-py3-none-any.whl
PySocks
pyspnego
python-daemon
python-dateutil
python-gnupg
pytz
pyvmomi
pywinrm
PyYAML
referencing
requests
requestsexceptions
requests-ntlm
requests-oauthlib
resolvelib
rpds-py
rsa
s3transfer
six
sniffio
std-uritemplate
stevedore
tabulate
textfsm
time-machine
ttp
typing_extensions
tzdata
uamqp
urllib3
vapi-common-client @ file://localhost//docker_share/package/lib/vapi-common-client/vapi_common_client-
vapi-runtime @ file://localhost//docker_share/package/lib/vapi-runtime/vapi_runtime-2.34.0-py2.py3-none-
vmwarecloud-aws @ file://localhost//docker_share/package/lib/vmwarecloud-aws/vmwarecloud_aws-1.64.1
vmwarecloud-draas @ file://localhost//docker_share/package/lib/vmwarecloud-draas/vmwarecloud_draas-1
vmware-vapi-common-client
vmware-vapi-runtime
vmware_vcenter

Таблица 10 – продолжение

Пакет
vsphere-automation-sdk @ file:///docker_share/dist/vsphere_automation_sdk-1.87.0-py3-none-any.whl
websocket-client
wrapt
xmltodict
yarl
zipp

Коллекции Ansible

Коллекция	Версия
amazon.aws	9.0.0
ansible.posix	1.5.4
ansible.utils	3.0.0
awx.awx	24.6.1
azure.azcollection	3.0.0
cloud.terraform	3.0.0
community.vmware	5.0.1
google.cloud	1.4.1
kubernetes.core	5.0.0
kubevirt.core	2.1.0
openstack.cloud	2.2.0
ovirt.ovirt	3.2.0
redhatinsights.insights	1.3.0
theforeman.foreman	4.2.0
vmware.vmware	1.10.1

0.1.0

Пакеты APT

Пакет	Версия	Описание
adduser	3.134+astra.se10	add and remove u
ansible-core	2.15.10-1	Configuration man
ansible-sign	0.1.1+aa1.1.0	Ansible content va
apt	2.6.1+ci202312061551+astra2+b1	commandline pack
apt-transport-https	2.6.1+ci202312061551+astra2+b1	transitional packag
astra-archive-keyring	2023.12	GnuPG archive key
astra-version	8.1.41+v1.8.1.12	Update Astra versi
base-files	12.4+deb12u1.astra6	Debian base syste
base-passwd	3.6.1+b1	Debian base syste
bash	5.2.15-2.astra.se1+b2	GNU Bourne Again
binutils	2.40-2+b1	GNU assembler, li
binutils-common:amd64	2.40-2+b1	Common files for t
binutils-x86-64-linux-gnu	2.40-2+b1	GNU binary utilitie
bsdutils	1:2.38.1-5+deb12u1+astra13+b1	basic utilities from
ca-certificates	20230311+b2	Common CA certif
coreutils	9.1-1+ci202402121122+astra2+b1	GNU core utilities
cpp	4:12.2.0-3+b1	GNU C preprocesso
cpp-12	12.2.0-14.astra3+b1	GNU C preprocesso
curl	7.88.1-10+deb12u5+b2	command line tool
dash	0.5.12-2.astra1	POSIX-compliant sh
debconf	1.5.82+b1	Debian configurati
debianutils	5.7-0.5~deb12u1+b1	Miscellaneous utili
diffutils	1:3.8-4+b1	File comparison ut

Пакет	Версия	Описание
dirmngr	2.2.40-1.1.astra6+b2	GNU privacy guard
dpkg	1.21.22.astra.se3+b2	Debian package m
e2fsprogs	1.47.0-2+b1	ext2/ext3/ext4 file
expect	5.45.4-2+ci202403211247+astra1+b1	Automates interac
findutils	4.9.0-4+b1	utilities for finding
fontconfig-config	2.14.2-4ubuntu1+b1	generic font config
fonts-noto-core	20201225-1	No Tofu font famili
gawk	1:5.2.1-2+b1	GNU awk, a patter
gcc	4:12.2.0-3+b1	GNU C compiler
gcc-12	12.2.0-14.astra3+b1	GNU C compiler
gcc-12-base:amd64	12.2.0-14.astra3+b1	GCC, the GNU Com
git	1:2.39.2-1.1+ci3	fast, scalable, distr
git-man	1:2.39.2-1.1+ci3	fast, scalable, distr
gnupg	2.2.40-1.1.astra6+b2	GNU privacy guard
gnupg-l10n	2.2.40-1.1.astra6+b2	GNU privacy guard
gnupg-utils	2.2.40-1.1.astra6+b2	GNU privacy guard
gpg	2.2.40-1.1.astra6+b2	GNU Privacy Guard
gpg-agent	2.2.40-1.1.astra6+b2	GNU privacy guard
gpg-wks-client	2.2.40-1.1.astra6+b2	GNU privacy guard
gpg-wks-server	2.2.40-1.1.astra6+b2	GNU privacy guard
gpgconf	2.2.40-1.1.astra6+b2	GNU privacy guard
gpgsm	2.2.40-1.1.astra6+b2	GNU privacy guard
gpgv	2.2.40-1.1.astra6+b2	GNU privacy guard
grep	3.8-5+b1	GNU grep, egrep a
gzip	1.12-1+b1	GNU compression
hostname	3.23+nmu1+b1	utility to set/show
icu-devtools	72.1-3+b1	Development utilit
ieee-data	20220827.1	OUI and IAB listing
init-system-helpers	1.65.2+b1	helper tools for all
javascript-common	11+nmu1+b1	Base support for Ja
less	590-2.1~deb12u2	pager program sim
libabsl20220623:amd64	20220623.1-1+b1	extensions to the C
libacl1:amd64	2.3.1-3+b1	access control list
libaom3:amd64	3.6.0-1+ci202406111403+astra3	AV1 Video Codec L
libapt-pkg6.0:amd64	2.6.1+ci202312061551+astra2+b1	package managem
libasan8:amd64	12.2.0-14.astra3+b1	AddressSanitizer –
libassuan0:amd64	2.5.5-5+b1	IPC library for the G
libatomic1:amd64	12.2.0-14.astra3+b1	support library pro
libattr1:amd64	1:2.5.1-4+ci202403112058+astra3+b1	extended attribute
libaudit-common	1:3.0.9-1+ci202401241435+astra3+b2	Dynamic library fo
libaudit1:amd64	1:3.0.9-1+ci202401241435+astra3+b2	Dynamic library fo
libavif15:amd64	0.11.1-1+b1	Library for handlin
libbinutils:amd64	2.40-2+b1	GNU binary utilitie
libblkid1:amd64	2.38.1-5+deb12u1+astra13+b1	block device ID lib
libbrotli1:amd64	1.0.9-2+b1	library implementi
libbsd0:amd64	0.11.7-2+b1	utility functions fro
libbz2-1.0:amd64	1.0.8-5+b1	high-quality block-
libc-bin	2.36-9+deb12u7+ci202405171200+astra5+b1	GNU C Library: Bin
libc-dev-bin	2.36-9+deb12u7+ci202405171200+astra5+b1	GNU C Library: Dev
libc-devtools	2.36-9+deb12u7+ci202405171200+astra5+b1	GNU C Library: Dev
libc6:amd64	2.36-9+deb12u7+ci202405171200+astra5+b1	GNU C Library: Sha
libc6-dev:amd64	2.36-9+deb12u7+ci202405171200+astra5+b1	GNU C Library: Dev
libcap-ng0:amd64	0.8.3-1+b2	alternate POSIX ca
libcap2:amd64	1:2.66-4+b1	POSIX 1003.1e cap
libcbor0.8:amd64	0.8.0-2+b1	library for parsing
libcc1-0:amd64	12.2.0-14.astra3+b1	GCC cc1 plugin for
libcom-err2:amd64	1.47.0-2+b1	common error des

Пакет	Версия	Описание
libcrypt-dev:amd64	1:4.4.33-2+b1	libcrypt developme
libcrypt1:amd64	1:4.4.33-2+b1	libcrypt shared libr
libctf-nobfd0:amd64	2.40-2+b1	Compact C Type Fo
libctf0:amd64	2.40-2+b1	Compact C Type Fo
libcurl3-gnutls:amd64	7.88.1-10+deb12u5+b2	easy-to-use client-
libcurl4:amd64	7.88.1-10+deb12u5+b2	easy-to-use client-
libdav1d7:amd64	1.4.1-1	fast and small AV1
libdb5.3:amd64	5.3.28+dfsg2-1+ci202311141605+astra1+b1	Berkeley v5.3 Data
libde265-0:amd64	1.0.11-1+deb12u2+b1	Open H.265 video
libdebconfclient0:amd64	0.270astra1+b1	Debian Configurati
libdeflate0:amd64	1.14-1+b1	fast, whole-buffer
libedit2:amd64	3.1-20221030-2+b1	BSD editline and h
liberror-perl	0.17029-2+b1	Perl module for err
libexpat1:amd64	2.5.0-1+ci202405221303+astra2	XML parsing C libra
libexpat1-dev:amd64	2.5.0-1+ci202405221303+astra2	XML parsing C libra
libext2fs2:amd64	1.47.0-2+b1	ext2/ext3/ext4 file
libffi8:amd64	3.4.4-1+b1	Foreign Function In
libfido2-1:amd64	1.12.0-2+b2	library for generat
libfile-find-rule-perl	0.34-3+b1	module to search f
libfontconfig1:amd64	2.14.2-4ubuntu1+b1	generic font config
libfreetype6:amd64	2.12.1+dfsg-5+b1	FreeType 2 font en
libgav1-1:amd64	0.18.0-1+b1	AV1 decoder devel
libgcc-12-dev:amd64	12.2.0-14.astra3+b1	GCC support librar
libgcc-s1:amd64	12.2.0-14.astra3+b1	GCC support librar
libgcrypt20:amd64	1.10.1-3.astra2+b1	LGPL Crypto library
libgd3:amd64	2.3.3-9+b2	GD Graphics Librar
libgdbm-compat4:amd64	1.23-3+b1	GNU dbm database
libgdbm6:amd64	1.23-3+b1	GNU dbm database
libgmp10:amd64	2:6.2.1+dfsg1-1.1+b1	Multiprecision arith
libgnutls30:amd64	3.7.9-2+deb12u2+b1	GNU TLS library - r
libgomp1:amd64	12.2.0-14.astra3+b1	GCC OpenMP (GOM
libgost	2.0.2-5+ci4+b1	This package conta
libgpg-error0:amd64	1.46-1+b1	GnuPG developme
libgprofng0:amd64	2.40-2+b1	GNU Next Generat
libgssapi-krb5-2:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerberos runti
libheif1:amd64	1.15.1-1+ci2	ISO/IEC 23008-12:
libhogweed6:amd64	3.8.1-2+b1	low level cryptogra
libicu-dev:amd64	72.1-3+b1	Development files
libicu72:amd64	72.1-3+b1	International Comp
libidn2-0:amd64	2.3.3-1+b1	Internationalized d
libisl23:amd64	0.25-1.1+b1	manipulating sets
libitm1:amd64	12.2.0-14.astra3+b1	GNU Transactional
libjansson4:amd64	2.14-2+b1	C library for encod
libjbig0:amd64	2.1-6.1+b1	JBIGkit libraries
libjpeg62-turbo:amd64	1:2.1.5-2+b1	libjpeg-turbo JPEG
libjs-jquery	3.6.1+dfsg+~3.5.14-1+b1	JavaScript library f
libjs-sphinxdoc	5.3.0-4+b1	JavaScript support
libjs-underscore	1.13.4~dfsg+~1.11.4-3	JavaScript's functio
libk5crypto3:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerberos runti
libkeyutils1:amd64	1.6.3-2+b1	Linux Key Manage
libkrb5-3:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerberos runti
libkrb5support0:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerberos runti
libksba8:amd64	1.6.3-2+b1	X.509 and CMS su
libldap-2.5-0:amd64	2.5.13+dfsg-5+ci202405061914+astra2	OpenLDAP libraries
liblerc4:amd64	4.0.0+ds-2+b1	Limited Error Raste
liblocale-gettext-perl	1.07-5+b1	module using libc
liblsan0:amd64	12.2.0-14.astra3+b1	LeakSanitizer – a r

Пакет	Версия	Описание
liblz4-1:amd64	1.9.4-1+b1	Fast LZ compressio
liblzma5:amd64	5.4.1-0.2+b1	XZ-format compres
libmd0:amd64	1.0.4-2+b1	message digest fu
libmount1:amd64	2.38.1-5+deb12u1+astra13+b1	device mounting li
libmpc3:amd64	1.3.1-1+b1	multiple precision
libmpfr6:amd64	4.2.0-1+b1	multiple precision
libncursesw6:amd64	6.4-4+b1	shared libraries for
libnettle8:amd64	3.8.1-2+b1	low level cryptogra
libnghttp2-14:amd64	1.52.0-1+deb12u1+ci1+b1	library implementi
libnpth0:amd64	1.6-3+b1	replacement for GI
libnsl-dev:amd64	1.3.0-2+b1	libnsl development
libnsl2:amd64	1.3.0-2+b1	Public client interfa
libnuma1:amd64	2.0.16-1+b1	Libraries for contro
libnumber-compare-perl	0.03-3+b1	module for perform
libp11-kit0:amd64	0.24.1-2+b1	library for loading
libpam-modules:amd64	1.5.2-6+deb12u1.astra.se21+b1	Pluggable Authent
libpam-modules-bin	1.5.2-6+deb12u1.astra.se21+b1	Pluggable Authent
libpam-runtime	1.5.2-6+deb12u1.astra.se21+b1	Runtime support fo
libpam0g:amd64	1.5.2-6+deb12u1.astra.se21+b1	Pluggable Authent
libparsec-base3	3.9+ci132	base library for PA
libparsec-cap3	3.9+ci132	capabilities library
libpcre2-8-0:amd64	10.42-1+b1	New Perl Compatib
libpdp	3.9+ci132	library for PARSEC
libperl5.36:amd64	5.36.0-7+deb12u1+ci202404171352+astra2	shared Perl library
libpng16-16:amd64	1.6.39-2+b1	PNG library - runtir
libproc2-0:amd64	2:4.0.2-3+b1	library for accessin
libpsl5:amd64	0.21.2-1+b1	Library for Public S
libpython3-dev:amd64	3.11.2-1+b1	header files and a
libpython3-stdlib:amd64	3.11.2-1+b1	interactive high-lev
libpython3.11:amd64	3.11.2-6astra4+b1	Shared Python run
libpython3.11-dev:amd64	3.11.2-6astra4+b1	Header files and a
libpython3.11-minimal:amd64	3.11.2-6astra4+b1	Minimal subset of
libpython3.11-stdlib:amd64	3.11.2-6astra4+b1	Interactive high-le
libquadmath0:amd64	12.2.0-14.astra3+b1	GCC Quad-Precisio
librav1e0:amd64	0.7.1-2+b1	Fastest and safest
libreadline8:amd64	8.2-1.3+b1	GNU readline and
librtmp1:amd64	2.4+20151223.gitfa8646d.1-2+b1	toolkit for RTMP str
libsasl2-2:amd64	2.1.28+dfsg-10+b2	Cyrus SASL - authe
libsasl2-modules-db:amd64	2.1.28+dfsg-10+b2	Cyrus SASL - plugg
libseccomp2:amd64	2.5.4-1+b2	high level interface
libselinux1:amd64	3.4-1+b1	SELinux runtime sh
libsemanage-common	3.4-1+b1	Common files for S
libsemanage2:amd64	3.4-1+b1	SELinux policy ma
libsepol2:amd64	3.4-2.1+b1	SELinux library for
libsigsegv2:amd64	2.14-1+b1	Library for handlin
libsmartcols1:amd64	2.38.1-5+deb12u1+astra13+b1	smart column outp
libsodium23:amd64	1.0.18-1+b1	Network communic
libsqlite3-0:amd64	3.40.1-2+ci202405281356+astra2	SQLite 3 shared lib
libss2:amd64	1.47.0-2+b1	command-line inte
libssh2-1:amd64	1.10.0-3+b2	SSH2 client-side lib
libssl3:amd64	3.2.0-2-astra5+ci1	Secure Sockets Lay
libstdc++6:amd64	12.2.0-14.astra3+b1	GNU Standard C++
libsvtav1enc1:amd64	1.4.1+dfsg-1+b1	Scalable Video Tec
libsystemd0:amd64	252.17-1~deb12u1astra.se1+ci13+b1	systemd utility libr
libtasn1-6:amd64	4.19.0-2+b1	Manage ASN.1 stru
libtcl8.6:amd64	8.6.13+dfsg-2+b1	Tcl (the Tool Comm
libtext-charwidth-perl:amd64	0.04-11+b1	get display widths

Пакет	Версия	Описание
libtext-glob-perl	0.11-3+b1	Perl module for ma
libtext-iconv-perl:amd64	1.7-8+b1	module to convert
libtext-wrapi18n-perl	0.06-10+b1	internationalized s
libtiff6:amd64	4.5.0-6+deb12u1+ci202405171501+astra2	Tag Image File For
libtinfo6:amd64	6.4-4+b1	shared low-level te
libtirpc-common	1.3.3+ds-1+b1	transport-indepen
libtirpc-dev:amd64	1.3.3+ds-1+b1	transport-indepen
libtirpc3:amd64	1.3.3+ds-1+b1	transport-indepen
libtsan2:amd64	12.2.0-14.astra3+b1	ThreadSanitizer – a
libubsan1:amd64	12.2.0-14.astra3+b1	UBSan – undefined
libudev1:amd64	252.17-1~deb12u1astra.se1+ci13+b1	libudev shared libr
libunistring2:amd64	1.0-2+b1	Unicode string libr
libuuid1:amd64	2.38.1-5+deb12u1+astra13+b1	Universally Unique
libwebp7:amd64	1.2.4-0.2+deb12u1+b1	Lossy compression
libx11-6:amd64	2:1.8.7-1astra3+b1	X11 client-side libr
libx11-data	2:1.8.7-1astra3+b1	X11 client-side libr
libx265-199:amd64	3.5-2+b1	H.265/HEVC video
libxau6:amd64	1:1.0.9-1+b1	X11 authorisation
libxcb1:amd64	1.15-1astra.se3+b1	X C Binding
libxdmcp6:amd64	1:1.1.2-3+b1	X11 Display Manag
libxml2:amd64	2.9.14+dfsg-1.3~deb12u1+ci202405141738+astra2	GNOME XML librar
libxml2-dev:amd64	2.9.14+dfsg-1.3~deb12u1+ci202405141738+astra2	GNOME XML librar
libxpm4:amd64	1:3.5.17-1+b1	X11 pixmap library
libxslt1.1:amd64	1.1.35-1+b1	XSLT 1.0 processin
libxxhash0:amd64	0.8.1-1+b1	shared library for x
libyaml-0-2:amd64	0.2.5-1+b1	Fast YAML 1.1 pars
libyuv0:amd64	0.0~git20230123.b2528b0-1+b1	Library for YUV sca
libzstd1:amd64	1.5.4+dfsg2-5+b1	fast lossless comp
linux-libc-dev:amd64	5.4.0-54astra7+ci98	latest linux-libc-de
linux-libc-dev-6.1.90-1:amd64	6.1.90-1.astra2+ci15	Linux Kernel Heade
login	1:4.13+dfsg1-1.astra.se10+b1	system login tools
logsave	1.47.0-2+b1	save the output of
lsb-base	11.6	transitional packag
manpages	6.03-2	Manual pages abo
manpages-dev	6.03-2	Manual pages abo
mawk	1.3.4.20200120-3.1+b1	Pattern scanning a
media-types	10.0.0	List of standard m
mount	2.38.1-5+deb12u1+astra13+b1	tools for mounting
ncurses-base	6.4-4+b1	basic terminal type
ncurses-bin	6.4-4+b1	terminal-related pr
netbase	6.4	Basic TCP/IP netwo
openssh-client	1:9.6p1-2~deb10u1astra8se5	secure shell (SSH)
openssl	3.2.0-2-astra5+ci1	Secure Sockets Lay
passwd	1:4.13+dfsg1-1.astra.se10+b1	change and admin
patch	2.7.6-7+b1	Apply a diff file to
perl	5.36.0-7+deb12u1+ci202404171352+astra2	Larry Wall’s Practi
perl-base	5.36.0-7+deb12u1+ci202404171352+astra2	minimal Perl syste
perl-modules-5.36	5.36.0-7+deb12u1+ci202404171352+astra2	Core Perl modules
pinentry-curses	1.2.1-1+b1	curses-based PIN c
procps	2:4.0.2-3+b1	/proc file system u
python3	3.11.2-1+b1	interactive high-le
python3-appdirs	1.4.4-3	determining appro
python3-argcomplete	2.0.0-1	bash tab completi
python3-bcrypt	3.2.2-1+b1	password hashing
python3-blinker	1.5-1	Fast, simple object
python3-boto3	1.26.27+dfsg-1	Python interface to
python3-botocore	1.29.27+repack-1+b1	Low-level, data-dri

Пакет	Версия	Описание
python3-bs4	4.11.2-2	error-tolerant HTM
python3-cachetools	5.2.0-1	extensible memoiz
python3-certifi	2022.9.24-1	root certificates fo
python3-cffi-backend:amd64	1.15.1-5+b1	Foreign Function In
python3-chardet	5.1.0+dfsg-2	Universal Characte
python3-charset-normalizer	3.0.1-2	charset, encoding
python3-cryptography	38.0.4-3+ci202405171208+astra3+b1	Python library exp
python3-daemon	2.3.2-1	library for making
python3-dateutil	2.8.2-2+b1	powerful extensior
python3-decorator	5.1.1-3	simplify usage of P
python3-deprecation	2.1.0-2	Library to handle a
python3-dev	3.11.2-1+b1	header files and a
python3-distlib	0.3.6-1	low-level compone
python3-distutils	3.11.2-3+b1	distutils package f
python3-dnspython	2.3.0-1+b1	DNS toolkit for Pyt
python3-dogpile.cache	1.1.8-2	caching front-end
python3-google-auth	1.5.1-3+b1	Google Authentica
python3-html5lib	1.1-3	HTML parser/toker
python3-httplib2	0.20.4-3	comprehensive HT
python3-idna	3.3-1	Python IDNA2008
python3-importlib-metadata	4.12.0-1	library to access th
python3-iso8601	1.0.2-1	Python module to
python3-jinja2	3.1.2-1	small but fast and
python3-jmespath	1.0.1-1	JSON Matching Exp
python3-json-pointer	2.3-2	resolve JSON point
python3-jsonpatch	1.32-2	library to apply JS
python3-jwt	2.6.0-1+b1	Python 3 impleme
python3-keystoneauth1	5.0.0-2	authentication libr
python3-kubernetes	22.6.0-2+b1	Kubernetes Python
python3-lib2to3	3.11.2-3+b1	Interactive high-le
python3-lockfile	1:0.12.2-2.2	file locking library
python3-lxml:amd64	4.9.2-1+b1	pythonic binding fo
python3-mako	1.2.4+ds-1	fast and lightweigh
python3-markupsafe	2.1.2-1+b1	HTML/XHTML/XML
python3-minimal	3.11.2-1+b1	minimal subset of
python3-more-itertools	8.10.0-2	library with routine
python3-munch	2.5.0-2	dot-accessible dict
python3-nacl	1.5.0-2+b1	Python bindings to
python3-netaddr	0.8.0-2	manipulation of va
python3-netifaces:amd64	0.11.0-2+b1	portable network i
python3-ntlm-auth	1.4.0-2	NTLM low-level Pyt
python3-oauthlib	3.2.2-1+b1	generic, spec-com
python3-openstacksdk	0.101.0-2+b1	SDK for building ap
python3-os-service-types	1.7.0-2	lib for consuming O
python3-packaging	23.0-1	core utilities for py
python3-paramiko	2.12.0-2+b1	Make ssh v2 conne
python3-pbr	5.10.0-2+b1	inject useful and s
python3-pexpect	4.8.0-4	Python 3 module fo
python3-pip	23.0.1+dfsg-1	Python package in
python3-pkg-resources	66.1.1-1	Package Discovery
python3-ptyprocess	0.7.0-5	Run a subprocess i
python3-pyasn1	0.4.8-3	ASN.1 library for P
python3-pyasn1-modules	0.2.8-1	Collection of proto
python3-pyparsing	3.0.9-1	alternative to crea
python3-requests	2.28.1+dfsg-1+b1	elegant and simple
python3-requests-ntlm	1.1.0-3+b1	Adds support for N
python3-requests-oauthlib	1.3.0+ds-1	module providing O

Пакет	Версия	Описание
python3-requestsexceptions	1.4.0-3+b1	import exceptions
python3-resolvelib	0.9.0-2	module to resolve
python3-rsa	4.8-1	Pure-Python RSA i
python3-s3transfer	0.6.0-1	Amazon S3 Transfe
python3-setuptools	66.1.1-1	Python3 Distutils E
python3-six	1.16.0-4	Python 2 and 3 co
python3-soupsieve	2.3.2-1	modern CSS select
python3-stevedore	4.0.2-2	manage dynamic p
python3-urllib3	1.26.12-1+ci202406111901+astra2	HTTP library with t
python3-webencodings	0.5.1-5	Python implement
python3-websocket	1.2.3-1	WebSocket client l
python3-wheel	0.38.4-2	built-package form
python3-winrm	0.3.0-4+deb12u1	Python 3 library fo
python3-xmltodict	0.13.0-1	Makes working wit
python3-yaml	6.0-3+b1	YAML parser and e
python3-zipp	1.0.0-6	pathlib-compatible
python3.11	3.11.2-6astra4+b1	Interactive high-le
python3.11-ansible-runner	2.4.0+aa1.1.0	Consistent Ansible
python3.11-dev	3.11.2-6astra4+b1	Header files and a
python3.11-minimal	3.11.2-6astra4+b1	Minimal subset of
python3.11-pypsrp	0.8.1+aa1.1.0	PowerShell Remoti
python3.11-pyspnego	0.11.0+aa1.1.0	Windows Negotiate
python3.11-python-gnupg	0.5.2+aa1.1.0	A wrapper for the
python3.11-requests	2.32.3+aa1.1.0	Python HTTP for H
python3.11-urllib3	2.2.3+aa1.1.0	HTTP library with t
readline-common	8.2-1.3+b1	GNU readline and
rpcsvc-proto	1.4.3-1+b1	RPC protocol comp
sed	4.9-1+b1	GNU stream editor
sensible-utils	0.0.17+nmu1	Utilities for sensibl
sshpass	1.09-1+b1	Non-interactive ssh
sysvinit-utils	3.06-4+b1	System-V-like utilit
tar	1.34+dfsg-1.2+deb12u1+b1	GNU version of the
tcl-expect:amd64	5.45.4-2+ci202403211247+astra1+b1	Automates interac
tcl8.6	8.6.13+dfsg-2+b1	Tcl (the Tool Comm
tofu	1.6.2	OpenTofu is an infr
tzdata	2024a-4	time zone and day
usrmerge	37~deb12u1+b1	Convert the syster
util-linux	2.38.1-5+deb12u1+astra13+b1	miscellaneous syst
util-linux-extra	2.38.1-5+deb12u1+astra13+b1	interactive login to
zlib1g:amd64	1:1.2.13.dfsg-1+b1	compression librar
zlib1g-dev:amd64	1:1.2.13.dfsg-1+b1	compression librar

Пакеты Python

Пакет
adal
aiohappyeyeballs
aiohttp
aiosignal
ansible-core
ansible-runner
ansible-sign @ file:///docker_share/dist/ansible_sign-0.1.1-py3-none-any.whl
anyio
appdirs
applicationinsights

Таблица 12 – продолжение с

Пакет
argcomplete
attrs
awxkit
azure-cli-core
azure-cli-telemetry
azure-common
azure-containerregistry
azure-core
azure-identity
azure-iot-hub
azure-keyvault
azure-keyvault-certificates
azure-keyvault-keys
azure-keyvault-secrets
azure-mgmt-apimanagement
azure-mgmt-authorization
azure-mgmt-automation
azure-mgmt-batch
azure-mgmt-cdn
azure-mgmt-compute
azure-mgmt-containerinstance
azure-mgmt-containerregistry
azure-mgmt-containerservice
azure-mgmt-core
azure-mgmt-cosmosdb
azure-mgmt-datafactory
azure-mgmt-devtestlabs
azure-mgmt-dns
azure-mgmt-eventhub
azure-mgmt-hdinsight
azure-mgmt-iothub
azure-mgmt-keyvault
azure-mgmt-loganalytics
azure-mgmt-managedservices
azure-mgmt-managementgroups
azure-mgmt-marketplaceordering
azure-mgmt-monitor
azure-mgmt-network
azure-mgmt-notificationhubs
azure-mgmt-nspkg
azure-mgmt-privatedns
azure-mgmt-rdbms
azure-mgmt-recoveryservices
azure-mgmt-recoveryservicesbackup
azure-mgmt-redis
azure-mgmt-resource
azure-mgmt-search
azure-mgmt-servicebus
azure-mgmt-sql
azure-mgmt-storage
azure-mgmt-trafficmanager
azure-mgmt-web
azure-nspkg
azure-storage-blob
bcrypt
beautifulsoup4
blinker

Таблица 12 – продолжение с

Пакет
boto3
botocore
cachetools
certifi
cffi
chardet
charset-normalizer
cryptography
decorator
Deprecated
deprecation
distlib
distro
dnspython
dogpile.cache
dumb-init
durationpy
frozenlist
future
google-api-core
googleapis-common-protos
google-auth
google-cloud-core
google-cloud-storage
google-crc32c
google-resumable-media
h11
h2
hpack
html5lib
httpcore
httplib2
httpx
humanfriendly
hyperframe
idna
importlib_metadata
iso8601
isodate
Jinja2
jmespath
jsonpatch
jsonpointer
jsonschema
jsonschema-specifications
keystoneauth1
knack
kubernetes
lockfile
lxml
Mako
MarkupSafe
microsoft-kiota-abstractions
microsoft-kiota-authentication-azure
microsoft-kiota-http
microsoft-kiota-serialization-form
microsoft-kiota-serialization-json

Таблица 12 – продолжение с

Пакет
microsoft-kiota-serialization-multipart
microsoft-kiota-serialization-text
more-itertools
msal
msal-extensions
msgraph-core
msgraph-sdk
msrest
msrestazure
multidict
munch
netaddr
netifaces
nsx_policy_python_sdk @ file://localhost//tmp/pip-req-build-gk_7_hlr/lib/nsx-policy-python-sdk/nsx_policy_p
nsx_python_sdk @ file://localhost//tmp/pip-req-build-gk_7_hlr/lib/nsx-python-sdk/nsx_python_sdk-4.2.0-py
nsx-vmc-aws-integration-python-sdk @ file://localhost//tmp/pip-req-build-gk_7_hlr/lib/nsx-vmc-aws-integra
nsx-vmc-policy-python-sdk @ file://localhost//tmp/pip-req-build-gk_7_hlr/lib/nsx-vmc-policy-python-sdk/nsx
ntlm-auth
oauthlib
openstacksdk
opentelemetry-api
opentelemetry-sdk
opentelemetry-semantic-conventions
os-service-types
ovirt-engine-sdk-python
ovirt-imageio
packaging
paramiko
pbr
pexpect
pkginfo
platformdirs
portalocker
propcache
protobuf
proto-plus
psutil
ptyprocess
pyasn1
pyasn1-modules
pycparser
pycurl
Pygments
PyJWT
PyNaCl
pyOpenSSL
pyparsing
pypsrp @ file:///docker_share/dist/pypsrp-0.8.1-py3-none-any.whl
PySocks
pyspnego
python-daemon
python-dateutil
python-gnupg
pytz
pyvmomi
pywinrm
PyYAML

Таблица 12 – продолжение с

Пакет
referencing
requests
requestsexceptions
requests-ntlm
requests-oauthlib
resolvelib
rpds-py
rsa
s3transfer
six
sniffio
soupsieve
std-uritemplate
stevedore
tabulate
textfsm
ttp
typing_extensions
uamqp
urllib3
vmwarecloud-aws @ file://localhost//tmp/pip-req-build-gk_7_hlr/lib/vmwarecloud-aws/vmwarecloud_aws-1.
vmwarecloud-draas @ file://localhost//tmp/pip-req-build-gk_7_hlr/lib/vmwarecloud-draas/vmwarecloud_dra
vmware-vapi-common-client
vmware-vapi-runtime
vmware_vcenter
vsphere-automation-sdk @ git+https://github.com/vmware/vsphere-automation-sdk-python.git@145786b8
webencodings
websocket-client
wrapt
xmltodict
yarl
zipp

Коллекции Ansible

Коллекция	Версия
amazon.aws	9.0.0
ansible.posix	1.5.4
ansible.utils	3.0.0
awx.awx	24.6.1
azure.azcollection	3.0.0
cloud.terraform	3.0.0
community.vmware	5.0.1
google.cloud	1.4.1
kubernetes.core	5.0.0
kubevirt.core	2.1.0
openstack.cloud	2.2.0
ovirt.ovirt	3.2.0
redhatinsights.insights	1.3.0
theforeman.foreman	4.2.0
vmware.vmware	1.9.0

Образы aa-minimal-de

1.0.2

Пакеты APT

Пакет	Версия	Описание
adduser	3.134+astra.se10	add and remove users
ansible-core	2.15.10-1	Configuration manage
apt	2.6.1+ci202312061551+astra2+b1	commandline package
apt-transport-https	2.6.1+ci202312061551+astra2+b1	transitional package fo
astra-archive-keyring	2023.12	GnuPG archive keys of
astra-version	8.1.41+v1.8.1.12	Update Astra version
base-files	12.4+deb12u1.astra6	Debian base system m
base-passwd	3.6.1+b1	Debian base system m
bash	5.2.15-2.astra.se1+b2	GNU Bourne Again SHe
bsdutils	1:2.38.1-5+deb12u1+astra13+b1	basic utilities from 4.4
ca-certificates	20230311+b2	Common CA certificate
ca-certificates-java	20230620~deb12u1+b1	Common CA certificate
coreutils	9.1-1+ci202402121122+astra2+b1	GNU core utilities
curl	7.88.1-10+deb12u5+b2	command line tool for
dash	0.5.12-2.astra1	POSIX-compliant shell
debconf	1.5.82+b1	Debian configuration m
debianutils	5.7-0.5~deb12u1+b1	Miscellaneous utilities
diffutils	1:3.8-4+b1	File comparison utilitie
dirmngr	2.2.40-1.1.astra6+b2	GNU privacy guard - ne
dpkg	1.21.22.astra.se3+b2	Debian package mana
e2fsprogs	1.47.0-2+b1	ext2/ext3/ext4 file syst
expect	5.45.4-2+ci202403211247+astra1+b1	Automates interactive
findutils	4.9.0-4+b1	utilities for finding files
fontconfig-config	2.14.2-4ubuntu1+b1	generic font configurat
fonts-noto-core	20201225-1	No Tofu font families w
gawk	1:5.2.1-2+b1	GNU awk, a pattern sc
gcc-12-base:amd64	12.2.0-14.astra3+b1	GCC, the GNU Compile
gnupg	2.2.40-1.1.astra6+b2	GNU privacy guard - a
gnupg-l10n	2.2.40-1.1.astra6+b2	GNU privacy guard - lo
gnupg-utils	2.2.40-1.1.astra6+b2	GNU privacy guard - ut
gpg	2.2.40-1.1.astra6+b2	GNU Privacy Guard – m
gpg-agent	2.2.40-1.1.astra6+b2	GNU privacy guard - cr
gpg-wks-client	2.2.40-1.1.astra6+b2	GNU privacy guard - W
gpg-wks-server	2.2.40-1.1.astra6+b2	GNU privacy guard - W
gpgconf	2.2.40-1.1.astra6+b2	GNU privacy guard - co
gpgsm	2.2.40-1.1.astra6+b2	GNU privacy guard - S/
gpgv	2.2.40-1.1.astra6+b2	GNU privacy guard - si
grep	3.8-5+b1	GNU grep, egrep and f
gzip	1.12-1+b1	GNU compression utilit
hostname	3.23+nmu1+b1	utility to set/show the
ieee-data	20220827.1	OUI and IAB listings
init-system-helpers	1.65.2+b1	helper tools for all init
java-common	0.74+b1	Base package for Java
libacl1:amd64	2.3.1-3+b1	access control list - sha
libapt-pkg6.0:amd64	2.6.1+ci202312061551+astra2+b1	package management
libasound2:amd64	1.2.8-1+b1	shared library for ALSA
libasound2-data	1.2.8-1+b1	Configuration files and
libassuan0:amd64	2.5.5-5+b1	IPC library for the Gnu
libattr1:amd64	1:2.5.1-4+ci202403112058+astra3+b1	extended attribute har
libaudit-common	1:3.0.9-1+ci202401241435+astra3+b2	Dynamic library for sec
libaudit1:amd64	1:3.0.9-1+ci202401241435+astra3+b2	Dynamic library for sec
libavahi-client3:amd64	0.8-10+b1	Avahi client library

Таблица 13 – продолжение с предыдущей

Пакет	Версия	Описание
libavahi-common-data:amd64	0.8-10+b1	Avahi common data fil
libavahi-common3:amd64	0.8-10+b1	Avahi common library
libblkid1:amd64	2.38.1-5+deb12u1+astra13+b1	block device ID library
libbrotli1:amd64	1.0.9-2+b1	library implementing b
libbsd0:amd64	0.11.7-2+b1	utility functions from B
libbz2-1.0:amd64	1.0.8-5+b1	high-quality block-sort
libc-bin	2.36-9+deb12u7+ci202405171200+astra5+b1	GNU C Library: Binaries
libc6:amd64	2.36-9+deb12u7+ci202405171200+astra5+b1	GNU C Library: Shared
libcap-ng0:amd64	0.8.3-1+b2	alternate POSIX capab
libcap2:amd64	1:2.66-4+b1	POSIX 1003.1e capabil
libcbor0.8:amd64	0.8.0-2+b1	library for parsing and
libcom-err2:amd64	1.47.0-2+b1	common error descript
libcrypt1:amd64	1:4.4.33-2+b1	libcrypt shared library
libcups2:amd64	2.4.10-1astra.se2	Common UNIX Printing
libcurl4:amd64	7.88.1-10+deb12u5+b2	easy-to-use client-side
libdb5.3:amd64	5.3.28+dfsg2-1+ci202311141605+astra1+b1	Berkeley v5.3 Databas
libdbus-1-3:amd64	1.14.8-2~deb12u1+ci12+b2	simple interprocess me
libdebconfclient0:amd64	0.270astra1+b1	Debian Configuration M
libedit2:amd64	3.1-20221030-2+b1	BSD editline and histor
libexpat1:amd64	2.5.0-1+ci202405221303+astra2	XML parsing C library -
libext2fs2:amd64	1.47.0-2+b1	ext2/ext3/ext4 file syst
libffi8:amd64	3.4.4-1+b1	Foreign Function Interf
libfido2-1:amd64	1.12.0-2+b2	library for generating a
libfile-find-rule-perl	0.34-3+b1	module to search for fi
libfontconfig1:amd64	2.14.2-4ubuntu1+b1	generic font configurat
libfreetype6:amd64	2.12.1+dfsg-5+b1	FreeType 2 font engine
libgcc-s1:amd64	12.2.0-14.astra3+b1	GCC support library
libgcrypt20:amd64	1.10.1-3.astra2+b1	LGPL Crypto library - ru
libgdbm-compat4:amd64	1.23-3+b1	GNU dbm database rou
libgdbm6:amd64	1.23-3+b1	GNU dbm database rou
libglib2.0-0:amd64	2.74.6-2+deb12u2+b1	GLib library of C routin
libgmp10:amd64	2:6.2.1+dfsg1-1.1+b1	Multiprecision arithmet
libgnutls30:amd64	3.7.9-2+deb12u2+b1	GNU TLS library - main
libgost	2.0.2-5+ci4+b1	This package contains
libgpg-error0:amd64	1.46-1+b1	GnuPG development ru
libgraphite2-3:amd64	1.3.14-1+b1	Font rendering engine
libgssapi-krb5-2:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerberos runtime l
libharfbuzz0b:amd64	6.0.0+dfsg-3+b1	OpenType text shaping
libhogweed6:amd64	3.8.1-2+b1	low level cryptographi
libidn2-0:amd64	2.3.3-1+b1	Internationalized doma
libjpeg62-turbo:amd64	1:2.1.5-2+b1	libjpeg-turbo JPEG runt
libk5crypto3:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerberos runtime l
libkeyutils1:amd64	1.6.3-2+b1	Linux Key Managemen
libkrb5-3:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerberos runtime l
libkrb5support0:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerberos runtime l
libksba8:amd64	1.6.3-2+b1	X.509 and CMS suppor
liblcms2-2:amd64	2.14-2+b1	Little CMS 2 color man
libldap-2.5-0:amd64	2.5.13+dfsg-5+ci202405061914+astra2	OpenLDAP libraries
liblocale-gettext-perl	1.07-5+b1	module using libc func
liblz4-1:amd64	1.9.4-1+b1	Fast LZ compression al
liblzma5:amd64	5.4.1-0.2+b1	XZ-format compression
libmd0:amd64	1.0.4-2+b1	message digest functio
libmount1:amd64	2.38.1-5+deb12u1+astra13+b1	device mounting librar
libmpfr6:amd64	4.2.0-1+b1	multiple precision float
libncursesw6:amd64	6.4-4+b1	shared libraries for ter
libnettle8:amd64	3.8.1-2+b1	low level cryptographi
libnghttp2-14:amd64	1.52.0-1+deb12u1+ci1+b1	library implementing H

Таблица 13 – продолжение с предыдущей

Пакет	Версия	Описание
libnpth0:amd64	1.6-3+b1	replacement for GNU P
libnsl2:amd64	1.3.0-2+b1	Public client interface f
libnspr4:amd64	2:4.35-1+b1	NetScape Portable Run
libnss3:amd64	2:3.100-1+ci1	Network Security Serv
libnumber-compare-perl	0.03-3+b1	module for performing
libp11-kit0:amd64	0.24.1-2+b1	library for loading and
libpam-modules:amd64	1.5.2-6+deb12u1.astra.se21+b1	Pluggable Authenticati
libpam-modules-bin	1.5.2-6+deb12u1.astra.se21+b1	Pluggable Authenticati
libpam-runtime	1.5.2-6+deb12u1.astra.se21+b1	Runtime support for th
libpam0g:amd64	1.5.2-6+deb12u1.astra.se21+b1	Pluggable Authenticati
libparsec-base3	3.9+ci132	base library for PARSE
libparsec-cap3	3.9+ci132	capabilities library for
libpcre2-8-0:amd64	10.42-1+b1	New Perl Compatible R
libpcsclite1:amd64	1.9.9-2+b1	Middleware to access a
libpdp	3.9+ci132	library for PARSEC DP s
libperl5.36:amd64	5.36.0-7+deb12u1+ci202404171352+astra2	shared Perl library
libpng16-16:amd64	1.6.39-2+b1	PNG library - runtime (
libproc2-0:amd64	2:4.0.2-3+b1	library for accessing pr
libpsl5:amd64	0.21.2-1+b1	Library for Public Suffix
libpython3-stdlib:amd64	3.11.2-1+b1	interactive high-level o
libpython3.11-minimal:amd64	3.11.2-6astra4+b1	Minimal subset of the I
libpython3.11-stdlib:amd64	3.11.2-6astra4+b1	Interactive high-level o
libreadline8:amd64	8.2-1.3+b1	GNU readline and histo
librtmp1:amd64	2.4+20151223.gitfa8646d.1-2+b1	toolkit for RTMP stream
libsasl2-2:amd64	2.1.28+dfsg-10+b2	Cyrus SASL - authentic
libsasl2-modules-db:amd64	2.1.28+dfsg-10+b2	Cyrus SASL - pluggable
libseccomp2:amd64	2.5.4-1+b2	high level interface to
libselinux1:amd64	3.4-1+b1	SELinux runtime share
libsemanage-common	3.4-1+b1	Common files for SELin
libsemanage2:amd64	3.4-1+b1	SELinux policy manage
libsepol2:amd64	3.4-2.1+b1	SELinux library for ma
libsigsegv2:amd64	2.14-1+b1	Library for handling pa
libsmartcols1:amd64	2.38.1-5+deb12u1+astra13+b1	smart column output a
libsodium23:amd64	1.0.18-1+b1	Network communicatio
libsqlite3-0:amd64	3.40.1-2+ci202405281356+astra2	SQLite 3 shared library
libss2:amd64	1.47.0-2+b1	command-line interfac
libssh2-1:amd64	1.10.0-3+b2	SSH2 client-side library
libssl3:amd64	3.2.0-2-astra5+ci1	Secure Sockets Layer t
libstdc++6:amd64	12.2.0-14.astra3+b1	GNU Standard C++ Lib
libsystemd0:amd64	252.17-1~deb12u1astra.se1+ci13+b1	systemd utility library
libtasn1-6:amd64	4.19.0-2+b1	Manage ASN.1 structu
libtcl8.6:amd64	8.6.13+dfsg-2+b1	Tcl (the Tool Command
libtext-charwidth-perl:amd64	0.04-11+b1	get display widths of c
libtext-glob-perl	0.11-3+b1	Perl module for matchi
libtext-iconv-perl:amd64	1.7-8+b1	module to convert betw
libtext-wrapi18n-perl	0.06-10+b1	internationalized subst
libtinfo6:amd64	6.4-4+b1	shared low-level termi
libtirpc-common	1.3.3+ds-1+b1	transport-independent
libtirpc3:amd64	1.3.3+ds-1+b1	transport-independent
libudev1:amd64	252.17-1~deb12u1astra.se1+ci13+b1	libudev shared library
libunistring2:amd64	1.0-2+b1	Unicode string library f
libuuid1:amd64	2.38.1-5+deb12u1+astra13+b1	Universally Unique ID l
libxxhash0:amd64	0.8.1-1+b1	shared library for xxha
libyaml-0-2:amd64	0.2.5-1+b1	Fast YAML 1.1 parser a
libzstd1:amd64	1.5.4+dfsg2-5+b1	fast lossless compress
login	1:4.13+dfsg1-1.astra.se10+b1	system login tools
logsave	1.47.0-2+b1	save the output of a co

Таблица 13 – продолжение с предыдущей

Пакет	Версия	Описание
lsb-base	11.6	transitional package fo
mawk	1.3.4.20200120-3.1+b1	Pattern scanning and t
media-types	10.0.0	List of standard media
mount	2.38.1-5+deb12u1+astra13+b1	tools for mounting and
ncurses-base	6.4-4+b1	basic terminal type de
ncurses-bin	6.4-4+b1	terminal-related progra
netbase	6.4	Basic TCP/IP networkin
openjdk-17-jre-headless:amd64	17.0.11+9-1~deb12u1+b1	OpenJDK Java runtime,
openssh-client	1:9.6p1-2~deb10u1astra8se5	secure shell (SSH) clie
openssl	3.2.0-2-astra5+ci1	Secure Sockets Layer t
passwd	1:4.13+dfsg1-1.astra.se10+b1	change and administer
perl	5.36.0-7+deb12u1+ci202404171352+astra2	Larry Wall’s Practical E
perl-base	5.36.0-7+deb12u1+ci202404171352+astra2	minimal Perl system
perl-modules-5.36	5.36.0-7+deb12u1+ci202404171352+astra2	Core Perl modules
pinentry-curses	1.2.1-1+b1	curses-based PIN or pa
procps	2:4.0.2-3+b1	/proc file system utiliti
python3	3.11.2-1+b1	interactive high-level o
python3-bcrypt	3.2.2-1+b1	password hashing libra
python3-certifi	2022.9.24-1	root certificates for val
python3-cffi-backend:amd64	1.15.1-5+b1	Foreign Function Interf
python3-chardet	5.1.0+dfsg-2	Universal Character En
python3-charset-normalizer	3.0.1-2	charset, encoding and
python3-cryptography	38.0.4-3+ci202405171208+astra3+b1	Python library exposin
python3-daemon	2.3.2-1	library for making a Un
python3-distutils	3.11.2-3+b1	distutils package for Py
python3-dnspython	2.3.0-1+b1	DNS toolkit for Python
python3-httplib2	0.20.4-3	comprehensive HTTP c
python3-idna	3.3-1	Python IDNA2008 (RFC
python3-jinja2	3.1.2-1	small but fast and easy
python3-lib2to3	3.11.2-3+b1	Interactive high-level o
python3-lockfile	1:0.12.2-2.2	file locking library for P
python3-markupsafe	2.1.2-1+b1	HTML/XHTML/XML strin
python3-minimal	3.11.2-1+b1	minimal subset of the
python3-nacl	1.5.0-2+b1	Python bindings to libs
python3-netaddr	0.8.0-2	manipulation of variou
python3-ntlm-auth	1.4.0-2	NTLM low-level Python
python3-packaging	23.0-1	core utilities for python
python3-paramiko	2.12.0-2+b1	Make ssh v2 connectio
python3-pexpect	4.8.0-4	Python 3 module for au
python3-pip	23.0.1+dfsg-1	Python package install
python3-pkg-resources	66.1.1-1	Package Discovery and
python3-ptyprocess	0.7.0-5	Run a subprocess in a
python3-pyparsing	3.0.9-1	alternative to creating
python3-requests	2.28.1+dfsg-1+b1	elegant and simple HT
python3-requests-ntlm	1.1.0-3+b1	Adds support for NTLM
python3-resolvelib	0.9.0-2	module to resolve abst
python3-setuptools	66.1.1-1	Python3 Distutils Enha
python3-six	1.16.0-4	Python 2 and 3 compa
python3-urllib3	1.26.12-1+ci202406111901+astra2	HTTP library with threa
python3-wheel	0.38.4-2	built-package format fo
python3-winrm	0.3.0-4+deb12u1	Python 3 library for Wi
python3-xmltodict	0.13.0-1	Makes working with XM
python3-yaml	6.0-3+b1	YAML parser and emitt
python3.11	3.11.2-6astra4+b1	Interactive high-level o
python3.11-ansible-runner	2.4.0+aa1.1.0	Consistent Ansible Pyt
python3.11-minimal	3.11.2-6astra4+b1	Minimal subset of the I
python3.11-pypsrp	0.8.1+aa1.1.0	PowerShell Remoting P

Таблица 13 – продолжение с предыдущей

Пакет	Версия	Описание
python3.11-pyspnego	0.11.0+aa1.1.0	Windows Negotiate Au
python3.11-requests	2.32.3+aa1.1.0	Python HTTP for Huma
python3.11-urllib3	2.2.3+aa1.1.0	HTTP library with threa
readline-common	8.2-1.3+b1	GNU readline and histo
sed	4.9-1+b1	GNU stream editor for
sensible-utils	0.0.17+nmu1	Utilities for sensible alt
sshpass	1.09-1+b1	Non-interactive ssh pa
sysvinit-utils	3.06-4+b1	System-V-like utilities
tar	1.34+dfsg-1.2+deb12u1+b1	GNU version of the tar
tcl-expect:amd64	5.45.4-2+ci202403211247+astra1+b1	Automates interactive
tcl8.6	8.6.13+dfsg-2+b1	Tcl (the Tool Command
tzdata	2024a-4	time zone and daylight
usrmerge	37~deb12u1+b1	Convert the system to
util-linux	2.38.1-5+deb12u1+astra13+b1	miscellaneous system
util-linux-extra	2.38.1-5+deb12u1+astra13+b1	interactive login tools
zlib1g:amd64	1:1.2.13.dfsg-1+b1	compression library - r

Пакеты Python

Пакет	Версия
aiohttp	3.9.5
aiosignal	1.3.1
ansible-core	2.15.10
ansible_rulebook	1.1.2
ansible-runner	2.4.0
attrs	24.3.0
bcrypt	3.2.2
certifi	2022.9.24
chardet	5.1.0
charset-normalizer	3.0.1
cryptography	38.0.4
dnspython	2.3.0
dpath	2.2.0
drools_jpy	0.3.9
dumb-init	1.2.5
frozenlist	1.5.0
httplib2	0.20.4
idna	3.3
janus	1.2.0
Jinja2	3.1.2
jpy	1.1.0
jsonschema	4.23.0
jsonschema-specifications	2024.10.1
lockfile	0.12.2
MarkupSafe	2.1.2
multidict	6.1.0
netaddr	0.8.0
ntlm-auth	1.4.0
packaging	23.0
paramiko	2.12.0
pexpect	4.8.0
propcache	0.2.0
psycopg	3.2.3
psycopg-binary	3.2.3
ptyprocess	0.7.0

продолжается на следующей странице

Таблица 14 – продолжение с предыдущей страницы

Пакет	Версия
PyNaCl	1.5.0
pyparsing	3.0.9
pypsrp @ file:///docker_share/dist/pypsrp-0.8.1-py3-none-any.whl	
pyspnego	0.11.0
python-daemon	2.3.2
pywinrm	0.3.0
PyYAML	6.0
referencing	0.35.1
requests	2.32.3
requests-ntlm	1.1.0
resolvelib	0.9.0
rpds-py	0.21.0
six	1.16.0
typing_extensions	4.12.2
urllib3	2.2.3
watchdog	6.0.0
websockets	13.1
xmltodict	0.13.0
xxhash	3.5.0
yarl	1.17.1

Образы `aa-full-de`

`1.0.1`

Пакеты APT

Пакет	Версия	Описание
adduser	3.134+astra.se10	add and remove users
ansible-core	2.15.10-1	Configuration manage
apt	2.6.1+ci202312061551+astra2+b1	commandline package
apt-transport-https	2.6.1+ci202312061551+astra2+b1	transitional package fo
astra-archive-keyring	2023.12	GnuPG archive keys of
astra-version	8.1.41+v1.8.1.12	Update Astra version
base-files	12.4+deb12u1.astra6	Debian base system m
base-passwd	3.6.1+b1	Debian base system m
bash	5.2.15-2.astra.se1+b2	GNU Bourne Again SHe
bsdutils	1:2.38.1-5+deb12u1+astra13+b1	basic utilities from 4.4
ca-certificates	20230311+b2	Common CA certificate
ca-certificates-java	20230620~deb12u1+b1	Common CA certificate
coreutils	9.1-1+ci202402121122+astra2+b1	GNU core utilities
curl	7.88.1-10+deb12u5+b2	command line tool for
dash	0.5.12-2.astra1	POSIX-compliant shell
debconf	1.5.82+b1	Debian configuration m
debianutils	5.7-0.5~deb12u1+b1	Miscellaneous utilities
diffutils	1:3.8-4+b1	File comparison utilitie
dirmngr	2.2.40-1.1.astra6+b2	GNU privacy guard - n
dpkg	1.21.22.astra.se3+b2	Debian package mana
e2fsprogs	1.47.0-2+b1	ext2/ext3/ext4 file syst
expect	5.45.4-2+ci202403211247+astra1+b1	Automates interactive
findutils	4.9.0-4+b1	utilities for finding files
fontconfig-config	2.14.2-4ubuntu1+b1	generic font configurat
fonts-noto-core	20201225-1	No Tofu font families w
gawk	1:5.2.1-2+b1	GNU awk, a pattern sc
gcc-12-base:amd64	12.2.0-14.astra3+b1	GCC, the GNU Compile
gnupg	2.2.40-1.1.astra6+b2	GNU privacy guard - a

Таблица 15 – продолжение с предыдущей

Пакет	Версия	Описание
gnupg-l10n	2.2.40-1.1.astra6+b2	GNU privacy guard - lo
gnupg-utils	2.2.40-1.1.astra6+b2	GNU privacy guard - ut
gpg	2.2.40-1.1.astra6+b2	GNU Privacy Guard – m
gpg-agent	2.2.40-1.1.astra6+b2	GNU privacy guard - cr
gpg-wks-client	2.2.40-1.1.astra6+b2	GNU privacy guard - W
gpg-wks-server	2.2.40-1.1.astra6+b2	GNU privacy guard - W
gpgconf	2.2.40-1.1.astra6+b2	GNU privacy guard - co
gpgsm	2.2.40-1.1.astra6+b2	GNU privacy guard - S/
gpgv	2.2.40-1.1.astra6+b2	GNU privacy guard - si
grep	3.8-5+b1	GNU grep, egrep and f
gzip	1.12-1+b1	GNU compression utilit
hostname	3.23+nmu1+b1	utility to set/show the
ieee-data	20220827.1	OUI and IAB listings
init-system-helpers	1.65.2+b1	helper tools for all init
java-common	0.74+b1	Base package for Java
libacl1:amd64	2.3.1-3+b1	access control list - sha
libapt-pkg6.0:amd64	2.6.1+ci202312061551+astra2+b1	package management
libasound2:amd64	1.2.8-1+b1	shared library for ALSA
libasound2-data	1.2.8-1+b1	Configuration files and
libassuan0:amd64	2.5.5-5+b1	IPC library for the Gnu
libattr1:amd64	1:2.5.1-4+ci202403112058+astra3+b1	extended attribute har
libaudit-common	1:3.0.9-1+ci202401241435+astra3+b2	Dynamic library for sec
libaudit1:amd64	1:3.0.9-1+ci202401241435+astra3+b2	Dynamic library for sec
libavahi-client3:amd64	0.8-10+b1	Avahi client library
libavahi-common-data:amd64	0.8-10+b1	Avahi common data fil
libavahi-common3:amd64	0.8-10+b1	Avahi common library
libblkid1:amd64	2.38.1-5+deb12u1+astra13+b1	block device ID library
libbrotli1:amd64	1.0.9-2+b1	library implementing b
libbsd0:amd64	0.11.7-2+b1	utility functions from B
libbz2-1.0:amd64	1.0.8-5+b1	high-quality block-sort
libc-bin	2.36-9+deb12u7+ci202405171200+astra5+b1	GNU C Library: Binaries
libc6:amd64	2.36-9+deb12u7+ci202405171200+astra5+b1	GNU C Library: Shared
libcap-ng0:amd64	0.8.3-1+b2	alternate POSIX capab
libcap2:amd64	1:2.66-4+b1	POSIX 1003.1e capabil
libcbor0.8:amd64	0.8.0-2+b1	library for parsing and
libcom-err2:amd64	1.47.0-2+b1	common error descript
libcrypt1:amd64	1:4.4.33-2+b1	libcrypt shared library
libcups2:amd64	2.4.10-1astra.se2	Common UNIX Printing
libcurl4:amd64	7.88.1-10+deb12u5+b2	easy-to-use client-side
libdb5.3:amd64	5.3.28+dfsg2-1+ci202311141605+astra1+b1	Berkeley v5.3 Databas
libdbus-1-3:amd64	1.14.8-2~deb12u1+ci12+b2	simple interprocess me
libdebconfclient0:amd64	0.270astra1+b1	Debian Configuration M
libedit2:amd64	3.1-20221030-2+b1	BSD editline and histor
libexpat1:amd64	2.5.0-1+ci202405221303+astra2	XML parsing C library -
libext2fs2:amd64	1.47.0-2+b1	ext2/ext3/ext4 file syst
libffi8:amd64	3.4.4-1+b1	Foreign Function Interf
libfido2-1:amd64	1.12.0-2+b2	library for generating a
libfile-find-rule-perl	0.34-3+b1	module to search for fi
libfontconfig1:amd64	2.14.2-4ubuntu1+b1	generic font configurat
libfreetype6:amd64	2.12.1+dfsg-5+b1	FreeType 2 font engine
libgcc-s1:amd64	12.2.0-14.astra3+b1	GCC support library
libgcrypt20:amd64	1.10.1-3.astra2+b1	LGPL Crypto library - ru
libgdbm-compat4:amd64	1.23-3+b1	GNU dbm database rou
libgdbm6:amd64	1.23-3+b1	GNU dbm database rou
libglib2.0-0:amd64	2.74.6-2+deb12u2+b1	GLib library of C routin
libgmp10:amd64	2:6.2.1+dfsg1-1.1+b1	Multiprecision arithmet
libgnutls30:amd64	3.7.9-2+deb12u2+b1	GNU TLS library - main

Таблица 15 – продолжение с предыдущей

Пакет	Версия	Описание
libgost	2.0.2-5+ci4+b1	This package contains
libgpg-error0:amd64	1.46-1+b1	GnuPG development ru
libgraphite2-3:amd64	1.3.14-1+b1	Font rendering engine
libgssapi-krb5-2:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerberos runtime l
libharfbuzz0b:amd64	6.0.0+dfsg-3+b1	OpenType text shaping
libhogweed6:amd64	3.8.1-2+b1	low level cryptographi
libidn2-0:amd64	2.3.3-1+b1	Internationalized doma
libjpeg62-turbo:amd64	1:2.1.5-2+b1	libjpeg-turbo JPEG runt
libk5crypto3:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerberos runtime l
libkeyutils1:amd64	1.6.3-2+b1	Linux Key Managemen
libkrb5-3:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerberos runtime l
libkrb5support0:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerberos runtime l
libksba8:amd64	1.6.3-2+b1	X.509 and CMS suppor
liblcms2-2:amd64	2.14-2+b1	Little CMS 2 color man
libldap-2.5-0:amd64	2.5.13+dfsg-5+ci202405061914+astra2	OpenLDAP libraries
liblocale-gettext-perl	1.07-5+b1	module using libc func
liblz4-1:amd64	1.9.4-1+b1	Fast LZ compression al
liblzma5:amd64	5.4.1-0.2+b1	XZ-format compressior
libmd0:amd64	1.0.4-2+b1	message digest functio
libmount1:amd64	2.38.1-5+deb12u1+astra13+b1	device mounting librar
libmpfr6:amd64	4.2.0-1+b1	multiple precision float
libncursesw6:amd64	6.4-4+b1	shared libraries for ter
libnettle8:amd64	3.8.1-2+b1	low level cryptographi
libnghttp2-14:amd64	1.52.0-1+deb12u1+ci1+b1	library implementing H
libnpth0:amd64	1.6-3+b1	replacement for GNU P
libnsl2:amd64	1.3.0-2+b1	Public client interface f
libnspr4:amd64	2:4.35-1+b1	NetScape Portable Run
libnss3:amd64	2:3.100-1+ci1	Network Security Serv
libnumber-compare-perl	0.03-3+b1	module for performing
libp11-kit0:amd64	0.24.1-2+b1	library for loading and
libpam-modules:amd64	1.5.2-6+deb12u1.astra.se21+b1	Pluggable Authenticati
libpam-modules-bin	1.5.2-6+deb12u1.astra.se21+b1	Pluggable Authenticati
libpam-runtime	1.5.2-6+deb12u1.astra.se21+b1	Runtime support for th
libpam0g:amd64	1.5.2-6+deb12u1.astra.se21+b1	Pluggable Authenticati
libparsec-base3	3.9+ci132	base library for PARSE
libparsec-cap3	3.9+ci132	capabilities library for
libpcre2-8-0:amd64	10.42-1+b1	New Perl Compatible R
libpcsclite1:amd64	1.9.9-2+b1	Middleware to access a
libpdp	3.9+ci132	library for PARSEC DP s
libperl5.36:amd64	5.36.0-7+deb12u1+ci202404171352+astra2	shared Perl library
libpng16-16:amd64	1.6.39-2+b1	PNG library - runtime (
libproc2-0:amd64	2:4.0.2-3+b1	library for accessing pr
libpsl5:amd64	0.21.2-1+b1	Library for Public Suffix
libpython3-stdlib:amd64	3.11.2-1+b1	interactive high-level o
libpython3.11-minimal:amd64	3.11.2-6astra4+b1	Minimal subset of the I
libpython3.11-stdlib:amd64	3.11.2-6astra4+b1	Interactive high-level o
libreadline8:amd64	8.2-1.3+b1	GNU readline and histo
librtmp1:amd64	2.4+20151223.gitfa8646d.1-2+b1	toolkit for RTMP stream
libsasl2-2:amd64	2.1.28+dfsg-10+b2	Cyrus SASL - authentic
libsasl2-modules-db:amd64	2.1.28+dfsg-10+b2	Cyrus SASL - pluggable
libseccomp2:amd64	2.5.4-1+b2	high level interface to
libselinux1:amd64	3.4-1+b1	SELinux runtime share
libsemanage-common	3.4-1+b1	Common files for SELin
libsemanage2:amd64	3.4-1+b1	SELinux policy manage
libsepol2:amd64	3.4-2.1+b1	SELinux library for ma
libsigsegv2:amd64	2.14-1+b1	Library for handling pa
libsmartcols1:amd64	2.38.1-5+deb12u1+astra13+b1	smart column output a

Таблица 15 – продолжение с предыдущей

Пакет	Версия	Описание
libsodium23:amd64	1.0.18-1+b1	Network communicatio
libsqlite3-0:amd64	3.40.1-2+ci202405281356+astra2	SQLite 3 shared library
libss2:amd64	1.47.0-2+b1	command-line interfac
libssh2-1:amd64	1.10.0-3+b2	SSH2 client-side library
libssl3:amd64	3.2.0-2-astra5+ci1	Secure Sockets Layer t
libstdc++6:amd64	12.2.0-14.astra3+b1	GNU Standard C++ Lib
libsystemd0:amd64	252.17-1~deb12u1astra.se1+ci13+b1	systemd utility library
libtasn1-6:amd64	4.19.0-2+b1	Manage ASN.1 structu
libtcl8.6:amd64	8.6.13+dfsg-2+b1	Tcl (the Tool Command
libtext-charwidth-perl:amd64	0.04-11+b1	get display widths of c
libtext-glob-perl	0.11-3+b1	Perl module for matchi
libtext-iconv-perl:amd64	1.7-8+b1	module to convert betw
libtext-wrapi18n-perl	0.06-10+b1	internationalized subst
libtinfo6:amd64	6.4-4+b1	shared low-level termi
libtirpc-common	1.3.3+ds-1+b1	transport-independent
libtirpc3:amd64	1.3.3+ds-1+b1	transport-independent
libudev1:amd64	252.17-1~deb12u1astra.se1+ci13+b1	libudev shared library
libunistring2:amd64	1.0-2+b1	Unicode string library f
libuuid1:amd64	2.38.1-5+deb12u1+astra13+b1	Universally Unique ID l
libxxhash0:amd64	0.8.1-1+b1	shared library for xxha
libyaml-0-2:amd64	0.2.5-1+b1	Fast YAML 1.1 parser a
libzstd1:amd64	1.5.4+dfsg2-5+b1	fast lossless compress
login	1:4.13+dfsg1-1.astra.se10+b1	system login tools
logsave	1.47.0-2+b1	save the output of a co
lsb-base	11.6	transitional package fo
mawk	1.3.4.20200120-3.1+b1	Pattern scanning and t
media-types	10.0.0	List of standard media
mount	2.38.1-5+deb12u1+astra13+b1	tools for mounting and
ncurses-base	6.4-4+b1	basic terminal type de
ncurses-bin	6.4-4+b1	terminal-related progra
netbase	6.4	Basic TCP/IP networkin
openjdk-17-jre-headless:amd64	17.0.11+9-1~deb12u1+b1	OpenJDK Java runtime,
openssh-client	1:9.6p1-2~deb10u1astra8se5	secure shell (SSH) clie
openssl	3.2.0-2-astra5+ci1	Secure Sockets Layer t
passwd	1:4.13+dfsg1-1.astra.se10+b1	change and administe
perl	5.36.0-7+deb12u1+ci202404171352+astra2	Larry Wall’s Practical E
perl-base	5.36.0-7+deb12u1+ci202404171352+astra2	minimal Perl system
perl-modules-5.36	5.36.0-7+deb12u1+ci202404171352+astra2	Core Perl modules
pinentry-curses	1.2.1-1+b1	curses-based PIN or pa
procps	2:4.0.2-3+b1	/proc file system utiliti
python3	3.11.2-1+b1	interactive high-level o
python3-bcrypt	3.2.2-1+b1	password hashing libra
python3-certifi	2022.9.24-1	root certificates for val
python3-cffi-backend:amd64	1.15.1-5+b1	Foreign Function Interf
python3-chardet	5.1.0+dfsg-2	Universal Character En
python3-charset-normalizer	3.0.1-2	charset, encoding and
python3-cryptography	38.0.4-3+ci202405171208+astra3+b1	Python library exposin
python3-daemon	2.3.2-1	library for making a Un
python3-distutils	3.11.2-3+b1	distutils package for Py
python3-dnspython	2.3.0-1+b1	DNS toolkit for Python
python3-httplib2	0.20.4-3	comprehensive HTTP c
python3-idna	3.3-1	Python IDNA2008 (RFC
python3-jinja2	3.1.2-1	small but fast and easy
python3-kafka	2.0.2-3	Pure Python client for A
python3-lib2to3	3.11.2-3+b1	Interactive high-level o
python3-lockfile	1:0.12.2-2.2	file locking library for P
python3-markupsafe	2.1.2-1+b1	HTML/XHTML/XML strin

Таблица 15 – продолжение с предыдущей

Пакет	Версия	Описание
python3-minimal	3.11.2-1+b1	minimal subset of the
python3-nacl	1.5.0-2+b1	Python bindings to libs
python3-netaddr	0.8.0-2	manipulation of variou
python3-ntlm-auth	1.4.0-2	NTLM low-level Python
python3-packaging	23.0-1	core utilities for pytho
python3-paramiko	2.12.0-2+b1	Make ssh v2 connectio
python3-pexpect	4.8.0-4	Python 3 module for a
python3-pip	23.0.1+dfsg-1	Python package install
python3-pkg-resources	66.1.1-1	Package Discovery and
python3-ptyprocess	0.7.0-5	Run a subprocess in a
python3-pyparsing	3.0.9-1	alternative to creating
python3-requests	2.28.1+dfsg-1+b1	elegant and simple HT
python3-requests-ntlm	1.1.0-3+b1	Adds support for NTLM
python3-resolvelib	0.9.0-2	module to resolve abst
python3-setuptools	66.1.1-1	Python3 Distutils Enha
python3-six	1.16.0-4	Python 2 and 3 compa
python3-systemd	235-1+b1	Python 3 bindings for s
python3-urllib3	1.26.12-1+ci202406111901+astra2	HTTP library with threa
python3-wheel	0.38.4-2	built-package format f
python3-winrm	0.3.0-4+deb12u1	Python 3 library for Wi
python3-xmltodict	0.13.0-1	Makes working with XM
python3-yaml	6.0-3+b1	YAML parser and emitt
python3.11	3.11.2-6astra4+b1	Interactive high-level o
python3.11-ansible-runner	2.4.0+aa1.1.0	Consistent Ansible Pyt
python3.11-minimal	3.11.2-6astra4+b1	Minimal subset of the
python3.11-pypsrp	0.8.1+aa1.1.0	PowerShell Remoting P
python3.11-pyspnego	0.11.0+aa1.1.0	Windows Negotiate Au
python3.11-requests	2.32.3+aa1.1.0	Python HTTP for Huma
python3.11-urllib3	2.2.3+aa1.1.0	HTTP library with threa
readline-common	8.2-1.3+b1	GNU readline and histo
sed	4.9-1+b1	GNU stream editor for
sensible-utils	0.0.17+nmu1	Utilities for sensible alt
sshpass	1.09-1+b1	Non-interactive ssh pa
sysvinit-utils	3.06-4+b1	System-V-like utilities
tar	1.34+dfsg-1.2+deb12u1+b1	GNU version of the tar
tcl-expect:amd64	5.45.4-2+ci202403211247+astra1+b1	Automates interactive
tcl8.6	8.6.13+dfsg-2+b1	Tcl (the Tool Command
tzdata	2024a-4	time zone and daylight
usrmerge	37~deb12u1+b1	Convert the system to
util-linux	2.38.1-5+deb12u1+astra13+b1	miscellaneous system
util-linux-extra	2.38.1-5+deb12u1+astra13+b1	interactive login tools
zlib1g:amd64	1:1.2.13.dfsg-1+b1	compression library - r

Пакеты Python

Пакет	Версия
aiobotocore	2.15.2
aiohttp	3.9.5
aioitertools	0.12.0
aiokafka	0.12.0
aiosignal	1.3.1
ansible-core	2.15.10
ansible_rulebook	1.1.2
ansible-runner	2.4.0
async-timeout	5.0.1

продолжается на следующей странице

Таблица 16 – продолжение с предыдущей страницы

Пакет	Версия
attrs	24.3.0
azure-core	1.31.0
azure-servicebus	7.13.0
bcrypt	3.2.2
botocore	1.35.36
certifi	2022.9.24
chardet	5.1.0
charset-normalizer	3.0.1
cryptography	38.0.4
dnspython	2.3.0
dpath	2.2.0
drools_jpy	0.3.9
dumb-init	1.2.5
frozenlist	1.5.0
httplib2	0.20.4
idna	3.3
isodate	0.7.2
janus	1.2.0
Jinja2	3.1.2
jmespath	1.0.1
jpy	1.1.0
jsonschema	4.23.0
jsonschema-specifications	2024.10.1
kafka-python	2.0.2
lockfile	0.12.2
MarkupSafe	2.1.2
multidict	6.1.0
netaddr	0.8.0
ntlm-auth	1.4.0
packaging	23.0
paramiko	2.12.0
pexpect	4.8.0
propcache	0.2.0
psycopg	3.2.3
psycopg-binary	3.2.3
psycopg-pool	3.2.4
ptyprocess	0.7.0
PyNaCl	1.5.0
pyparsing	3.0.9
pypsrp @ file:///docker_share/dist/pypsrp-0.8.1-py3-none-any.whl	
pyspnego	0.11.0
python-daemon	2.3.2
python-dateutil	2.9.0.post0
pywinrm	0.3.0
PyYAML	6.0
referencing	0.35.1
requests	2.32.3
requests-ntlm	1.1.0
resolvelib	0.9.0
rpds-py	0.21.0
six	1.16.0
systemd-python	235
typing_extensions	4.12.2
urllib3	2.2.3
watchdog	6.0.0
websockets	13.1
wrapt	1.16.0

продолжается на следующей странице

Таблица 16 – продолжение с предыдущей страницы

Пакет	Версия
xmltodict	0.13.0
xxhash	3.5.0
yarl	1.17.1

Образы aa-cdk

0.1.3

Пакеты APT

Пакет	Версия	Описани
adduser	3.134+astra.se10	add and r
ansible-builder	3.1.0+aa1.1.0	An Ansibl
ansible-compat	24.9.1+aa1.1.0	Ansible c
ansible-core	2.15.10-1	Configura
ansible-creator	24.10.1+aa2.0.1	A CLI tool
ansible-lint	24.9.2+aa1.1.0	Checks p
ansible-navigator	24.9.0+aa1.1.1	A text-ba
ansible-runner	2.4.0+aa1.1.1	Consister
ansible-sign	0.1.1+aa1.1.0	Ansible c
apt	2.6.1+ci202312061551+astra2+b1	command
apt-transport-https	2.6.1+ci202312061551+astra2+b1	transition
astra-archive-keyring	2023.12	GnuPG ar
astra-version	8.1.41+v1.8.1.12	Update A
base-files	12.4+deb12u1.astra6	Debian ba
base-passwd	3.6.1+b1	Debian ba
bash	5.2.15-2.astra.se1+b2	GNU Bou
bc	1.07.1-3+b1	GNU bc a
binutils	2.40-2+b1	GNU asse
binutils-common:amd64	2.40-2+b1	Common
binutils-x86-64-linux-gnu	2.40-2+b1	GNU bina
bsdutils	1:2.38.1-5+deb12u1+astra13+b1	basic utili
ca-certificates	20230311+b2	Common
conmon	2.1.6+ds1-1+b1	OCI conta
containernetworking-plugins	1.1.1+ds1-3+b1	standard
containers-storage	1.51.0+ds1-2+b1	CLI tools
coreutils	9.1-1+ci202402121122+astra2+b1	GNU core
cpp	4:12.2.0-3+b1	GNU C pr
cpp-12	12.2.0-14.astra3+b1	GNU C pr
crun	1.8.1-1+deb12u1+b1	lightweig
cryptsetup	2:2.6.1-4~deb12u2+b2	disk encr
cryptsetup-bin	2:2.6.1-4~deb12u2+b2	disk encr
curl	7.88.1-10+deb12u5+b2	command
dash	0.5.12-2.astra1	POSIX-co
dbus	1.14.8-2~deb12u1+ci12+b2	simple in
dbus-bin	1.14.8-2~deb12u1+ci12+b2	simple in
dbus-daemon	1.14.8-2~deb12u1+ci12+b2	simple in
dbus-session-bus-common	1.14.8-2~deb12u1+ci12+b2	simple in
dbus-system-bus-common	1.14.8-2~deb12u1+ci12+b2	simple in
dbus-user-session	1.14.8-2~deb12u1+ci12+b2	simple in
debconf	1.5.82+b1	Debian co
debianutils	5.7-0.5~deb12u1+b1	Miscellan
diffutils	1:3.8-4+b1	File comp
dirmngr	2.2.40-1.1.astra6+b2	GNU priva
dmsetup	2:1.02.185-2+b1	Linux Ker
dpkg	1.21.22.astra.se3+b2	Debian pa

Таблица 17 – продол

Пакет	Версия	Описани
e2fsprogs	1.47.0-2+b1	ext2/ext3
expect	5.45.4-2+ci202403211247+astra1+b1	Automate
findutils	4.9.0-4+b1	utilities fo
fuse-overlayfs	1.10-1+b1	implemen
fuse3	3.14.0-4+b1	Filesyster
gawk	1:5.2.1-2+b1	GNU awk
gcc	4:12.2.0-3+b1	GNU C co
gcc-12	12.2.0-14.astra3+b1	GNU C co
gcc-12-base:amd64	12.2.0-14.astra3+b1	GCC, the
git	1:2.39.2-1.1+ci3	fast, scal
git-man	1:2.39.2-1.1+ci3	fast, scal
gnupg	2.2.40-1.1.astra6+b2	GNU priv
gnupg-l10n	2.2.40-1.1.astra6+b2	GNU priv
gnupg-utils	2.2.40-1.1.astra6+b2	GNU priv
golang-github-containers-common	0.57.4+ds1-2+b1	Common
golang-github-containers-image	5.29.3-1	Configura
gpg	2.2.40-1.1.astra6+b2	GNU Priv
gpg-agent	2.2.40-1.1.astra6+b2	GNU priv
gpg-wks-client	2.2.40-1.1.astra6+b2	GNU priv
gpg-wks-server	2.2.40-1.1.astra6+b2	GNU priv
gpgconf	2.2.40-1.1.astra6+b2	GNU priv
gpgsm	2.2.40-1.1.astra6+b2	GNU priv
gpgv	2.2.40-1.1.astra6+b2	GNU priv
grep	3.8-5+b1	GNU grep
gzip	1.12-1+b1	GNU com
hostname	3.23+nmu1+b1	utility to
ieee-data	20220827.1	OUI and I
init-system-helpers	1.65.2+b1	helper to
iptables	1.8.9-2+b1	administr
less	590-2.1~deb12u2	pager pro
libacl1:amd64	2.3.1-3+b1	access co
libapparmor1:amd64	3.0.8-3+b1	changeha
libapt-pkg6.0:amd64	2.6.1+ci202312061551+astra2+b1	package
libargon2-1:amd64	0~20171227-0.3+deb12u1+b1	memory-
libasan8:amd64	12.2.0-14.astra3+b1	AddressS
libassuan0:amd64	2.5.5-5+b1	IPC librar
libatomic1:amd64	12.2.0-14.astra3+b1	support li
libattr1:amd64	1:2.5.1-4+ci202403112058+astra3+b1	extended
libaudit-common	1:3.0.9-1+ci202401241435+astra3+b2	Dynamic
libaudit1:amd64	1:3.0.9-1+ci202401241435+astra3+b2	Dynamic
libbinutils:amd64	2.40-2+b1	GNU bina
libblkid1:amd64	2.38.1-5+deb12u1+astra13+b1	block dev
libbrotli1:amd64	1.0.9-2+b1	library im
libbsd0:amd64	0.11.7-2+b1	utility fun
libbz2-1.0:amd64	1.0.8-5+b1	high-qual
libc-bin	2.36-9+deb12u7+ci202405171200+astra5+b1	GNU C Lib
libc-dev-bin	2.36-9+deb12u7+ci202405171200+astra5+b1	GNU C Lib
libc6:amd64	2.36-9+deb12u7+ci202405171200+astra5+b1	GNU C Lib
libc6-dev:amd64	2.36-9+deb12u7+ci202405171200+astra5+b1	GNU C Lib
libcap-ng0:amd64	0.8.3-1+b2	alternate
libcap2:amd64	1:2.66-4+b1	POSIX 10
libcbor0.8:amd64	0.8.0-2+b1	library fo
libcc1-0:amd64	12.2.0-14.astra3+b1	GCC cc1
libcom-err2:amd64	1.47.0-2+b1	common
libcrypt-dev:amd64	1:4.4.33-2+b1	libcrypt d
libcrypt1:amd64	1:4.4.33-2+b1	libcrypt s
libcryptsetup12:amd64	2:2.6.1-4~deb12u2+b2	disk encr

Таблица 17 – продол

Пакет	Версия	Описани
libctf-nobfd0:amd64	2.40-2+b1	Compact
libctf0:amd64	2.40-2+b1	Compact
libcurl3-gnutls:amd64	7.88.1-10+deb12u5+b2	easy-to-u
libcurl4:amd64	7.88.1-10+deb12u5+b2	easy-to-u
libdb5.3:amd64	5.3.28+dfsg2-1+ci202311141605+astra1+b1	Berkeley
libdbus-1-3:amd64	1.14.8-2~deb12u1+ci12+b2	simple in
libdebconfclient0:amd64	0.270astra1+b1	Debian C
libdevmapper1.02.1:amd64	2:1.02.185-2+b1	Linux Ker
libedit2:amd64	3.1-20221030-2+b1	BSD editl
liberror-perl	0.17029-2+b1	Perl modu
libexpat1:amd64	2.5.0-1+ci202405221303+astra2	XML pars
libexpat1-dev:amd64	2.5.0-1+ci202405221303+astra2	XML pars
libext2fs2:amd64	1.47.0-2+b1	ext2/ext3
libfdisk1:amd64	2.38.1-5+deb12u1+astra13+b1	fdisk part
libffi8:amd64	3.4.4-1+b1	Foreign F
libfido2-1:amd64	1.12.0-2+b2	library fo
libfile-find-rule-perl	0.34-3+b1	module to
libflygetexe	1.0.14+ci2+b1	libflygete
libfuse3-3:amd64	3.14.0-4+b1	Filesyster
libgcc-12-dev:amd64	12.2.0-14.astra3+b1	GCC supp
libgcc-s1:amd64	12.2.0-14.astra3+b1	GCC supp
libgcrypt20:amd64	1.10.1-3.astra2+b1	LGPL Cryp
libgdbm-compat4:amd64	1.23-3+b1	GNU dbm
libgdbm6:amd64	1.23-3+b1	GNU dbm
libglib2.0-0:amd64	2.74.6-2+deb12u2+b1	GLib libra
libgmp10:amd64	2:6.2.1+dfsg1-1.1+b1	Multiprec
libgnutls30:amd64	3.7.9-2+deb12u2+b1	GNU TLS
libgomp1:amd64	12.2.0-14.astra3+b1	GCC Ope
libgost	2.0.2-5+ci4+b1	This pack
libgpg-error0:amd64	1.46-1+b1	GnuPG de
libgpgme11:amd64	1.18.0-3+b1	GPGME -
libgprofng0:amd64	2.40-2+b1	GNU Next
libgssapi-krb5-2:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerbe
libhogweed6:amd64	3.8.1-2+b1	low level
libicu72:amd64	72.1-3+b1	Internatio
libidn2-0:amd64	2.3.3-1+b1	Internatio
libip4tc2:amd64	1.8.9-2+b1	netfilter l
libip6tc2:amd64	1.8.9-2+b1	netfilter l
libisl23:amd64	0.25-1.1+b1	manipula
libitm1:amd64	12.2.0-14.astra3+b1	GNU Tran
libjansson4:amd64	2.14-2+b1	C library
libjs-jquery	3.6.1+dfsg+~3.5.14-1+b1	JavaScrip
libjs-sphinxdoc	5.3.0-4+b1	JavaScrip
libjs-underscore	1.13.4~dfsg+~1.11.4-3	JavaScrip
libjson-c5:amd64	0.16-2+b1	JSON ma
libk5crypto3:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerbe
libkeyutils1:amd64	1.6.3-2+b1	Linux Key
libkmod2:amd64	30+20230519-1ubuntu3astra1+b1	libkmod s
libkrb5-3:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerbe
libkrb5support0:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerbe
libksba8:amd64	1.6.3-2+b1	X.509 an
libldap-2.5-0:amd64	2.5.13+dfsg-5+ci202405061914+astra2	OpenLDA
liblocale-gettext-perl	1.07-5+b1	module u
liblsan0:amd64	12.2.0-14.astra3+b1	LeakSani
liblua5.3-0:amd64	5.3.6-2+b1	Shared li
liblz4-1:amd64	1.9.4-1+b1	Fast LZ co
liblzma5:amd64	5.4.1-0.2+b1	XZ-forma

Таблица 17 – продол

Пакет	Версия	Описани
libmd0:amd64	1.0.4-2+b1	message
libmnl0:amd64	1.0.4-3+b1	minimalis
libmount1:amd64	2.38.1-5+deb12u1+astra13+b1	device m
libmpc3:amd64	1.3.1-1+b1	multiple
libmpfr6:amd64	4.2.0-1+b1	multiple
libncurses-dev:amd64	6.4-4+b1	develope
libncurses5-dev:amd64	6.4-4+b1	transition
libncurses6:amd64	6.4-4+b1	shared lib
libncursesw6:amd64	6.4-4+b1	shared lib
libnetfilter-conntrack3:amd64	1.0.9-3+b1	Netfilter
libnettle8:amd64	3.8.1-2+b1	low level
libnfnetlink0:amd64	1.0.2-2+b1	Netfilter
libnftnl11:amd64	1.2.4-2+b1	Netfilter
libnghttp2-14:amd64	1.52.0-1+deb12u1+ci1+b1	library im
libnl-3-200:amd64	3.7.0-0.2+b2	library fo
libnpth0:amd64	1.6-3+b1	replacem
libnsl-dev:amd64	1.3.0-2+b1	libnsl dev
libnsl2:amd64	1.3.0-2+b1	Public cli
libnuma1:amd64	2.0.16-1+b1	Libraries
libnumber-compare-perl	0.03-3+b1	module fo
libopendbx1	1.4.6-16+b1	Lightweig
libopenscap25	1.3.7+dfsg-1+deb12u1+b1	libraries e
libp11-kit0:amd64	0.24.1-2+b1	library fo
libpam-modules:amd64	1.5.2-6+deb12u1.astra.se21+b1	Pluggable
libpam-modules-bin	1.5.2-6+deb12u1.astra.se21+b1	Pluggable
libpam-runtime	1.5.2-6+deb12u1.astra.se21+b1	Runtime s
libpam-systemd:amd64	252.17-1~deb12u1astra.se1+ci13+b1	system a
libpam0g:amd64	1.5.2-6+deb12u1.astra.se21+b1	Pluggable
libparsec-aud3	3.9+ci132	audit libra
libparsec-base3	3.9+ci132	base libra
libparsec-cap3	3.9+ci132	capabiliti
libparsec-mic3	3.9+ci132	mandate
libpcre2-8-0:amd64	10.42-1+b1	New Perl
libpcre3:amd64	2:8.39-15+b1	Old Perl 5
libpdp	3.9+ci132	library fo
libperl5.36:amd64	5.36.0-7+deb12u1+ci202404171352+astra2	shared Pe
libpopt0:amd64	1.19+dfsg-1+b1	lib for pa
libproc2-0:amd64	2:4.0.2-3+b1	library fo
libprotobuf32:amd64	3.21.12-3+b1	protocol b
libpsl5:amd64	0.21.2-1+b1	Library fo
libpython3-dev:amd64	3.11.2-1+b1	header fil
libpython3-stdlib:amd64	3.11.2-1+b1	interactiv
libpython3.11:amd64	3.11.2-6astra4+b1	Shared Py
libpython3.11-dev:amd64	3.11.2-6astra4+b1	Header fi
libpython3.11-minimal:amd64	3.11.2-6astra4+b1	Minimal s
libpython3.11-stdlib:amd64	3.11.2-6astra4+b1	Interactiv
libqrencode4:amd64	4.1.1-1+b1	QR Code
libquadmath0:amd64	12.2.0-14.astra3+b1	GCC Qua
libreadline8:amd64	8.2-1.3+b1	GNU read
librpm9	4.18.0+dfsg-1+deb12u1+b2	RPM shar
librpmio9	4.18.0+dfsg-1+deb12u1+b2	RPM IO sh
librtmp1:amd64	2.4+20151223.gitfa8646d.1-2+b1	toolkit fo
libsasl2-2:amd64	2.1.28+dfsg-10+b2	Cyrus SA
libsasl2-modules-db:amd64	2.1.28+dfsg-10+b2	Cyrus SA
libseccomp2:amd64	2.5.4-1+b2	high leve
libselinux1:amd64	3.4-1+b1	SELinux
libsemanage-common	3.4-1+b1	Common

Таблица 17 – продол

Пакет	Версия	Описани
libsemanage2:amd64	3.4-1+b1	SELinux p
libsepol2:amd64	3.4-2.1+b1	SELinux l
libsigsegv2:amd64	2.14-1+b1	Library fo
libslirp0:amd64	4.7.0-1+b1	General p
libsmartcols1:amd64	2.38.1-5+deb12u1+astra13+b1	smart col
libsodium23:amd64	1.0.18-1+b1	Network c
libsqlite3-0:amd64	3.40.1-2+ci202405281356+astra2	SQLite 3 s
libss2:amd64	1.47.0-2+b1	command
libssh-4:amd64	0.10.6-0+deb12u1+b2	tiny C SS
libssh2-1:amd64	1.10.0-3+b2	SSH2 clie
libssl3:amd64	3.2.0-2-astra5+ci1	Secure So
libstdc++6:amd64	12.2.0-14.astra3+b1	GNU Stan
libsubid4:amd64	1:4.13+dfsg1-1.astra.se10+b1	subordina
libsystemd-shared:amd64	252.17-1~deb12u1astra.se1+ci13+b1	systemd
libsystemd0:amd64	252.17-1~deb12u1astra.se1+ci13+b1	systemd
libtasn1-6:amd64	4.19.0-2+b1	Manage A
libtcl8.6:amd64	8.6.13+dfsg-2+b1	Tcl (the To
libtext-charwidth-perl:amd64	0.04-11+b1	get displa
libtext-glob-perl	0.11-3+b1	Perl modu
libtext-iconv-perl:amd64	1.7-8+b1	module to
libtext-wrapi18n-perl	0.06-10+b1	internatio
libtinfo6:amd64	6.4-4+b1	shared lo
libtirpc-common	1.3.3+ds-1+b1	transport
libtirpc-dev:amd64	1.3.3+ds-1+b1	transport
libtirpc3:amd64	1.3.3+ds-1+b1	transport
libtsan2:amd64	12.2.0-14.astra3+b1	ThreadSa
libubsan1:amd64	12.2.0-14.astra3+b1	UBSan – u
libudev1:amd64	252.17-1~deb12u1astra.se1+ci13+b1	libudev s
libunistring2:amd64	1.0-2+b1	Unicode s
libuuid1:amd64	2.38.1-5+deb12u1+astra13+b1	Universal
libvirt-clients	10.5.0-1.astra.se06	Programs
libvirt0:amd64	10.5.0-1.astra.se06	library fo
libxml2:amd64	2.9.14+dfsg-1.3~deb12u1+ci202405141738+astra2	GNOME X
libxmlsec1:amd64	1.2.37-2+b2	XML secu
libxmlsec1-openssl:amd64	1.2.37-2+b2	Openssl e
libxslt1.1:amd64	1.1.35-1+b1	XSLT 1.0
libxtables12:amd64	1.8.9-2+b1	netfilter x
libxxhash0:amd64	0.8.1-1+b1	shared lib
libyajl2:amd64	2.1.0-3+deb12u2+b1	Yet Anoth
libyaml-0-2:amd64	0.2.5-1+b1	Fast YAML
libzstd1:amd64	1.5.4+dfsg2-5+b1	fast lossl
linux-libc-dev:amd64	5.4.0-54astra7+ci98	latest linu
linux-libc-dev-6.1.90-1:amd64	6.1.90-1.astra2+ci15	Linux Ker
login	1:4.13+dfsg1-1.astra.se10+b1	system lo
logsave	1.47.0-2+b1	save the
lsb-base	11.6	transition
lsb-release	12.0-1+b1	Linux Sta
mawk	1.3.4.20200120-3.1+b1	Pattern sc
media-types	10.0.0	List of sta
molecule	24.9.0+aa1.1.2	Molecule
molecule-libvirt	0.2.0+aa1.1.1	libvirt Mo
molecule-molecule-brest	1.1.0+aa1.1.0	Brest Mol
molecule-molecule-yandexcloud	0.3.1+aa1.1.0	YandexCl
molecule-plugins	23.5.3+aa1.1.0	Molecule
mount	2.38.1-5+deb12u1+astra13+b1	tools for m
ncurses-base	6.4-4+b1	basic ter
ncurses-bin	6.4-4+b1	terminal-

Таблица 17 – продол

Пакет	Версия	Описани
netbase	6.4	Basic TCP
openscap-common	1.3.7+dfsg-1+deb12u1+b1	OpenSCA
openscap-scanner	1.3.7+dfsg-1+deb12u1+b1	OpenSca
openssh-client	1:9.6p1-2~deb10u1astra8se5	secure sh
openssl	3.2.0-2-astra5+ci1	Secure So
oval-db	0.0.1+ci3	Install ov
passwd	1:4.13+dfsg1-1.astra.se10+b1	change a
perl	5.36.0-7+deb12u1+ci202404171352+astra2	Larry Wal
perl-base	5.36.0-7+deb12u1+ci202404171352+astra2	minimal F
perl-modules-5.36	5.36.0-7+deb12u1+ci202404171352+astra2	Core Perl
pinentry-curses	1.2.1-1+b1	curses-ba
podman	4.7.2.astra1	engine to
procps	2:4.0.2-3+b1	/proc file
pytest-ansible	24.9.0+aa1.1.0	Plugin for
python3	3.11.2-1+b1	interactiv
python3-attr	22.2.0-1	Attributes
python3-bcrypt	3.2.2-1+b1	password
python3-blinker	1.5-1	Fast, simp
python3-certifi	2022.9.24-1	root certi
python3-cffi	1.15.1-5+b1	Foreign F
python3-cffi-backend:amd64	1.15.1-5+b1	Foreign F
python3-chardet	5.1.0+dfsg-2	Universal
python3-charset-normalizer	3.0.1-2	charset, e
python3-click	8.1.3-2	Wrapper
python3-colorama	0.4.6-2	Cross-pla
python3-cryptography	38.0.4-3+ci202405171208+astra3+b1	Python lib
python3-daemon	2.3.2-1	library fo
python3-decorator	5.1.1-3	simplify u
python3-dev	3.11.2-1+b1	header fil
python3-distlib	0.3.6-1	low-level
python3-distro	1.8.0-1	Linux OS
python3-distutils	3.11.2-3+b1	distutils p
python3-dnspython	2.3.0-1+b1	DNS toolk
python3-docker	5.0.3-1	Python 3
python3-dogpile.cache	1.1.8-2	caching f
python3-execnet	1.9.0-1	rapid mul
python3-filelock	3.9.0-1	platform
python3-httplib2	0.20.4-3	comprehe
python3-idna	3.3-1	Python ID
python3-importlib-metadata	4.12.0-1	library to
python3-iniconfig	1.1.1-2	brain-dea
python3-iso8601	1.0.2-1	Python m
python3-jinja2	3.1.2-1	small but
python3-jmespath	1.0.1-1	JSON Mat
python3-json-pointer	2.3-2	resolve JS
python3-jsonpatch	1.32-2	library to
python3-jsonschema	4.10.3-1	An(other)
python3-jwt	2.6.0-1+b1	Python 3
python3-keystoneauth1	5.0.0-2	authentic
python3-lib2to3	3.11.2-3+b1	Interactiv
python3-libvirt	10.5.0-1astra.se01	libvirt Pyt
python3-lockfile	1:0.12.2-2.2	file lockin
python3-lxml:amd64	4.9.2-1+b1	pythonic
python3-mako	1.2.4+ds-1	fast and l
python3-markdown-it	2.1.0-5	Python po
python3-markupsafe	2.1.2-1+b1	HTML/XHT
python3-mdurl	0.1.2-1	Python po

Таблица 17 – продол

Пакет	Версия	Описани
python3-minimal	3.11.2-1+b1	minimal s
python3-more-itertools	8.10.0-2	library wi
python3-mypy-extensions	0.4.3-4	Experime
python3-nacl	1.5.0-2+b1	Python bi
python3-netaddr	0.8.0-2	manipula
python3-netifaces:amd64	0.11.0-2+b1	portable
python3-ntlm-auth	1.4.0-2	NTLM low
python3-oauthlib	3.2.2-1+b1	generic, s
python3-openssl	23.0.0-1+b1	Python 3
python3-os-service-types	1.7.0-2	lib for co
python3-packaging	23.0-1	core utilit
python3-paramiko	2.12.0-2+b1	Make ssh
python3-pathspec	0.11.0-1	utility libr
python3-pbr	5.10.0-2+b1	inject use
python3-pexpect	4.8.0-4	Python 3
python3-pip	23.0.1+dfsg-1	Python pa
python3-pkg-resources	66.1.1-1	Package
python3-platformdirs	2.6.0-1	determin
python3-pluggy	1.0.0+repack-1	plugin an
python3-ply	3.11-5+b1	Lex and Y
python3-protobuf	3.21.12-3+b1	Python 3
python3-ptyprocess	0.7.0-5	Run a sub
python3-py	1.11.0-1	Advanced
python3-pycparser	2.21-1	C parser
python3-pygments	2.14.0+dfsg-1	syntax hi
python3-pyparsing	3.0.9-1	alternativ
python3-pyrsistent:amd64	0.18.1-1+b1	persisten
python3-pytest	7.2.1-2	Simple, p
python3-pytest-xdist	3.1.0-1	xdist plug
python3-requests	2.28.1+dfsg-1+b1	elegant a
python3-requests-ntlm	1.1.0-3+b1	Adds sup
python3-requestsexceptions	1.4.0-3+b1	import ex
python3-resolvelib	0.9.0-2	module to
python3-rich	13.3.1-1	render ric
python3-ruamel.yaml.clib:amd64	0.2.7-1+b1	C version
python3-selinux	3.4-1+b1	Python3
python3-setuptools	66.1.1-1	Python3
python3-six	1.16.0-4	Python 2
python3-stevedore	4.0.2-2	manage d
python3-tblib	1.7.0-3	Python 3
python3-urllib3	1.26.12-1+ci202406111901+astra2	HTTP libr
python3-websocket	1.2.3-1	WebSock
python3-wheel	0.38.4-2	built-pack
python3-winrm	0.3.0-4+deb12u1	Python 3
python3-xmltodict	0.13.0-1	Makes wo
python3-yaml	6.0-3+b1	YAML par
python3-zipp	1.0.0-6	pathlib-co
python3.11	3.11.2-6astra4+b1	Interactiv
python3.11-aenum	3.1.15+aa1.1.0	Advanced
python3.11-bindep	2.11.0+aa1.2.0	Binary de
python3.11-black	24.10.0+aa1.1.0	The unco
python3.11-bracex	2.5.post1+aa1.1.0	Bash styl
python3.11-cachetools	5.5.0+aa1.1.0	Extensibl
python3.11-chardet	5.2.0+aa1.1.0	Universal
python3.11-click-help-colors	0.9.4+aa1.1.0	Colorizati
python3.11-cryptography	42.0.8+aa1.1.0	cryptogra
python3.11-dev	3.11.2-6astra4+b1	Header fi

Таблица 17 – продол

Пакет	Версия	Описани
python3.11-dicttoxml	1.7.16+aa1.1.0	Converts
python3.11-distlib	0.3.9+aa1.1.0	Distributi
python3.11-enrich	1.2.7+aa1.1.0	enrich
python3.11-filelock	3.16.1+aa1.1.0	A platforr
python3.11-googleapis-common-protos	1.65.0+aa1.1.0	Common
python3.11-grpcio	1.67.1+aa1.1.0	HTTP/2-b
python3.11-minimal	3.11.2-6astra4+b1	Minimal s
python3.11-onigurumacffi	1.3.0+aa1.1.0	python cf
python3.11-openstacksdk	4.1.0+aa1.1.0	An SDK fo
python3.11-packaging	24.2+aa1.1.0	Core utili
python3.11-parsley	1.3+aa1.1.0	Parsing a
python3.11-platformdirs	4.3.6+aa1.1.0	A small P
python3.11-pluggy	1.5.0+aa1.1.0	plugin an
python3.11-protobuf	4.25.5+aa1.1.0	Protocol B
python3.11-pyjwt	2.9.0+aa1.1.0	JSON Web
python3.11-pyone	6.6.3+aa1.1.0	Python Bi
python3.11-pyproject-api	1.8.0+aa1.1.0	API to int
python3.11-pypsrp	0.8.1+aa1.1.0	PowerShe
python3.11-pyspnego	0.11.0+aa1.1.0	Windows
python3.11-python-gnupg	0.5.2+aa1.1.0	A wrappe
python3.11-python-vagrant	1.0.0+aa1.1.0	Python bi
python3.11-requests	2.32.3+aa1.1.0	Python H
python3.11-ruamel.yaml	0.18.6+aa1.1.0	ruamel.ya
python3.11-subprocess-tee	0.4.2+aa1.1.0	subproce
python3.11-tox	4.23.2+aa1.1.0	tox is a g
python3.11-urllib3	2.2.3+aa1.1.0	HTTP libr
python3.11-virtualenv	20.27.1+aa1.1.0	Virtual Py
python3.11-wcmatch	10.0+aa1.1.0	Wildcard/
python3.11-yamllint	1.35.1+aa1.1.0	A linter fo
python3.11-yandexcloud	0.324.0+aa1.1.0	The Yand
readline-common	8.2-1.3+b1	GNU read
rpcsvc-proto	1.4.3-1+b1	RPC proto
runc	1.1.12~astra1+ci3+b1	Open Cor
sed	4.9-1+b1	GNU strea
sensible-utils	0.0.17+nmu1	Utilities fo
slirp4netns	1.2.0-1+b1	User-mod
sshpass	1.09-1+b1	Non-inter
systemd	252.17-1~deb12u1astra.se1+ci13+b1	system a
systemd-sysv	252.17-1~deb12u1astra.se1+ci13+b1	system a
sysvinit-utils	3.06-4+b1	System-V
tar	1.34+dfsg-1.2+deb12u1+b1	GNU vers
tcl-expect:amd64	5.45.4-2+ci202403211247+astra1+b1	Automate
tcl8.6	8.6.13+dfsg-2+b1	Tcl (the To
tox-ansible	24.9.0+aa1.1.0	A radical
tzdata	2024a-4	time zone
uidmap	1:4.13+dfsg1-1.astra.se10+b1	programs
usrmerge	37~deb12u1+b1	Convert t
util-linux	2.38.1-5+deb12u1+astra13+b1	miscellan
util-linux-extra	2.38.1-5+deb12u1+astra13+b1	interactiv
uuid-runtime	2.38.1-5+deb12u1+astra13+b1	runtime c
zlib1g:amd64	1:1.2.13.dfsg-1+b1	compress
zlib1g-dev:amd64	1:1.2.13.dfsg-1+b1	compress

Пакеты Python

Пакет
aenum
ansible-builder @ file:///docker_share/dist/ansible_builder-3.1.0-py3-none-any.whl
ansible-compat
ansible-core
ansible-creator @ file:///docker_share/dist/ansible_creator-24.10.1-py3-none-any.whl
ansible-lint @ file:///docker_share/dist/ansible_lint-24.9.2-py3-none-any.whl
ansible-navigator @ file:///docker_share/dist/ansible_navigator-24.9.0-py3-none-any.whl
ansible-pylibssh
ansible-runner @ file:///docker_share/dist/ansible_runner-2.4.0-py3-none-any.whl
ansible-sign @ file:///docker_share/dist/ansible_sign-0.1.1-py3-none-any.whl
attrs
bcrypt
bindep
black
blinker
bracex
cachetools
certifi
cffi
cfgv
chardet
charset-normalizer
click
click-help-colors
colorama
cryptography
Cython
decorator
dicttoxml
distlib
distro
dnspython
docker
dogpile.cache
dumb-init
enrich
execnet
filelock
googleapis-common-protos
grpcio
httplib2
identify
idna
importlib-metadata
iniconfig
iso8601
Jinja2
jmespath
jsonpatch
jsonpointer
jsonschema
jxmlease
keystoneauth1
libvirt-python
lockfile
lxml

продолжается на

Таблица 18 – продолжение с предыдущей страницы

Пакет
Mako
markdown-it-py
MarkupSafe
mdurl
molecule @ file:///docker_share/dist/molecule-24.9.0-py3-none-any.whl
molecule-libvirt @ file:///docker_share/dist/molecule_libvirt-0.2.0-py3-none-any.whl
molecule_molecule_brest @ file:///docker_share/dist/molecule_molecule_brest-1.1.0-py3-none-any.whl
molecule_molecule_yandexcloud @ file:///docker_share/dist/molecule_molecule_yandexcloud-0.3.1-py3-no
molecule-plugins @ file:///docker_share/dist/molecule_plugins-23.5.3-py3-none-any.whl
more-itertools
mypy-extensions
ncclient
netaddr
netifaces
nodeenv
ntlm-auth
oauthlib
onigurumacffi
openstacksdk
os-service-types
packaging
paramiko
Parsley
pathspec
pbr
pexpect
platformdirs
pluggy
ply
pre_commit
protobuf
ptyprocess
py
pycparser
Pygments
PyJWT
PyNaCl
pyone
pyOpenSSL
pyparsing
pyproject-api
pypsrp @ file:///docker_share/dist/pypsrp-0.8.1-py3-none-any.whl
pyrsistent
pyspnego
pytest
pytest-ansible @ file:///docker_share/dist/pytest_ansible-24.9.0-py3-none-any.whl
pytest-xdist
python-daemon
python-gnupg
python-vagrant
pywinrm
PyYAML
requests
requestsexceptions
requests-ntlm
resolvelib
rich

продолжается на

Таблица 18 – продолжение с предыдущей страницы

Пакет
ruamel.yaml
ruamel.yaml.clib
selinux
six
stevedore
subprocess-tee
tblib
textfsm
tox
tox-ansible @ file:///docker_share/dist/tox_ansible-24.9.0-py3-none-any.whl
ttp
urllib3
virtualenv
wcmatch
websocket-client
xmltodict
yamllint
yandexcloud
zipp

Коллекции Ansible

Коллекция	Версия
ansible.netcommon	7.1.0
ansible.posix	2.0.0
ansible.utils	5.1.2
astra.brest	4.0.3
astra.yandexcloud	0.4.0
community.crypto	2.24.0
community.general	10.3.0
community.libvirt	1.3.1

0.1.2

Пакеты APT

Пакет	Версия	Описани
adduser	3.134+astra.se10	add and r
ansible-builder	3.1.0+aa1.1.0	An Ansibl
ansible-compat	24.9.1+aa1.1.0	Ansible c
ansible-core	2.15.10-1	Configura
ansible-creator	24.10.1+aa2.0.1	A CLI tool
ansible-lint	24.9.2+aa1.1.0	Checks p
ansible-navigator	24.9.0+aa1.1.1	A text-ba
ansible-runner	2.4.0+aa1.1.1	Consister
ansible-sign	0.1.1+aa1.1.0	Ansible c
apt	2.6.1+ci202312061551+astra2+b1	command
apt-transport-https	2.6.1+ci202312061551+astra2+b1	transition
astra-archive-keyring	2023.12	GnuPG ar
astra-version	8.1.41+v1.8.1.12	Update A
base-files	12.4+deb12u1.astra6	Debian ba
base-passwd	3.6.1+b1	Debian ba
bash	5.2.15-2.astra.se1+b2	GNU Bou
bc	1.07.1-3+b1	GNU bc a

Таблица 19 – продол

Пакет	Версия	Описани
binutils	2.40-2+b1	GNU asse
binutils-common:amd64	2.40-2+b1	Common
binutils-x86-64-linux-gnu	2.40-2+b1	GNU bina
bsdutils	1:2.38.1-5+deb12u1+astra13+b1	basic utili
ca-certificates	20230311+b2	Common
conmon	2.1.6+ds1-1+b1	OCI conta
containernetworking-plugins	1.1.1+ds1-3+b1	standard
containers-storage	1.51.0+ds1-2+b1	CLI tools
coreutils	9.1-1+ci202402121122+astra2+b1	GNU core
cpp	4:12.2.0-3+b1	GNU C pr
cpp-12	12.2.0-14.astra3+b1	GNU C pr
crun	1.8.1-1+deb12u1+b1	lightweig
cryptsetup	2:2.6.1-4~deb12u2+b2	disk encr
cryptsetup-bin	2:2.6.1-4~deb12u2+b2	disk encr
curl	7.88.1-10+deb12u5+b2	command
dash	0.5.12-2.astra1	POSIX-co
dbus	1.14.8-2~deb12u1+ci12+b2	simple in
dbus-bin	1.14.8-2~deb12u1+ci12+b2	simple in
dbus-daemon	1.14.8-2~deb12u1+ci12+b2	simple in
dbus-session-bus-common	1.14.8-2~deb12u1+ci12+b2	simple in
dbus-system-bus-common	1.14.8-2~deb12u1+ci12+b2	simple in
dbus-user-session	1.14.8-2~deb12u1+ci12+b2	simple in
debconf	1.5.82+b1	Debian co
debianutils	5.7-0.5~deb12u1+b1	Miscellan
diffutils	1:3.8-4+b1	File comp
dirmngr	2.2.40-1.1.astra6+b2	GNU priva
dmsetup	2:1.02.185-2+b1	Linux Ker
dpkg	1.21.22.astra.se3+b2	Debian pa
e2fsprogs	1.47.0-2+b1	ext2/ext3
expect	5.45.4-2+ci202403211247+astra1+b1	Automate
findutils	4.9.0-4+b1	utilities fo
fuse-overlayfs	1.10-1+b1	implemen
fuse3	3.14.0-4+b1	Filesyster
gawk	1:5.2.1-2+b1	GNU awk
gcc	4:12.2.0-3+b1	GNU C co
gcc-12	12.2.0-14.astra3+b1	GNU C co
gcc-12-base:amd64	12.2.0-14.astra3+b1	GCC, the
git	1:2.39.2-1.1+ci3	fast, scal
git-man	1:2.39.2-1.1+ci3	fast, scal
gnupg	2.2.40-1.1.astra6+b2	GNU priva
gnupg-l10n	2.2.40-1.1.astra6+b2	GNU priva
gnupg-utils	2.2.40-1.1.astra6+b2	GNU priva
golang-github-containers-common	0.57.4+ds1-2+b1	Common
golang-github-containers-image	5.29.3-1	Configura
gpg	2.2.40-1.1.astra6+b2	GNU Priva
gpg-agent	2.2.40-1.1.astra6+b2	GNU priva
gpg-wks-client	2.2.40-1.1.astra6+b2	GNU priva
gpg-wks-server	2.2.40-1.1.astra6+b2	GNU priva
gpgconf	2.2.40-1.1.astra6+b2	GNU priva
gpgsm	2.2.40-1.1.astra6+b2	GNU priva
gpgv	2.2.40-1.1.astra6+b2	GNU priva
grep	3.8-5+b1	GNU grep
gzip	1.12-1+b1	GNU com
hostname	3.23+nmu1+b1	utility to
ieee-data	20220827.1	OUI and I
init-system-helpers	1.65.2+b1	helper to
iptables	1.8.9-2+b1	administr

Таблица 19 – продол

Пакет	Версия	Описани
less	590-2.1~deb12u2	pager pro
libacl1:amd64	2.3.1-3+b1	access co
libapparmor1:amd64	3.0.8-3+b1	changeha
libapt-pkg6.0:amd64	2.6.1+ci202312061551+astra2+b1	package
libargon2-1:amd64	0~20171227-0.3+deb12u1+b1	memory-
libasan8:amd64	12.2.0-14.astra3+b1	AddressS
libassuan0:amd64	2.5.5-5+b1	IPC librar
libatomic1:amd64	12.2.0-14.astra3+b1	support li
libattr1:amd64	1:2.5.1-4+ci202403112058+astra3+b1	extended
libaudit-common	1:3.0.9-1+ci202401241435+astra3+b2	Dynamic
libaudit1:amd64	1:3.0.9-1+ci202401241435+astra3+b2	Dynamic
libbinutils:amd64	2.40-2+b1	GNU bina
libblkid1:amd64	2.38.1-5+deb12u1+astra13+b1	block dev
libbrotli1:amd64	1.0.9-2+b1	library im
libbsd0:amd64	0.11.7-2+b1	utility fun
libbz2-1.0:amd64	1.0.8-5+b1	high-qual
libc-bin	2.36-9+deb12u7+ci202405171200+astra5+b1	GNU C Lib
libc-dev-bin	2.36-9+deb12u7+ci202405171200+astra5+b1	GNU C Lib
libc6:amd64	2.36-9+deb12u7+ci202405171200+astra5+b1	GNU C Lib
libc6-dev:amd64	2.36-9+deb12u7+ci202405171200+astra5+b1	GNU C Lib
libcap-ng0:amd64	0.8.3-1+b2	alternate
libcap2:amd64	1:2.66-4+b1	POSIX 10
libcbor0.8:amd64	0.8.0-2+b1	library fo
libcc1-0:amd64	12.2.0-14.astra3+b1	GCC cc1
libcom-err2:amd64	1.47.0-2+b1	common
libcrypt-dev:amd64	1:4.4.33-2+b1	libcrypt d
libcrypt1:amd64	1:4.4.33-2+b1	libcrypt s
libcryptsetup12:amd64	2:2.6.1-4~deb12u2+b2	disk encr
libctf-nobfd0:amd64	2.40-2+b1	Compact
libctf0:amd64	2.40-2+b1	Compact
libcurl3-gnutls:amd64	7.88.1-10+deb12u5+b2	easy-to-u
libcurl4:amd64	7.88.1-10+deb12u5+b2	easy-to-u
libdb5.3:amd64	5.3.28+dfsg2-1+ci202311141605+astra1+b1	Berkeley
libdbus-1-3:amd64	1.14.8-2~deb12u1+ci12+b2	simple in
libdebconfclient0:amd64	0.270astra1+b1	Debian C
libdevmapper1.02.1:amd64	2:1.02.185-2+b1	Linux Ker
libedit2:amd64	3.1-20221030-2+b1	BSD editl
liberror-perl	0.17029-2+b1	Perl modu
libexpat1:amd64	2.5.0-1+ci202405221303+astra2	XML pars
libexpat1-dev:amd64	2.5.0-1+ci202405221303+astra2	XML pars
libext2fs2:amd64	1.47.0-2+b1	ext2/ext3
libfdisk1:amd64	2.38.1-5+deb12u1+astra13+b1	fdisk part
libffi8:amd64	3.4.4-1+b1	Foreign F
libfido2-1:amd64	1.12.0-2+b2	library fo
libfile-find-rule-perl	0.34-3+b1	module to
libflygetexe	1.0.14+ci2+b1	libflygete
libfuse3-3:amd64	3.14.0-4+b1	Filesyster
libgcc-12-dev:amd64	12.2.0-14.astra3+b1	GCC supp
libgcc-s1:amd64	12.2.0-14.astra3+b1	GCC supp
libgcrypt20:amd64	1.10.1-3.astra2+b1	LGPL Cry
libgdbm-compat4:amd64	1.23-3+b1	GNU dbm
libgdbm6:amd64	1.23-3+b1	GNU dbm
libglib2.0-0:amd64	2.74.6-2+deb12u2+b1	GLib libra
libgmp10:amd64	2:6.2.1+dfsg1-1.1+b1	Multiprec
libgnutls30:amd64	3.7.9-2+deb12u2+b1	GNU TLS
libgomp1:amd64	12.2.0-14.astra3+b1	GCC Ope
libgost	2.0.2-5+ci4+b1	This pack

Таблица 19 – продол

Пакет	Версия	Описани
libgpg-error0:amd64	1.46-1+b1	GnuPG de
libgpgme11:amd64	1.18.0-3+b1	GPGME -
libgprofng0:amd64	2.40-2+b1	GNU Next
libgssapi-krb5-2:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerbe
libhogweed6:amd64	3.8.1-2+b1	low level
libicu72:amd64	72.1-3+b1	Internatio
libidn2-0:amd64	2.3.3-1+b1	Internatio
libip4tc2:amd64	1.8.9-2+b1	netfilter l
libip6tc2:amd64	1.8.9-2+b1	netfilter l
libisl23:amd64	0.25-1.1+b1	manipula
libitm1:amd64	12.2.0-14.astra3+b1	GNU Tran
libjansson4:amd64	2.14-2+b1	C library
libjs-jquery	3.6.1+dfsg+~3.5.14-1+b1	JavaScrip
libjs-sphinxdoc	5.3.0-4+b1	JavaScrip
libjs-underscore	1.13.4~dfsg+~1.11.4-3	JavaScrip
libjson-c5:amd64	0.16-2+b1	JSON ma
libk5crypto3:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerbe
libkeyutils1:amd64	1.6.3-2+b1	Linux Key
libkmod2:amd64	30+20230519-1ubuntu3astra1+b1	libkmod s
libkrb5-3:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerbe
libkrb5support0:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerbe
libksba8:amd64	1.6.3-2+b1	X.509 an
libldap-2.5-0:amd64	2.5.13+dfsg-5+ci202405061914+astra2	OpenLDA
liblocale-gettext-perl	1.07-5+b1	module u
liblsan0:amd64	12.2.0-14.astra3+b1	LeakSani
liblua5.3-0:amd64	5.3.6-2+b1	Shared lib
liblz4-1:amd64	1.9.4-1+b1	Fast LZ co
liblzma5:amd64	5.4.1-0.2+b1	XZ-forma
libmd0:amd64	1.0.4-2+b1	message
libmnl0:amd64	1.0.4-3+b1	minimalis
libmount1:amd64	2.38.1-5+deb12u1+astra13+b1	device m
libmpc3:amd64	1.3.1-1+b1	multiple p
libmpfr6:amd64	4.2.0-1+b1	multiple p
libncurses-dev:amd64	6.4-4+b1	develope
libncurses5-dev:amd64	6.4-4+b1	transition
libncurses6:amd64	6.4-4+b1	shared lib
libncursesw6:amd64	6.4-4+b1	shared lib
libnetfilter-conntrack3:amd64	1.0.9-3+b1	Netfilter n
libnettle8:amd64	3.8.1-2+b1	low level
libnfnetlink0:amd64	1.0.2-2+b1	Netfilter n
libnftnl11:amd64	1.2.4-2+b1	Netfilter n
libnghttp2-14:amd64	1.52.0-1+deb12u1+ci1+b1	library im
libnl-3-200:amd64	3.7.0-0.2+b2	library fo
libnpth0:amd64	1.6-3+b1	replacem
libnsl-dev:amd64	1.3.0-2+b1	libnsl dev
libnsl2:amd64	1.3.0-2+b1	Public cli
libnuma1:amd64	2.0.16-1+b1	Libraries
libnumber-compare-perl	0.03-3+b1	module fo
libopendbx1	1.4.6-16+b1	Lightweig
libopenscap25	1.3.7+dfsg-1+deb12u1+b1	libraries e
libp11-kit0:amd64	0.24.1-2+b1	library fo
libpam-modules:amd64	1.5.2-6+deb12u1.astra.se21+b1	Pluggable
libpam-modules-bin	1.5.2-6+deb12u1.astra.se21+b1	Pluggable
libpam-runtime	1.5.2-6+deb12u1.astra.se21+b1	Runtime s
libpam-systemd:amd64	252.17-1~deb12u1astra.se1+ci13+b1	system a
libpam0g:amd64	1.5.2-6+deb12u1.astra.se21+b1	Pluggable
libparsec-aud3	3.9+ci132	audit libr

Таблица 19 – продол

Пакет	Версия	Описани
libparsec-base3	3.9+ci132	base libra
libparsec-cap3	3.9+ci132	capabiliti
libparsec-mic3	3.9+ci132	mandate
libpcre2-8-0:amd64	10.42-1+b1	New Perl
libpcre3:amd64	2:8.39-15+b1	Old Perl 5
libpdp	3.9+ci132	library fo
libperl5.36:amd64	5.36.0-7+deb12u1+ci202404171352+astra2	shared Pe
libpopt0:amd64	1.19+dfsg-1+b1	lib for pa
libproc2-0:amd64	2:4.0.2-3+b1	library fo
libprotobuf32:amd64	3.21.12-3+b1	protocol b
libpsl5:amd64	0.21.2-1+b1	Library fo
libpython3-dev:amd64	3.11.2-1+b1	header fil
libpython3-stdlib:amd64	3.11.2-1+b1	interactiv
libpython3.11:amd64	3.11.2-6astra4+b1	Shared Py
libpython3.11-dev:amd64	3.11.2-6astra4+b1	Header fi
libpython3.11-minimal:amd64	3.11.2-6astra4+b1	Minimal s
libpython3.11-stdlib:amd64	3.11.2-6astra4+b1	Interactiv
libqrencode4:amd64	4.1.1-1+b1	QR Code
libquadmath0:amd64	12.2.0-14.astra3+b1	GCC Qua
libreadline8:amd64	8.2-1.3+b1	GNU read
librpm9	4.18.0+dfsg-1+deb12u1+b2	RPM shar
librpmio9	4.18.0+dfsg-1+deb12u1+b2	RPM IO sh
librtmp1:amd64	2.4+20151223.gitfa8646d.1-2+b1	toolkit fo
libsasl2-2:amd64	2.1.28+dfsg-10+b2	Cyrus SA
libsasl2-modules-db:amd64	2.1.28+dfsg-10+b2	Cyrus SA
libseccomp2:amd64	2.5.4-1+b2	high leve
libselinux1:amd64	3.4-1+b1	SELinux r
libsemanage-common	3.4-1+b1	Common
libsemanage2:amd64	3.4-1+b1	SELinux p
libsepol2:amd64	3.4-2.1+b1	SELinux l
libsigsegv2:amd64	2.14-1+b1	Library fo
libslirp0:amd64	4.7.0-1+b1	General p
libsmartcols1:amd64	2.38.1-5+deb12u1+astra13+b1	smart col
libsodium23:amd64	1.0.18-1+b1	Network c
libsqlite3-0:amd64	3.40.1-2+ci202405281356+astra2	SQLite 3 s
libss2:amd64	1.47.0-2+b1	command
libssh-4:amd64	0.10.6-0+deb12u1+b2	tiny C SSH
libssh2-1:amd64	1.10.0-3+b2	SSH2 clie
libssl3:amd64	3.2.0-2-astra5+ci1	Secure So
libstdc++6:amd64	12.2.0-14.astra3+b1	GNU Stan
libsubid4:amd64	1:4.13+dfsg1-1.astra.se10+b1	subordina
libsystemd-shared:amd64	252.17-1~deb12u1astra.se1+ci13+b1	systemd
libsystemd0:amd64	252.17-1~deb12u1astra.se1+ci13+b1	systemd
libtasn1-6:amd64	4.19.0-2+b1	Manage A
libtcl8.6:amd64	8.6.13+dfsg-2+b1	Tcl (the To
libtext-charwidth-perl:amd64	0.04-11+b1	get displa
libtext-glob-perl	0.11-3+b1	Perl modu
libtext-iconv-perl:amd64	1.7-8+b1	module to
libtext-wrapi18n-perl	0.06-10+b1	internatio
libtinfo6:amd64	6.4-4+b1	shared lo
libtirpc-common	1.3.3+ds-1+b1	transport
libtirpc-dev:amd64	1.3.3+ds-1+b1	transport
libtirpc3:amd64	1.3.3+ds-1+b1	transport
libtsan2:amd64	12.2.0-14.astra3+b1	ThreadSa
libubsan1:amd64	12.2.0-14.astra3+b1	UBSan – u
libudev1:amd64	252.17-1~deb12u1astra.se1+ci13+b1	libudev sh
libunistring2:amd64	1.0-2+b1	Unicode s

Таблица 19 – продол

Пакет	Версия	Описани
libuuid1:amd64	2.38.1-5+deb12u1+astra13+b1	Universal
libvirt-clients	10.5.0-1.astra.se06	Programs
libvirt0:amd64	10.5.0-1.astra.se06	library fo
libxml2:amd64	2.9.14+dfsg-1.3~deb12u1+ci202405141738+astra2	GNOME X
libxmlsec1:amd64	1.2.37-2+b2	XML secu
libxmlsec1-openssl:amd64	1.2.37-2+b2	Openssl e
libxslt1.1:amd64	1.1.35-1+b1	XSLT 1.0
libxtables12:amd64	1.8.9-2+b1	netfilter x
libxxhash0:amd64	0.8.1-1+b1	shared lib
libyajl2:amd64	2.1.0-3+deb12u2+b1	Yet Anoth
libyaml-0-2:amd64	0.2.5-1+b1	Fast YAM
libzstd1:amd64	1.5.4+dfsg2-5+b1	fast lossl
linux-libc-dev:amd64	5.4.0-54astra7+ci98	latest lin
linux-libc-dev-6.1.90-1:amd64	6.1.90-1.astra2+ci15	Linux Ker
login	1:4.13+dfsg1-1.astra.se10+b1	system lo
logsave	1.47.0-2+b1	save the
lsb-base	11.6	transition
lsb-release	12.0-1+b1	Linux Sta
mawk	1.3.4.20200120-3.1+b1	Pattern s
media-types	10.0.0	List of sta
molecule	24.9.0+aa1.1.2	Molecule
molecule-libvirt	0.2.0+aa1.1.1	libvirt Mo
molecule-molecule-brest	1.1.0+aa1.1.0	Brest Mol
molecule-molecule-yandexcloud	0.3.1+aa1.1.0	YandexCl
molecule-plugins	23.5.3+aa1.1.0	Molecule
mount	2.38.1-5+deb12u1+astra13+b1	tools for
ncurses-base	6.4-4+b1	basic ter
ncurses-bin	6.4-4+b1	terminal-
netbase	6.4	Basic TCP
openscap-common	1.3.7+dfsg-1+deb12u1+b1	OpenSCA
openscap-scanner	1.3.7+dfsg-1+deb12u1+b1	OpenScap
openssh-client	1:9.6p1-2~deb10u1astra8se5	secure sh
openssl	3.2.0-2-astra5+ci1	Secure So
oval-db	0.0.1+ci3	Install ov
passwd	1:4.13+dfsg1-1.astra.se10+b1	change a
perl	5.36.0-7+deb12u1+ci202404171352+astra2	Larry Wal
perl-base	5.36.0-7+deb12u1+ci202404171352+astra2	minimal P
perl-modules-5.36	5.36.0-7+deb12u1+ci202404171352+astra2	Core Perl
pinentry-curses	1.2.1-1+b1	curses-ba
podman	4.7.2.astra1	engine to
procps	2:4.0.2-3+b1	/proc file
pytest-ansible	24.9.0+aa1.1.0	Plugin for
python3	3.11.2-1+b1	interactiv
python3-attr	22.2.0-1	Attributes
python3-bcrypt	3.2.2-1+b1	password
python3-blinker	1.5-1	Fast, simp
python3-certifi	2022.9.24-1	root certi
python3-cffi	1.15.1-5+b1	Foreign F
python3-cffi-backend:amd64	1.15.1-5+b1	Foreign F
python3-chardet	5.1.0+dfsg-2	Universal
python3-charset-normalizer	3.0.1-2	charset, e
python3-click	8.1.3-2	Wrapper
python3-colorama	0.4.6-2	Cross-pla
python3-cryptography	38.0.4-3+ci202405171208+astra3+b1	Python lib
python3-daemon	2.3.2-1	library fo
python3-decorator	5.1.1-3	simplify u
python3-dev	3.11.2-1+b1	header fil

Таблица 19 – продол

Пакет	Версия	Описани
python3-distlib	0.3.6-1	low-level
python3-distro	1.8.0-1	Linux OS
python3-distutils	3.11.2-3+b1	distutils p
python3-dnspython	2.3.0-1+b1	DNS toolk
python3-docker	5.0.3-1	Python 3
python3-dogpile.cache	1.1.8-2	caching f
python3-execnet	1.9.0-1	rapid mul
python3-filelock	3.9.0-1	platform
python3-httplib2	0.20.4-3	comprehe
python3-idna	3.3-1	Python ID
python3-importlib-metadata	4.12.0-1	library to
python3-iniconfig	1.1.1-2	brain-dea
python3-iso8601	1.0.2-1	Python m
python3-jinja2	3.1.2-1	small but
python3-jmespath	1.0.1-1	JSON Mat
python3-json-pointer	2.3-2	resolve JS
python3-jsonpatch	1.32-2	library to
python3-jsonschema	4.10.3-1	An(other)
python3-jwt	2.6.0-1+b1	Python 3
python3-keystoneauth1	5.0.0-2	authentic
python3-lib2to3	3.11.2-3+b1	Interactiv
python3-libvirt	10.5.0-1astra.se01	libvirt Pyt
python3-lockfile	1:0.12.2-2.2	file lockin
python3-lxml:amd64	4.9.2-1+b1	pythonic
python3-mako	1.2.4+ds-1	fast and l
python3-markdown-it	2.1.0-5	Python po
python3-markupsafe	2.1.2-1+b1	HTML/XHT
python3-mdurl	0.1.2-1	Python po
python3-minimal	3.11.2-1+b1	minimal s
python3-more-itertools	8.10.0-2	library wi
python3-mypy-extensions	0.4.3-4	Experime
python3-nacl	1.5.0-2+b1	Python bi
python3-netaddr	0.8.0-2	manipula
python3-netifaces:amd64	0.11.0-2+b1	portable
python3-ntlm-auth	1.4.0-2	NTLM low
python3-oauthlib	3.2.2-1+b1	generic, s
python3-openssl	23.0.0-1+b1	Python 3
python3-os-service-types	1.7.0-2	lib for co
python3-packaging	23.0-1	core utilit
python3-paramiko	2.12.0-2+b1	Make ssh
python3-pathspec	0.11.0-1	utility libr
python3-pbr	5.10.0-2+b1	inject use
python3-pexpect	4.8.0-4	Python 3
python3-pip	23.0.1+dfsg-1	Python pa
python3-pkg-resources	66.1.1-1	Package
python3-platformdirs	2.6.0-1	determin
python3-pluggy	1.0.0+repack-1	plugin an
python3-ply	3.11-5+b1	Lex and Y
python3-protobuf	3.21.12-3+b1	Python 3
python3-ptyprocess	0.7.0-5	Run a sub
python3-py	1.11.0-1	Advanced
python3-pycparser	2.21-1	C parser
python3-pygments	2.14.0+dfsg-1	syntax hi
python3-pyparsing	3.0.9-1	alternativ
python3-pyrsistent:amd64	0.18.1-1+b1	persisten
python3-pytest	7.2.1-2	Simple, p
python3-pytest-xdist	3.1.0-1	xdist plug

Таблица 19 – продол

Пакет	Версия	Описани
python3-requests	2.28.1+dfsg-1+b1	elegant a
python3-requests-ntlm	1.1.0-3+b1	Adds sup
python3-requestsexceptions	1.4.0-3+b1	import ex
python3-resolvelib	0.9.0-2	module t
python3-rich	13.3.1-1	render ric
python3-ruamel.yaml.clib:amd64	0.2.7-1+b1	C version
python3-selinux	3.4-1+b1	Python3
python3-setuptools	66.1.1-1	Python3
python3-six	1.16.0-4	Python 2
python3-stevedore	4.0.2-2	manage c
python3-tblib	1.7.0-3	Python 3
python3-urllib3	1.26.12-1+ci202406111901+astra2	HTTP libr
python3-websocket	1.2.3-1	WebSock
python3-wheel	0.38.4-2	built-pack
python3-winrm	0.3.0-4+deb12u1	Python 3
python3-xmltodict	0.13.0-1	Makes wo
python3-yaml	6.0-3+b1	YAML par
python3-zipp	1.0.0-6	pathlib-co
python3.11	3.11.2-6astra4+b1	Interactiv
python3.11-aenum	3.1.15+aa1.1.0	Advanced
python3.11-bindep	2.11.0+aa1.2.0	Binary de
python3.11-black	24.10.0+aa1.1.0	The unco
python3.11-bracex	2.5.post1+aa1.1.0	Bash styl
python3.11-cachetools	5.5.0+aa1.1.0	Extensibl
python3.11-chardet	5.2.0+aa1.1.0	Universal
python3.11-click-help-colors	0.9.4+aa1.1.0	Colorizati
python3.11-cryptography	42.0.8+aa1.1.0	cryptogra
python3.11-dev	3.11.2-6astra4+b1	Header fi
python3.11-dicttoxml	1.7.16+aa1.1.0	Converts
python3.11-distlib	0.3.9+aa1.1.0	Distributi
python3.11-enrich	1.2.7+aa1.1.0	enrich
python3.11-filelock	3.16.1+aa1.1.0	A platform
python3.11-googleapis-common-protos	1.65.0+aa1.1.0	Common
python3.11-grpcio	1.67.1+aa1.1.0	HTTP/2-b
python3.11-minimal	3.11.2-6astra4+b1	Minimal s
python3.11-onigurumacffi	1.3.0+aa1.1.0	python cf
python3.11-openstacksdk	4.1.0+aa1.1.0	An SDK fo
python3.11-packaging	24.2+aa1.1.0	Core utili
python3.11-parsley	1.3+aa1.1.0	Parsing a
python3.11-platformdirs	4.3.6+aa1.1.0	A small P
python3.11-pluggy	1.5.0+aa1.1.0	plugin an
python3.11-protobuf	4.25.5+aa1.1.0	Protocol B
python3.11-pyjwt	2.9.0+aa1.1.0	JSON Web
python3.11-pyone	6.6.3+aa1.1.0	Python Bi
python3.11-pyproject-api	1.8.0+aa1.1.0	API to int
python3.11-pypsrp	0.8.1+aa1.1.0	PowerShe
python3.11-pyspnego	0.11.0+aa1.1.0	Windows
python3.11-python-gnupg	0.5.2+aa1.1.0	A wrappe
python3.11-python-vagrant	1.0.0+aa1.1.0	Python bi
python3.11-requests	2.32.3+aa1.1.0	Python H
python3.11-ruamel.yaml	0.18.6+aa1.1.0	ruamel.ya
python3.11-subprocess-tee	0.4.2+aa1.1.0	subproce
python3.11-tox	4.23.2+aa1.1.0	tox is a g
python3.11-urllib3	2.2.3+aa1.1.0	HTTP libr
python3.11-virtualenv	20.27.1+aa1.1.0	Virtual Py
python3.11-wcmatch	10.0+aa1.1.0	Wildcard/
python3.11-yamllint	1.35.1+aa1.1.0	A linter fo

Таблица 19 – продол

Пакет	Версия	Описани
python3.11-yandexcloud	0.324.0+aa1.1.0	The Yand
readline-common	8.2-1.3+b1	GNU read
rpcsvc-proto	1.4.3-1+b1	RPC proto
runc	1.1.12~astra1+ci3+b1	Open Cor
sed	4.9-1+b1	GNU strea
sensible-utils	0.0.17+nmu1	Utilities fo
slirp4netns	1.2.0-1+b1	User-mod
sshpass	1.09-1+b1	Non-inter
systemd	252.17-1~deb12u1astra.se1+ci13+b1	system a
systemd-sysv	252.17-1~deb12u1astra.se1+ci13+b1	system a
sysvinit-utils	3.06-4+b1	System-V
tar	1.34+dfsg-1.2+deb12u1+b1	GNU vers
tcl-expect:amd64	5.45.4-2+ci202403211247+astra1+b1	Automate
tcl8.6	8.6.13+dfsg-2+b1	Tcl (the To
tox-ansible	24.9.0+aa1.1.0	A radical
tzdata	2024a-4	time zone
uidmap	1:4.13+dfsg1-1.astra.se10+b1	programs
usrmerge	37~deb12u1+b1	Convert t
util-linux	2.38.1-5+deb12u1+astra13+b1	miscellan
util-linux-extra	2.38.1-5+deb12u1+astra13+b1	interactiv
uuid-runtime	2.38.1-5+deb12u1+astra13+b1	runtime c
zlib1g:amd64	1:1.2.13.dfsg-1+b1	compress
zlib1g-dev:amd64	1:1.2.13.dfsg-1+b1	compress

Пакеты Python

Пакет
aenum
ansible-builder @ file:///docker_share/dist/ansible_builder-3.1.0-py3-none-any.whl
ansible-compat
ansible-core
ansible-creator @ file:///docker_share/dist/ansible_creator-24.10.1-py3-none-any.whl
ansible-lint @ file:///docker_share/dist/ansible_lint-24.9.2-py3-none-any.whl
ansible-navigator @ file:///docker_share/dist/ansible_navigator-24.9.0-py3-none-any.whl
ansible-pylibssh
ansible-runner @ file:///docker_share/dist/ansible_runner-2.4.0-py3-none-any.whl
ansible-sign @ file:///docker_share/dist/ansible_sign-0.1.1-py3-none-any.whl
attrs
bcrypt
bindep
black
blinker
bracex
cachetools
certifi
cffi
cfgv
chardet
charset-normalizer
click
click-help-colors
colorama
cryptography
Cython
decorator

продолжается на

Таблица 20 – продолжение с предыдущей страницы

Пакет
dicttoxml
distlib
distro
dnspython
docker
dogpile.cache
dumb-init
enrich
execnet
filelock
googleapis-common-protos
grpcio
httplib2
identify
idna
importlib-metadata
iniconfig
iso8601
Jinja2
jmespath
jsonpatch
jsonpointer
jsonschema
jxmlease
keystoneauth1
libvirt-python
lockfile
lxml
Mako
markdown-it-py
MarkupSafe
mdurl
molecule @ file:///docker_share/dist/molecule-24.9.0-py3-none-any.whl
molecule-libvirt @ file:///docker_share/dist/molecule_libvirt-0.2.0-py3-none-any.whl
molecule_molecule_brest @ file:///docker_share/dist/molecule_molecule_brest-1.1.0-py3-none-any.whl
molecule_molecule_yandexcloud @ file:///docker_share/dist/molecule_molecule_yandexcloud-0.3.1-py3-no
molecule-plugins @ file:///docker_share/dist/molecule_plugins-23.5.3-py3-none-any.whl
more-itertools
mypy-extensions
ncclient
netaddr
netifaces
nodeenv
ntlm-auth
oauthlib
onigurumacffi
openstacksdk
os-service-types
packaging
paramiko
Parsley
pathspec
pbr
pexpect
platformdirs
pluggy
ply

продолжается на

Таблица 20 – продолжение с предыдущей страницы

Пакет
pre_commit
protobuf
ptyprocess
py
pycparser
Pygments
PyJWT
PyNaCl
pyone
pyOpenSSL
pyparsing
pyproject-api
pypsrp @ file:///docker_share/dist/pypsrp-0.8.1-py3-none-any.whl
pyrsistent
pyspnego
pytest
pytest-ansible @ file:///docker_share/dist/pytest_ansible-24.9.0-py3-none-any.whl
pytest-xdist
python-daemon
python-gnupg
python-vagrant
pywinrm
PyYAML
requests
requestsexceptions
requests-ntlm
resolvelib
rich
ruamel.yaml
ruamel.yaml.clib
selinux
six
stevedore
subprocess-tee
tblib
textfsm
tox
tox-ansible @ file:///docker_share/dist/tox_ansible-24.9.0-py3-none-any.whl
ttp
urllib3
virtualenv
wcmatch
websocket-client
xmltodict
yamllint
yandexcloud
zipp

Коллекции Ansible

Коллекция	Версия
ansible.netcommon	7.1.0
ansible.posix	2.0.0
ansible.utils	5.1.2
astra.brest	4.0.3
astra.yandexcloud	0.4.0
community.crypto	2.24.0
community.general	10.3.0
community.libvirt	1.3.1

0.1.1

Пакеты APT

Пакет	Версия	Описани
adduser	3.134+astra.se10	add and r
ansible-builder	3.1.0+aa1.1.0	An Ansibl
ansible-compat	24.9.1+aa1.1.0	Ansible c
ansible-core	2.15.10-1	Configura
ansible-creator	24.10.1+aa2.0.1	A CLI tool
ansible-lint	24.9.2+aa1.1.0	Checks p
ansible-navigator	24.9.0+aa1.1.0	A text-ba
ansible-runner	2.4.0+aa1.1.0	Consister
ansible-sign	0.1.1+aa1.1.0	Ansible c
apt	2.6.1+ci202312061551+astra2+b1	command
apt-transport-https	2.6.1+ci202312061551+astra2+b1	transition
astra-archive-keyring	2023.12	GnuPG ar
astra-version	8.1.41+v1.8.1.12	Update A
base-files	12.4+deb12u1.astra6	Debian ba
base-passwd	3.6.1+b1	Debian ba
bash	5.2.15-2.astra.se1+b2	GNU Bou
bc	1.07.1-3+b1	GNU bc a
binutils	2.40-2+b1	GNU asse
binutils-common:amd64	2.40-2+b1	Common
binutils-x86-64-linux-gnu	2.40-2+b1	GNU bina
bsdutils	1:2.38.1-5+deb12u1+astra13+b1	basic utili
ca-certificates	20230311+b2	Common
conmon	2.1.6+ds1-1+b1	OCI conta
containernetworking-plugins	1.1.1+ds1-3+b1	standard
containers-storage	1.51.0+ds1-2+b1	CLI tools
coreutils	9.1-1+ci202402121122+astra2+b1	GNU core
cpp	4:12.2.0-3+b1	GNU C pr
cpp-12	12.2.0-14.astra3+b1	GNU C pr
crun	1.8.1-1+deb12u1+b1	lightweig
cryptsetup	2:2.6.1-4~deb12u2+b2	disk encr
cryptsetup-bin	2:2.6.1-4~deb12u2+b2	disk encr
curl	7.88.1-10+deb12u5+b2	command
dash	0.5.12-2.astra1	POSIX-co
dbus	1.14.8-2~deb12u1+ci12+b2	simple in
dbus-bin	1.14.8-2~deb12u1+ci12+b2	simple in
dbus-daemon	1.14.8-2~deb12u1+ci12+b2	simple in
dbus-session-bus-common	1.14.8-2~deb12u1+ci12+b2	simple in
dbus-system-bus-common	1.14.8-2~deb12u1+ci12+b2	simple in
dbus-user-session	1.14.8-2~deb12u1+ci12+b2	simple in
debconf	1.5.82+b1	Debian co

Таблица 21 – продол

Пакет	Версия	Описани
debianutils	5.7-0.5~deb12u1+b1	Miscellan
diffutils	1:3.8-4+b1	File comp
dirmngr	2.2.40-1.1.astra6+b2	GNU priva
dmsetup	2:1.02.185-2+b1	Linux Ker
dpkg	1.21.22.astra.se3+b2	Debian pa
e2fsprogs	1.47.0-2+b1	ext2/ext3
expect	5.45.4-2+ci202403211247+astra1+b1	Automate
findutils	4.9.0-4+b1	utilities fo
fuse-overlayfs	1.10-1+b1	implemen
fuse3	3.14.0-4+b1	Filesyster
gawk	1:5.2.1-2+b1	GNU awk
gcc	4:12.2.0-3+b1	GNU C co
gcc-12	12.2.0-14.astra3+b1	GNU C co
gcc-12-base:amd64	12.2.0-14.astra3+b1	GCC, the
git	1:2.39.2-1.1+ci3	fast, scal
git-man	1:2.39.2-1.1+ci3	fast, scal
gnupg	2.2.40-1.1.astra6+b2	GNU priva
gnupg-l10n	2.2.40-1.1.astra6+b2	GNU priva
gnupg-utils	2.2.40-1.1.astra6+b2	GNU priva
golang-github-containers-common	0.57.4+ds1-2+b1	Common
golang-github-containers-image	5.29.3-1	Configura
gpg	2.2.40-1.1.astra6+b2	GNU Priva
gpg-agent	2.2.40-1.1.astra6+b2	GNU priva
gpg-wks-client	2.2.40-1.1.astra6+b2	GNU priva
gpg-wks-server	2.2.40-1.1.astra6+b2	GNU priva
gpgconf	2.2.40-1.1.astra6+b2	GNU priva
gpgsm	2.2.40-1.1.astra6+b2	GNU priva
gpgv	2.2.40-1.1.astra6+b2	GNU priva
grep	3.8-5+b1	GNU grep
gzip	1.12-1+b1	GNU com
hostname	3.23+nmu1+b1	utility to
ieee-data	20220827.1	OUI and I
init-system-helpers	1.65.2+b1	helper to
iptables	1.8.9-2+b1	administr
libacl1:amd64	2.3.1-3+b1	access co
libapparmor1:amd64	3.0.8-3+b1	changeha
libapt-pkg6.0:amd64	2.6.1+ci202312061551+astra2+b1	package
libargon2-1:amd64	0~20171227-0.3+deb12u1+b1	memory-
libasan8:amd64	12.2.0-14.astra3+b1	AddressS
libassuan0:amd64	2.5.5-5+b1	IPC librar
libatomic1:amd64	12.2.0-14.astra3+b1	support li
libattr1:amd64	1:2.5.1-4+ci202403112058+astra3+b1	extended
libaudit-common	1:3.0.9-1+ci202401241435+astra3+b2	Dynamic
libaudit1:amd64	1:3.0.9-1+ci202401241435+astra3+b2	Dynamic
libbinutils:amd64	2.40-2+b1	GNU bina
libblkid1:amd64	2.38.1-5+deb12u1+astra13+b1	block dev
libbrotli1:amd64	1.0.9-2+b1	library im
libbsd0:amd64	0.11.7-2+b1	utility fun
libbz2-1.0:amd64	1.0.8-5+b1	high-qual
libc-bin	2.36-9+deb12u7+ci202405171200+astra5+b1	GNU C Li
libc-dev-bin	2.36-9+deb12u7+ci202405171200+astra5+b1	GNU C Li
libc6:amd64	2.36-9+deb12u7+ci202405171200+astra5+b1	GNU C Li
libc6-dev:amd64	2.36-9+deb12u7+ci202405171200+astra5+b1	GNU C Li
libcap-ng0:amd64	0.8.3-1+b2	alternate
libcap2:amd64	1:2.66-4+b1	POSIX 10
libcbor0.8:amd64	0.8.0-2+b1	library fo
libcc1-0:amd64	12.2.0-14.astra3+b1	GCC cc1

Таблица 21 – продол

Пакет	Версия	Описани
libcom-err2:amd64	1.47.0-2+b1	common
libcrypt-dev:amd64	1:4.4.33-2+b1	libcrypt d
libcrypt1:amd64	1:4.4.33-2+b1	libcrypt s
libcryptsetup12:amd64	2:2.6.1-4~deb12u2+b2	disk encr
libctf-nobfd0:amd64	2.40-2+b1	Compact
libctf0:amd64	2.40-2+b1	Compact
libcurl3-gnutls:amd64	7.88.1-10+deb12u5+b2	easy-to-u
libcurl4:amd64	7.88.1-10+deb12u5+b2	easy-to-u
libdb5.3:amd64	5.3.28+dfsg2-1+ci202311141605+astra1+b1	Berkeley
libdbus-1-3:amd64	1.14.8-2~deb12u1+ci12+b2	simple in
libdebconfclient0:amd64	0.270astra1+b1	Debian C
libdevmapper1.02.1:amd64	2:1.02.185-2+b1	Linux Ker
libedit2:amd64	3.1-20221030-2+b1	BSD editl
liberror-perl	0.17029-2+b1	Perl modu
libexpat1:amd64	2.5.0-1+ci202405221303+astra2	XML pars
libexpat1-dev:amd64	2.5.0-1+ci202405221303+astra2	XML pars
libext2fs2:amd64	1.47.0-2+b1	ext2/ext3
libfdisk1:amd64	2.38.1-5+deb12u1+astra13+b1	fdisk part
libffi8:amd64	3.4.4-1+b1	Foreign F
libfido2-1:amd64	1.12.0-2+b2	library fo
libfile-find-rule-perl	0.34-3+b1	module to
libflygetexe	1.0.14+ci2+b1	libflygete
libfuse3-3:amd64	3.14.0-4+b1	Filesyster
libgcc-12-dev:amd64	12.2.0-14.astra3+b1	GCC supp
libgcc-s1:amd64	12.2.0-14.astra3+b1	GCC supp
libgcrypt20:amd64	1.10.1-3.astra2+b1	LGPL Cry
libgdbm-compat4:amd64	1.23-3+b1	GNU dbm
libgdbm6:amd64	1.23-3+b1	GNU dbm
libglib2.0-0:amd64	2.74.6-2+deb12u2+b1	GLib libra
libgmp10:amd64	2:6.2.1+dfsg1-1.1+b1	Multiprec
libgnutls30:amd64	3.7.9-2+deb12u2+b1	GNU TLS
libgomp1:amd64	12.2.0-14.astra3+b1	GCC Ope
libgost	2.0.2-5+ci4+b1	This pack
libgpg-error0:amd64	1.46-1+b1	GnuPG de
libgpgme11:amd64	1.18.0-3+b1	GPGME -
libgprofng0:amd64	2.40-2+b1	GNU Next
libgssapi-krb5-2:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerbe
libhogweed6:amd64	3.8.1-2+b1	low level
libicu72:amd64	72.1-3+b1	Internatio
libidn2-0:amd64	2.3.3-1+b1	Internatio
libip4tc2:amd64	1.8.9-2+b1	netfilter l
libip6tc2:amd64	1.8.9-2+b1	netfilter l
libisl23:amd64	0.25-1.1+b1	manipula
libitm1:amd64	12.2.0-14.astra3+b1	GNU Tran
libjansson4:amd64	2.14-2+b1	C library
libjs-jquery	3.6.1+dfsg+~3.5.14-1+b1	JavaScrip
libjs-sphinxdoc	5.3.0-4+b1	JavaScrip
libjs-underscore	1.13.4~dfsg+~1.11.4-3	JavaScrip
libjson-c5:amd64	0.16-2+b1	JSON ma
libk5crypto3:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerbe
libkeyutils1:amd64	1.6.3-2+b1	Linux Key
libkmod2:amd64	30+20230519-1ubuntu3astra1+b1	libkmod s
libkrb5-3:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerbe
libkrb5support0:amd64	1.20.1-2+deb12u1astra1+b2	MIT Kerbe
libksba8:amd64	1.6.3-2+b1	X.509 an
libldap-2.5-0:amd64	2.5.13+dfsg-5+ci202405061914+astra2	OpenLDA
liblocale-gettext-perl	1.07-5+b1	module u

Таблица 21 – продол

Пакет	Версия	Описани
liblsan0:amd64	12.2.0-14.astra3+b1	LeakSani
liblua5.3-0:amd64	5.3.6-2+b1	Shared li
liblz4-1:amd64	1.9.4-1+b1	Fast LZ c
liblzma5:amd64	5.4.1-0.2+b1	XZ-forma
libmd0:amd64	1.0.4-2+b1	message
libmnl0:amd64	1.0.4-3+b1	minimalis
libmount1:amd64	2.38.1-5+deb12u1+astra13+b1	device m
libmpc3:amd64	1.3.1-1+b1	multiple
libmpfr6:amd64	4.2.0-1+b1	multiple
libncurses-dev:amd64	6.4-4+b1	develope
libncurses5-dev:amd64	6.4-4+b1	transition
libncurses6:amd64	6.4-4+b1	shared lib
libncursesw6:amd64	6.4-4+b1	shared lib
libnetfilter-conntrack3:amd64	1.0.9-3+b1	Netfilter
libnettle8:amd64	3.8.1-2+b1	low level
libnfnetlink0:amd64	1.0.2-2+b1	Netfilter
libnftnl11:amd64	1.2.4-2+b1	Netfilter
libnghttp2-14:amd64	1.52.0-1+deb12u1+ci1+b1	library im
libnpth0:amd64	1.6-3+b1	replacem
libnsl-dev:amd64	1.3.0-2+b1	libnsl dev
libnsl2:amd64	1.3.0-2+b1	Public cli
libnumber-compare-perl	0.03-3+b1	module fo
libopendbx1	1.4.6-16+b1	Lightweig
libopenscap25	1.3.7+dfsg-1+deb12u1+b1	libraries e
libp11-kit0:amd64	0.24.1-2+b1	library fo
libpam-modules:amd64	1.5.2-6+deb12u1.astra.se21+b1	Pluggable
libpam-modules-bin	1.5.2-6+deb12u1.astra.se21+b1	Pluggable
libpam-runtime	1.5.2-6+deb12u1.astra.se21+b1	Runtime s
libpam-systemd:amd64	252.17-1~deb12u1astra.se1+ci13+b1	system a
libpam0g:amd64	1.5.2-6+deb12u1.astra.se21+b1	Pluggable
libparsec-base3	3.9+ci132	base libra
libparsec-cap3	3.9+ci132	capabiliti
libpcre2-8-0:amd64	10.42-1+b1	New Perl
libpcre3:amd64	2:8.39-15+b1	Old Perl 5
libpdp	3.9+ci132	library fo
libperl5.36:amd64	5.36.0-7+deb12u1+ci202404171352+astra2	shared Pe
libpopt0:amd64	1.19+dfsg-1+b1	lib for pa
libproc2-0:amd64	2:4.0.2-3+b1	library fo
libprotobuf32:amd64	3.21.12-3+b1	protocol b
libpsl5:amd64	0.21.2-1+b1	Library fo
libpython3-dev:amd64	3.11.2-1+b1	header fil
libpython3-stdlib:amd64	3.11.2-1+b1	interactiv
libpython3.11:amd64	3.11.2-6astra4+b1	Shared Py
libpython3.11-dev:amd64	3.11.2-6astra4+b1	Header fi
libpython3.11-minimal:amd64	3.11.2-6astra4+b1	Minimal s
libpython3.11-stdlib:amd64	3.11.2-6astra4+b1	Interactiv
libqrencode4:amd64	4.1.1-1+b1	QR Code
libquadmath0:amd64	12.2.0-14.astra3+b1	GCC Qua
libreadline8:amd64	8.2-1.3+b1	GNU read
librpm9	4.18.0+dfsg-1+deb12u1+b2	RPM shar
librpmio9	4.18.0+dfsg-1+deb12u1+b2	RPM IO sh
librtmp1:amd64	2.4+20151223.gitfa8646d.1-2+b1	toolkit fo
libsasl2-2:amd64	2.1.28+dfsg-10+b2	Cyrus SA
libsasl2-modules-db:amd64	2.1.28+dfsg-10+b2	Cyrus SA
libseccomp2:amd64	2.5.4-1+b2	high leve
libselinux1:amd64	3.4-1+b1	SELinux
libsemanage-common	3.4-1+b1	Common

Таблица 21 – продол

Пакет	Версия	Описани
libsemanage2:amd64	3.4-1+b1	SELinux p
libsepol2:amd64	3.4-2.1+b1	SELinux l
libsigsegv2:amd64	2.14-1+b1	Library fo
libslirp0:amd64	4.7.0-1+b1	General p
libsmartcols1:amd64	2.38.1-5+deb12u1+astra13+b1	smart col
libsodium23:amd64	1.0.18-1+b1	Network c
libsqlite3-0:amd64	3.40.1-2+ci202405281356+astra2	SQLite 3 s
libss2:amd64	1.47.0-2+b1	command
libssh2-1:amd64	1.10.0-3+b2	SSH2 clie
libssl3:amd64	3.2.0-2-astra5+ci1	Secure So
libstdc++6:amd64	12.2.0-14.astra3+b1	GNU Stan
libsubid4:amd64	1:4.13+dfsg1-1.astra.se10+b1	subordina
libsystemd-shared:amd64	252.17-1~deb12u1astra.se1+ci13+b1	systemd
libsystemd0:amd64	252.17-1~deb12u1astra.se1+ci13+b1	systemd
libtasn1-6:amd64	4.19.0-2+b1	Manage A
libtcl8.6:amd64	8.6.13+dfsg-2+b1	Tcl (the To
libtext-charwidth-perl:amd64	0.04-11+b1	get displa
libtext-glob-perl	0.11-3+b1	Perl modu
libtext-iconv-perl:amd64	1.7-8+b1	module to
libtext-wrapi18n-perl	0.06-10+b1	internatio
libtinfo6:amd64	6.4-4+b1	shared lo
libtirpc-common	1.3.3+ds-1+b1	transport
libtirpc-dev:amd64	1.3.3+ds-1+b1	transport
libtirpc3:amd64	1.3.3+ds-1+b1	transport
libtsan2:amd64	12.2.0-14.astra3+b1	ThreadSa
libubsan1:amd64	12.2.0-14.astra3+b1	UBSan – u
libudev1:amd64	252.17-1~deb12u1astra.se1+ci13+b1	libudev sh
libunistring2:amd64	1.0-2+b1	Unicode s
libuuid1:amd64	2.38.1-5+deb12u1+astra13+b1	Universal
libxml2:amd64	2.9.14+dfsg-1.3~deb12u1+ci202405141738+astra2	GNOME X
libxmlsec1:amd64	1.2.37-2+b2	XML secu
libxmlsec1-openssl:amd64	1.2.37-2+b2	Openssl e
libxslt1.1:amd64	1.1.35-1+b1	XSLT 1.0
libxtables12:amd64	1.8.9-2+b1	netfilter x
libxxhash0:amd64	0.8.1-1+b1	shared lib
libyajl2:amd64	2.1.0-3+deb12u2+b1	Yet Anoth
libyaml-0-2:amd64	0.2.5-1+b1	Fast YAML
libzstd1:amd64	1.5.4+dfsg2-5+b1	fast lossl
linux-libc-dev:amd64	5.4.0-54astra7+ci98	latest linu
linux-libc-dev-6.1.90-1:amd64	6.1.90-1.astra2+ci15	Linux Ker
login	1:4.13+dfsg1-1.astra.se10+b1	system lo
logsave	1.47.0-2+b1	save the
lsb-base	11.6	transition
lsb-release	12.0-1+b1	Linux Sta
mawk	1.3.4.20200120-3.1+b1	Pattern sc
media-types	10.0.0	List of sta
molecule	24.9.0+aa1.1.2	Molecule
molecule-libvirt	0.2.0+aa1.1.0	libvirt Mo
molecule-molecule-brest	1.1.0+aa1.1.0	Brest Mol
molecule-molecule-yandexcloud	0.3.1+aa1.1.0	YandexCl
molecule-plugins	23.5.3+aa1.1.0	Molecule
mount	2.38.1-5+deb12u1+astra13+b1	tools for m
ncurses-base	6.4-4+b1	basic ter
ncurses-bin	6.4-4+b1	terminal-
netbase	6.4	Basic TCP
openscap-common	1.3.7+dfsg-1+deb12u1+b1	OpenSCA
openscap-scanner	1.3.7+dfsg-1+deb12u1+b1	OpenScap

Таблица 21 – продол

Пакет	Версия	Описани
openssh-client	1:9.6p1-2~deb10u1astra8se5	secure sh
openssl	3.2.0-2-astra5+ci1	Secure So
oval-db	0.0.1+ci3	Install ov
passwd	1:4.13+dfsg1-1.astra.se10+b1	change a
perl	5.36.0-7+deb12u1+ci202404171352+astra2	Larry Wal
perl-base	5.36.0-7+deb12u1+ci202404171352+astra2	minimal P
perl-modules-5.36	5.36.0-7+deb12u1+ci202404171352+astra2	Core Perl
pinentry-curses	1.2.1-1+b1	curses-ba
podman	4.7.2.astra1	engine to
procps	2:4.0.2-3+b1	/proc file
pytest-ansible	24.9.0+aa1.1.0	Plugin for
python3	3.11.2-1+b1	interactiv
python3-attr	22.2.0-1	Attributes
python3-bcrypt	3.2.2-1+b1	password
python3-blinker	1.5-1	Fast, simp
python3-certifi	2022.9.24-1	root certi
python3-cffi	1.15.1-5+b1	Foreign F
python3-cffi-backend:amd64	1.15.1-5+b1	Foreign F
python3-chardet	5.1.0+dfsg-2	Universal
python3-charset-normalizer	3.0.1-2	charset, e
python3-click	8.1.3-2	Wrapper
python3-colorama	0.4.6-2	Cross-pla
python3-cryptography	38.0.4-3+ci202405171208+astra3+b1	Python lib
python3-daemon	2.3.2-1	library fo
python3-decorator	5.1.1-3	simplify u
python3-dev	3.11.2-1+b1	header fil
python3-distlib	0.3.6-1	low-level
python3-distro	1.8.0-1	Linux OS
python3-distutils	3.11.2-3+b1	distutils p
python3-dnspython	2.3.0-1+b1	DNS toolk
python3-docker	5.0.3-1	Python 3
python3-dogpile.cache	1.1.8-2	caching f
python3-execnet	1.9.0-1	rapid mul
python3-filelock	3.9.0-1	platform
python3-httplib2	0.20.4-3	comprehe
python3-idna	3.3-1	Python ID
python3-importlib-metadata	4.12.0-1	library to
python3-iniconfig	1.1.1-2	brain-dea
python3-iso8601	1.0.2-1	Python m
python3-jinja2	3.1.2-1	small but
python3-jmespath	1.0.1-1	JSON Mat
python3-json-pointer	2.3-2	resolve JS
python3-jsonpatch	1.32-2	library to
python3-jsonschema	4.10.3-1	An(other)
python3-jwt	2.6.0-1+b1	Python 3
python3-keystoneauth1	5.0.0-2	authentic
python3-lib2to3	3.11.2-3+b1	Interactiv
python3-lockfile	1:0.12.2-2.2	file lockin
python3-lxml:amd64	4.9.2-1+b1	pythonic
python3-mako	1.2.4+ds-1	fast and l
python3-markdown-it	2.1.0-5	Python po
python3-markupsafe	2.1.2-1+b1	HTML/XHT
python3-mdurl	0.1.2-1	Python po
python3-minimal	3.11.2-1+b1	minimal s
python3-more-itertools	8.10.0-2	library wi
python3-mypy-extensions	0.4.3-4	Experime
python3-nacl	1.5.0-2+b1	Python bi

Таблица 21 – продол

Пакет	Версия	Описани
python3-netaddr	0.8.0-2	manipula
python3-netifaces:amd64	0.11.0-2+b1	portable
python3-ntlm-auth	1.4.0-2	NTLM low
python3-oauthlib	3.2.2-1+b1	generic, s
python3-openssl	23.0.0-1+b1	Python 3
python3-os-service-types	1.7.0-2	lib for co
python3-packaging	23.0-1	core utilit
python3-paramiko	2.12.0-2+b1	Make ssh
python3-pathspec	0.11.0-1	utility libr
python3-pbr	5.10.0-2+b1	inject use
python3-pexpect	4.8.0-4	Python 3
python3-pip	23.0.1+dfsg-1	Python pa
python3-pkg-resources	66.1.1-1	Package
python3-platformdirs	2.6.0-1	determin
python3-pluggy	1.0.0+repack-1	plugin an
python3-ply	3.11-5+b1	Lex and Y
python3-protobuf	3.21.12-3+b1	Python 3
python3-ptyprocess	0.7.0-5	Run a sub
python3-py	1.11.0-1	Advanced
python3-pycparser	2.21-1	C parser
python3-pygments	2.14.0+dfsg-1	syntax hi
python3-pyparsing	3.0.9-1	alternativ
python3-pyrsistent:amd64	0.18.1-1+b1	persisten
python3-pytest	7.2.1-2	Simple, p
python3-pytest-xdist	3.1.0-1	xdist plug
python3-requests	2.28.1+dfsg-1+b1	elegant a
python3-requests-ntlm	1.1.0-3+b1	Adds sup
python3-requestsexceptions	1.4.0-3+b1	import ex
python3-resolvelib	0.9.0-2	module to
python3-rich	13.3.1-1	render ric
python3-ruamel.yaml.clib:amd64	0.2.7-1+b1	C version
python3-selinux	3.4-1+b1	Python3
python3-setuptools	66.1.1-1	Python3
python3-six	1.16.0-4	Python 2
python3-stevedore	4.0.2-2	manage d
python3-tblib	1.7.0-3	Python 3
python3-urllib3	1.26.12-1+ci202406111901+astra2	HTTP libr
python3-websocket	1.2.3-1	WebSock
python3-wheel	0.38.4-2	built-pack
python3-winrm	0.3.0-4+deb12u1	Python 3
python3-xmltodict	0.13.0-1	Makes wo
python3-yaml	6.0-3+b1	YAML par
python3-zipp	1.0.0-6	pathlib-co
python3.11	3.11.2-6astra4+b1	Interactiv
python3.11-aenum	3.1.15+aa1.1.0	Advanced
python3.11-bindep	2.11.0+aa1.2.0	Binary de
python3.11-black	24.10.0+aa1.1.0	The unco
python3.11-bracex	2.5.post1+aa1.1.0	Bash styl
python3.11-cachetools	5.5.0+aa1.1.0	Extensibl
python3.11-chardet	5.2.0+aa1.1.0	Universal
python3.11-click-help-colors	0.9.4+aa1.1.0	Colorizati
python3.11-cryptography	42.0.8+aa1.1.0	cryptogra
python3.11-dev	3.11.2-6astra4+b1	Header fi
python3.11-dicttoxml	1.7.16+aa1.1.0	Converts
python3.11-distlib	0.3.9+aa1.1.0	Distributi
python3.11-enrich	1.2.7+aa1.1.0	enrich
python3.11-filelock	3.16.1+aa1.1.0	A platform

Таблица 21 – продол

Пакет	Версия	Описани
python3.11-googleapis-common-protos	1.65.0+aa1.1.0	Common
python3.11-grpcio	1.67.1+aa1.1.0	HTTP/2-b
python3.11-minimal	3.11.2-6astra4+b1	Minimal s
python3.11-onigurumacffi	1.3.0+aa1.1.0	python cf
python3.11-openstacksdk	4.1.0+aa1.1.0	An SDK fo
python3.11-packaging	24.2+aa1.1.0	Core utili
python3.11-parsley	1.3+aa1.1.0	Parsing a
python3.11-platformdirs	4.3.6+aa1.1.0	A small P
python3.11-pluggy	1.5.0+aa1.1.0	plugin an
python3.11-protobuf	4.25.5+aa1.1.0	Protocol B
python3.11-pyjwt	2.9.0+aa1.1.0	JSON Web
python3.11-pyone	6.6.3+aa1.1.0	Python Bi
python3.11-pyproject-api	1.8.0+aa1.1.0	API to int
python3.11-pypsrp	0.8.1+aa1.1.0	PowerShe
python3.11-pyspnego	0.11.0+aa1.1.0	Windows
python3.11-python-gnupg	0.5.2+aa1.1.0	A wrappe
python3.11-python-vagrant	1.0.0+aa1.1.0	Python bi
python3.11-requests	2.32.3+aa1.1.0	Python H
python3.11-ruamel.yaml	0.18.6+aa1.1.0	ruamel.ya
python3.11-subprocess-tee	0.4.2+aa1.1.0	subproce
python3.11-tox	4.23.2+aa1.1.0	tox is a g
python3.11-urllib3	2.2.3+aa1.1.0	HTTP libr
python3.11-virtualenv	20.27.1+aa1.1.0	Virtual Py
python3.11-wcmatch	10.0+aa1.1.0	Wildcard/
python3.11-yamllint	1.35.1+aa1.1.0	A linter fo
python3.11-yandexcloud	0.324.0+aa1.1.0	The Yand
readline-common	8.2-1.3+b1	GNU read
rpcsvc-proto	1.4.3-1+b1	RPC proto
runc	1.1.12~astra1+ci3+b1	Open Con
sed	4.9-1+b1	GNU strea
sensible-utils	0.0.17+nmu1	Utilities fo
slirp4netns	1.2.0-1+b1	User-mod
sshpass	1.09-1+b1	Non-inter
systemd	252.17-1~deb12u1astra.se1+ci13+b1	system a
systemd-sysv	252.17-1~deb12u1astra.se1+ci13+b1	system a
sysvinit-utils	3.06-4+b1	System-V
tar	1.34+dfsg-1.2+deb12u1+b1	GNU vers
tcl-expect:amd64	5.45.4-2+ci202403211247+astra1+b1	Automate
tcl8.6	8.6.13+dfsg-2+b1	Tcl (the To
tox-ansible	24.9.0+aa1.1.0	A radical
tzdata	2024a-4	time zone
uidmap	1:4.13+dfsg1-1.astra.se10+b1	programs
usrmerge	37~deb12u1+b1	Convert t
util-linux	2.38.1-5+deb12u1+astra13+b1	miscellan
util-linux-extra	2.38.1-5+deb12u1+astra13+b1	interactiv
zlib1g:amd64	1:1.2.13.dfsg-1+b1	compress
zlib1g-dev:amd64	1:1.2.13.dfsg-1+b1	compress

Пакеты Python

Пакет
aenum
ansible-builder @ file:///docker_share/dist/ansible_builder-3.1.0-py3-none-any.whl
ansible-compat @ file:///docker_share/dist/ansible_compat-24.9.1-py3-none-any.whl
ansible-core

продолжается на

Таблица 22 – продолжение с предыдущей страницы

Пакет
ansible-creator @ file:///docker_share/dist/ansible_creator-24.10.1-py3-none-any.whl
ansible-lint @ file:///docker_share/dist/ansible_lint-24.9.2-py3-none-any.whl
ansible-navigator @ file:///docker_share/dist/ansible_navigator-24.9.0-py3-none-any.whl
ansible-pylibssh
ansible-runner @ file:///docker_share/dist/ansible_runner-2.4.0-py3-none-any.whl
ansible-sign @ file:///docker_share/dist/ansible_sign-0.1.1-py3-none-any.whl
attrs
bcrypt
bindep
black
blinker
bracex
cachetools
certifi
cffi
cfgv
chardet
charset-normalizer
click
click-help-colors
colorama
cryptography
Cython
decorator
dicttoxml
distlib
distro
dnspython
docker
dogpile.cache
dumb-init
enrich
execnet
filelock
googleapis-common-protos
grpcio
httplib2
identify
idna
importlib-metadata
iniconfig
iso8601
Jinja2
jmespath
jsonpatch
jsonpointer
jsonschema
jxmlease
keystoneauth1
lockfile
lxml
Mako
markdown-it-py
MarkupSafe
mdurl
molecule @ file:///docker_share/dist/molecule-24.9.0-py3-none-any.whl
molecule-libvirt @ file:///docker_share/dist/molecule_libvirt-0.2.0-py3-none-any.whl

продолжается на

Таблица 22 – продолжение с предыдущей страницы

Пакет
molecule_molecule_brest @ file:///docker_share/dist/molecule_molecule_brest-1.1.0-py3-none-any.whl
molecule_molecule_yandexcloud @ file:///docker_share/dist/molecule_molecule_yandexcloud-0.3.1-py3-no
molecule-plugins @ file:///docker_share/dist/molecule_plugins-23.5.3-py3-none-any.whl
more-itertools
mypy-extensions
ncclient
netaddr
netifaces
nodeenv
ntlm-auth
oauthlib
onigurumacffi
openstacksdk
os-service-types
packaging
paramiko
Parsley
pathspec
pbr
pexpect
platformdirs
pluggy
ply
pre_commit
protobuf
ptyprocess
py
pycparser
Pygments
PyJWT
PyNaCl
pyone
pyOpenSSL
pyparsing
pyproject-api
pypsrp @ file:///docker_share/dist/pypsrp-0.8.1-py3-none-any.whl
pyrsistent
pyspnego
pytest
pytest-ansible @ file:///docker_share/dist/pytest_ansible-24.9.0-py3-none-any.whl
pytest-xdist
python-daemon
python-gnupg
python-vagrant
pywinrm
PyYAML
requests
requestsexceptions
requests-ntlm
resolvelib
rich
ruamel.yaml
ruamel.yaml.clib
selinux
six
stevedore
subprocess-tee

продолжается на

Таблица 22 – продолжение с предыдущей страницы

Пакет
tblib
textfsm
tox
tox-ansible @ file:///docker_share/dist/tox_ansible-24.9.0-py3-none-any.whl
ttp
urllib3
virtualenv
wcmatch
websocket-client
xmltodict
yamllint
yandexcloud
zipp

Коллекции Ansible

Коллекция	Версия
ansible.netcommon	7.1.0
ansible.posix	2.0.0
ansible.utils	5.1.2
astra.brest	4.0.3
astra.yandexcloud	0.4.0
community.crypto	2.24.0
community.general	10.3.0
community.libvirt	1.3.1

17.8.12 Файл определения среды исполнения

Файл определения среды исполнения имеет формат YAML и содержит настройки, используемые *Ansible Builder* при создании образа контейнера среды исполнения.

version

Целочисленное значение, которое задает версию схемы файла определения среды исполнения.

Значение по умолчанию: 1.

При использовании Ansible Builder версии `3.x` рекомендуется в этом поле установить значение 3:

```
---
version: 3
```

build_arg_defaults

Значения по умолчанию для переменных окружения, используемых при сборке образа среды исполнения:

- `ANSIBLE_GALAXY_CLI_COLLECTION_OPTS`;
- `ANSIBLE_GALAXY_CLI_ROLE_OPTS`;
- `PKGMGR_PRESERVE_CACHE`.

Значения, указанные в `build_arg_defaults`, будут сохранены в файл определения образа `Containerfile` и применены даже при запуске сборки образа среды исполнения вручную.

Подробное описание аргументов см. в *справочнике параметров и команд Ansible Builder*.

Примечание

Значения, заданные в значении аргумента командной строки `--build-arg`, имеют более высокий приоритет, чем заданные в файле определения среды исполнения.

dependencies

В разделе dependencies указываются пакеты и библиотеки, которые необходимо установить в конечный образ.

dependencies.ansible_core

Версия пакета `ansible-core`.

Важно

Версия пакета указывается в соответствии с требованиями **PEP 508**[256].

Список 33: Пример заполнения

```
dependencies:
  ansible_core:
    package_pip: ansible-core==2.15.6
```

dependencies.ansible_runner

Версия пакета `ansible-runner`.

Важно

Версия пакета указывается в соответствии с требованиями **PEP 508**[257].

Список 34: Пример заполнения

```
dependencies:
  ansible_runner:
    package_pip: ansible-runner==2.3.2
```

dependencies.galaxy

Список зависимостей Ansible или путь к файлу с таким списком.

Путь к файлу зависимостей указывается относительно каталога, содержащего файл определения среды исполнения. Рекомендуемое имя файла – `requirements.yml`.

При явном указании зависимостей в файле определения среды исполнения списки коллекций и ролей указываются в значениях параметров `dependencies.galaxy.collections` и `dependencies.galaxy.roles` соответственно.

Примеры заполнения:

Список 35: Имя файла со списком зависимостей

```
dependencies:
  galaxy: requirements.yml
```

[256] https://peps.python.org/pep-0508/
[257] https://peps.python.org/pep-0508/

Список 36: Список зависимостей

```
dependencies:
  galaxy:
    collections:
      - astra.ald_pro
      - astra.brest
      - astra.rupost
        version: 4.0.0
```

dependencies.python

Список зависимостей Python или путь к файлу со списком таких зависимостей.

Путь к файлу зависимостей указывается относительно каталога, содержащего файл определения среды исполнения. Рекомендуемое имя файла – `requirements.txt`.

Важно

Содержимое файла зависимостей Python должно быть оформлено в соответствии с требованиями **PEP 508**[258].

dependencies.python_interpreter.package_system

Название системного пакета, отвечающего за интерпретатор Python нужной версии.

Список 37: Пример заполнения

```
dependencies:
  python_interpreter:
    package_system: python3.11
```

dependencies.python_interpreter.python_path

Путь к интерпретатору Python, который должен использоваться по умолчанию.

Список 38: Пример заполнения

```
dependencies:
  python_interpreter:
    python_path: /usr/bin/python3.11
```

dependencies.system

Список системных пакетов в формате BINDEP[259] или путь к файлу с таким списком.

Рекомендуемое название файла – `bindep.txt`.

[258] https://peps.python.org/pep-0508/
[259] https://docs.opendev.org/opendev/bindep/latest/readme.html

Список 39: Пример заполнения

```
dependencies:
  system: bindep.txt
```

dependencies.exclude

Список зависимостей, которые *не* должны быть установлены в образ. Значения задаются в следующих параметрах:

- `dependencies.exclude.python` – список зависимостей Python.
- `dependencies.exclude.system` – список системных зависимостей.
- `dependencies.exclude.all_from_collections` – исключить все системные зависимости и зависимости Python, связанные с указанными коллекциями.

Сопоставление названий исключаемых зависимостей может выполняться по простому совпадению строки или регулярному выражению. В обоих случаях сравнение выполняется без учета регистра.

Чтобы строка интерпретировалась как регулярное выражение, добавьте в ее начало символ ~.

Примечание

Регулярное выражение должно соответствовать полному названию пакета или коллекции. Например, регулярное выражение `~foo.` соответствует строке `foobar` частично, а `~foo.+` – полностью.

При обеих формах сопоставления строка исключения будет сравниваться с простым названием пакета или коллекции. Например, чтобы исключить системный пакет с названием `foo [!platform:gentoo]`, строка исключения должна иметь значение `foo`. Чтобы исключить пакет Python `bar==1.0.0`, строка должна иметь значение `bar`.

Примеры:

Список 40: Простое сопоставление

```
dependencies:
 exclude:
   python:
     - docker
   system:
     - python3-Cython
```

Список 41: Регулярное выражение

```
dependencies:
  exclude:
    all_from_collections:
      - ~community\..+
```

Список 42: Имена файлов, содержащих различные зависимости

```
dependencies:
  python: requirements.txt
  system: bindep.txt
  galaxy: requirements.yml
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  ansible_core:
    package_pip: ansible-core==2.15.6
  ansible_runner:
    package_pip: ansible-runner==2.3.1
  python_interpreter:
    package_system: "python310"
    python_path: "/usr/bin/python3.10"
```

Список 43: Встроенные значения

```
dependencies:
  python:
    - pywinrm
  system:
    - iputils [platform:rpm]
  galaxy:
    collections:
      - name: community.windows
      - name: ansible.utils
        version: 2.10.1
  ansible_core:
    package_pip: ansible-core==2.15.6
  ansible_runner:
    package_pip: ansible-runner==2.3.1
  python_interpreter:
    package_system: "python310"
    python_path: "/usr/bin/python3.10"
```

images.base_image.name

Название базового образа контейнера.

Список 44: Пример заполнения

```
images:
  base_image:
    name: hub.astra-automation.ru/aa-1.2/aa-minimal-ee:latest
```

additional_build_files

Список файлов, которые должны быть добавлены в контекст сборки. Затем на них можно ссылаться или копировать в образ с помощью команд, указанных в `additional_build_steps`.

Каждый элемент списка должен быть словарем, содержащим следующие ключи:

- `src` – путь к исходному файлу или каталогу, который необходимо добавить в контекст сборки. Это может быть либо абсолютный путь (например, `/home/user/.ansible.cfg`), либо путь относительно файла описания среды исполнения. Относительные пути могут быть глобальным выражением, соответствующим одному или нескольким файлам (например, `files/*.cfg`).
- `dest` – путь к целевому размещению файла в подкаталоге контекста сборки. Если путь содержит названия несуществующих каталогов, они будут созданы автоматически.

Список 45: Пример заполнения

```
additional_build_files:
  - src: files/ansible.cfg
    dest: configs
```

additional_build_steps

Дополнительные команды, выполняемые на различных этапах сборки. Названия вложенных параметров состоят из префикса prepend_ или append_ и названия этапа.

Команды, перечисленные в разделах с префиксом prepend_, выполняются *до* соответствующих этапов.

Команды, перечисленные в разделах с префиксом append_, выполняются *после* соответствующих этапов.

Таким образом, необходимые команды могут быть указаны в следующих разделах, в порядке выполнения:

1. prepend_base;
2. append_base;
3. prepend_galaxy;
4. append_galaxy;
5. prepend_builder;
6. append_builder;
7. prepend_final;
8. after_final.

Команды могут быть заданы в виде многострочной записи или в отдельных элементах списка:

Список 46: Многострочная запись

```
additional_build_steps:
  prepend_final: |
    RUN whoami
    RUN cat /etc/os-release
```

Список 47: Список команд

```
additional_build_steps:
  append_final:
     - RUN echo
     - RUN ls -la /etc
```

ГЛАВА 18

Примеры решений

Приведенные примеры показывают различные области применения Automation Controller, Private Automation Hub, Ansible Navigator и коллекций Ansible для выполнения типовых задач управления IT-инфраструктурой.

18.1 Типовой проект

Использование типовой структуры проектов Ansible позволяет упростить их создание, использование и поддержку.

18.1.1 Файлы и каталоги

Структура типового проекта Ansible, для запуска которого используется *Automation Controller* или *Ansible Navigator*, имеет следующий вид:

```
project/
├── playbooks/
│   ├── playbook1.yml
│   ├── playbook2.yml
│   │   ...
│   └── playbookK.yml
├── environments/
│   ├── dev/
│   ├── stage/
│   │   ...
│   └── prod/
│       ├── group_vars/
│       │   ├── main.yml
│       │   ├── smth.yml
│       │   └── smth_else.yml
│       └── inventory.yml
├── vars/
│   ├── vars1.yml
│   ├── vars2.yml
│   │   ...
│   └── varsN.yml
├── files/
│   ├── file1
│   ├── file2
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
│   │   ...
│   └── fileM
└── execution-environment.yml
```

Указанные файлы и каталоги используются для следующих целей:

- `playbooks/`.

  Каталог с файлами наборов сценариев. Если проект содержит только один набор сценариев, допускается разместить файл с ним в корневом каталоге проекта.

- `environments/`.

  Каталоги с файлами для различных окружений.

  Внутри `environemnts/` для каждого окружения рекомендуется создать собственный каталог, в котором разместить:

  - каталог `group_vars/` с файлами переменных для групп узлов;
  - файл инвентаря `inventory.yml`.

  Примечание

  Описание инвентаря необходимо только в том случае, когда проект разрабатывается для определенного списка управляемых узлов.

  Если проект разрабатывается под одно окружение, каталог `environments/` и его подкаталоги можно не создавать, а его содержимое разместить в корне проекта:

  Список 1: Структура проекта для одного окружения

```
project/
├── playbooks/
│   ├── playbook1.yml
│   ├── playbook2.yml
│   │   ...
│   └── playbookK.yml
├── group_vars/
│   ├── main.yml
│   ├── smth.yml
│   └── smth_else.yml
├── inventory.yml
├── vars/
│   ├── vars1.yml
│   ├── vars2.yml
│   │   ...
│   └── varsN.yml
├── files/
│   ├── file1
│   ├── file2
│   │   ...
│   └── fileM
└── execution-environment.yml
```

  Рекомендуемое название файла инвентаря в этом случае – `inventory.yml`.

- `vars/`.

  Каталог с файлами, определяющими значения переменных для сценариев автоматизации.

  Если все переменные хранятся в одном файле, допускается разместить его в корневом каталоге проекта. Рекомендуемое название файла в этом случае – `vars.yml`.

- `files/`.

  Каталог с произвольным названием, который создают, если это необходимо для конкретного проекта. Он может понадобиться, например, для копирования файлов на управляемые узлы или для сборки образа среды исполнения.

- `execution-environment.yml`.

  Файл определения среды исполнения, используемой для запуска набора сценариев. Подробности см. в *инструкции по созданию собственных образов среды исполнения*.

Примечание

Если для указания зависимостей вы используете файл `requirements.yml`, поместите его в корневой каталог проекта.

18.1.2 Рекомендуемый порядок действий

При работе над проектом рекомендуется следующий порядок действий:

1. Определите список используемых коллекций, изучите особенности их использования.
2. Разработайте код и отладьте его с помощью *Ansible Navigator*. В производственной среде в качестве EE рекомендуется использовать образ `aa-full-ee`. Если он не содержит необходимых коллекций, создайте *собственный образ EE* и опубликуйте его в приватном реестре коллекций как описано в документе *Загрузка образов среды исполнения*.
3. Опубликуйте код проекта в системе управления исходным кодом (SCM, Source Code Management).

18.1.3 Применение проекта

Для запуска проекта рекомендуется использовать *Automation Controller* или *Ansible Navigator*.

Automation Controller

Чтобы использовать Automation Controller для запуска проекта, выполните следующие действия:

1. Создайте полномочия нужного типа, следуя *инструкции*.

   Понадобятся как минимум следующие полномочия:

Тип полномочия	Тип ресурса
Реестр контейнеров	Private Automation Hub
Система управления версиями (Source Control)	Система управления кодом, в которой хранится проект
Машина (Machine)	Управляемые узлы

2. Добавьте в Automation Controller образ среды исполнения, следуя *инструкции по загрузке образов*.
3. Создайте проект согласно *инструкции по созданию проектов*.
4. *Создайте инвентарь требуемого типа* и добавьте в него сведения об управляемых узлах.
5. Используйте набор сценариев из проекта для *создания шаблона заданий*.
6. Запустите задание, следуя *инструкции по запуску заданий на основе шаблона*.

Ansible Navigator

Чтобы использовать Ansible Navigator для запуска проекта, выполните следующие действия:

1. Загрузите код проекта в рабочий каталог.
2. Создайте в корневом каталоге проекта файл `ansible.cfg`, в котором укажите необходимые настройки, например, путь к инвентарю.
3. Для запуска набора сценариев используйте команду:

   ```
   ansible-navigator run --eei <image> path/to/playbook.yml
   ```

   Здесь:

   - <image> – название образа среды исполнения, используемого для запуска заданий;
   - `/path/to/playbook.yml` – путь к файлу набора сценариев относительно корневого каталога проекта.

18.2 Развертывание веб-сервера NGINX

Использование коллекции `astra.nginx` из реестра Automation Hub[260] позволяет автоматизировать процесс установки и настройки веб-сервера NGINX на необходимом количестве узлов.

18.2.1 Описание сценария

Процесс установки и настройки веб-сервера NGINX состоит из следующих этапов:

1. Создание стенда.

   В качестве стенда рекомендуется использовать управляемые узлы, обладающие следующими характеристиками:

Характеристика	Значение
Кол-во ядер CPU	≥ 2
Кол-во RAM, ГБ	≥ 2
Размер хранилища, ГБ	≥ 20
ОС	Astra Linux Special Edition 1.8.1.UU2

   Для подключения к стенду используется протокол SSH.

   Если у вас нет подходящего стенда, вы можете создать его с помощью *Vagrant*, используя приложенный `Vagrantfile`.

2. Подготовка проекта Ansible.

   Проект включает в себя следующие файлы:

   - `ansible.cfg` – настройки Ansible;
   - `inventory.yml` – описание инвентаря;
   - `playbooks/nginx.yml` – набор сценариев настройки управляемых узлов.

   В демонстрационных целях для NGINX задаются следующие настройки:

   - `worker_connections`: `2048`;
   - `keepalive_timeout`: `30`.

3. Запуск набора сценариев.
4. Проверка работоспособности развернутой инфраструктуры.

[260] https://hub.astra-automation.ru/

18.2.2 Подготовка к работе

Подготовьте окружение к управлению задачами автоматизации:

1. Настройте управляемые узлы согласно *инструкции*.
2. Настройте Private Automation Hub на синхронизацию с Automation Hub.
3. Загрузите в Private Automation Hub образы среды исполнения `aa-1.2/aa-full-ee`.
4. Изучите описание[261] коллекции `astra.nginx`.
5. Подготовьте каталог для хранения файлов проекта.

Automation Controller

Если для управления задачами автоматизации используется Automation Controller, выполните следующие действия:

1. В одном из сервисов хранения репозиториев Git создайте пустой репозиторий для проекта.
2. Клонируйте этот репозиторий на локальный компьютер.

Ansible Navigator

Если для управления задачами автоматизации используется Ansible Navigator, создайте в любом удобном месте пустой каталог, например:

```
mkdir ~/ansible-nginx/
```

18.2.3 Подготовка проекта Ansible

Подготовьте ресурсы, необходимые для использования Ansible.

1. Создайте файл `ansible.cfg` со следующим содержимым:

Список 2: `ansible.cfg`

```
[defaults]
host_key_checking = False
inventory = inventory.yml
```

2. Создайте файл инвентаря `inventory.yml`, например:

Список 3: `inventory.yml`

```
---
all:
  hosts:
    node1.example.com:
      ansible_host: 192.168.56.11
      ansible_user: vagrant
```

В этом примере показаны настройки подключения к одному управляемому узлу.

3. Создайте каталог `playbooks/`, а в нем – набор сценариев `nginx.yml`:

Список 4: `playbooks/nginx.yml`

```
---
- name: Install and configure NGINX server
  hosts: all
  become: true

  roles:
```

(продолжается на следующей странице)

[261] https://hub.astra-automation.ru/ui/repo/validated/astra/nginx/docs/

(продолжение с предыдущей страницы)

```
    - role: astra.nginx.nginx
      vars:
        nginx_worker_connections: 2048
        nginx_keepalive_timeout: 30
```

В этом файле настройки NGINX заданы через значения переменных роли `astra.nginx.nginx`.

4. Если для запуска заданий автоматизации используется Automation Controller, зафиксируйте сделанные в проекте изменения и опубликуйте их в репозитории Git.

18.2.4 Запуск набора сценариев

Способ запуска набора сценариев зависит от инструмента, используемого для управления задачами автоматизации.

Automation Controller

Для запуска задания автоматизации в Automation Controller выполните следующие действия:

1. Для доступа к управляемым узлам создайте необходимое количество полномочий типа *Машина* (Machine).
2. Если для доступа к репозиторию с кодом проекта требуется авторизация, создайте полномочие типа *Система управления версиями* (Source Control).
3. Создайте проект со следующими свойствами:
   - **Название**: *NGINX*.
   - **Среда исполнения**: выберите среду исполнения, использующую образ `aa-full-ee`.

     Примечание

     Среда исполнения на основе образа `aa-full-ee` используется в Automation Controller по умолчанию. Вы можете добавить среду исполнения самостоятельно, следуя *инструкции*.

   - **Тип системы управления исходными данными**: **Git**.
   - **URL системы управления исходными данными**: укажите ссылку на репозиторий с кодом проекта.
   - **Полномочия на систему управления исходными данными**: если для доступа к репозиторию с кодом проекта требуется авторизация, выберите созданное ранее полномочие типа *Система управления версиями* (Source Control).
4. Создайте обычный инвентарь и добавьте в него сведения об управляемых узлах.

   Совет

   Вы можете использовать файл `inventory.yml` в качестве источника сведений об управляемых узлах.

5. Создайте шаблон задания со следующими свойствами:
   - **Тип задания**: **Выполнение**.
   - **Инвентарь**: выберите инвентарь, созданный на предыдущем шаге.
   - **Проект**: *NGINX*.

- **Playbook**: `playbooks/nginx.yml`.

6. Запустите задание на основе созданного шаблона.

Ansible Navigator

При использовании Ansible Navigator для запуска набора сценариев выполните команду:

```
ansible-navigator run playbooks/nginx.yml \
   --eei private-hub.example.com/aa-1.2/aa-full-ee
```

При этом открывается псевдографический интерфейс, в котором показывается ход выполнения, например:

```
  Play name                            Ok Changed Unreachable  Failed Skipped Ignored In␣
↪progress  Task count   Progress
0|Install and configure NGINX server   5       0           0       0       0       0    ␣
↪      1           6   ██████


^b/PgUp page up       ^f/PgDn page down       ↑↓ scroll       esc back       [0-9] goto ␣
↪     :help help      Running
```

Для просмотра более подробной информации о ходе выполнения нажмите `0`.

Статус задания выводится в правом нижнем углу. Дождитесь перехода задания в статус `Successful`.

18.2.5 Проверка работоспособности развернутой инфраструктуры

Чтобы проверить корректность развертывания веб-сервера NGINX, выполните следующие действия:

1. Подключитесь к любому из управляемых узлов по SSH.
2. Для проверки статуса службы `nginx` выполните команду:

   ```
   systemctl status nginx
   ```

   При успешном развертывании в терминал выводится сообщение вида:

   ```
   ● nginx.service - A high performance web server and a reverse proxy server
      Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset:␣
   ↪enabled)
      Active: active (running) since Mon 2024-10-21 09:28:25 MSK; 2h 22min ago
        Docs: man:nginx(8)
    Main PID: 643 (nginx)
       Tasks: 3 (limit: 2586)
      Memory: 4.6M
         CPU: 14ms
      CGroup: /system.slice/nginx.service
              ├─643 nginx: master process /usr/sbin/nginx -g daemon on; master_
   ↪process on;
              ├─645 nginx: worker process
              └─646 nginx: worker process
   ```

3. Убедитесь, что в файле `/etc/nginx/nginx.conf` значения параметров `worker_connections` и `keepalive_timeout` соответствуют заданным в наборе сценариев (2048 и 30 соответственно).

18.2.6 Особенности проекта

Обратите внимание на следующие особенности выполненного проекта:

- использование образа среды исполнения `aa-full-ee` в Automation Controller и Ansible Navigator;

- настройка параметров роли через переменные.

18.2.7 Заключение

В этом сценарии вы познакомились с основными шагами по развертыванию веб-сервера NGINX с использованием коллекции `astra.nginx`. Из всей последовательности шагов важно выделить следующие действия:

- подготовка окружения;
- описание параметров доступа к управляемым узлам в файле инвентаря;
- связывание управляемых узлов и ролей в наборе сценариев;
- запуск набора сценариев с помощью Ansible Navigator и в Automation Controller;
- проверка корректности развертывания NGINX.

18.3 Пример организации процесса CI/CD

Рассмотрим организацию простейшего процесса непрерывной разработки и развертывания программных продуктов. При необходимости вы можете воспроизвести данный сценарий в собственном окружении.

18.3.1 Описание сценария

Приведен пример планирования и настройки непрерывного процесса разработки программного проекта (CI, continuous integration) и его развертывания (CD, continuous deployment).

Особенности сценария:

- Проект хранится в публично доступном репозитории под управлением Git, используя GitHub.
- Обработку процессов CI/CD выполняет Automation Controller.
- При создании запроса (PR, pull request) на внесение изменений в какую-либо ветку репозитория GitHub вызывает процесс CI, целью которого является проверка корректности разработанного программного продукта. При исполнении контролируемая программа оповещает о корректности путем возврата соответствующего кода:
  - `0` – программа работает корректно;
  - `1` – в программе обнаружена ошибка.

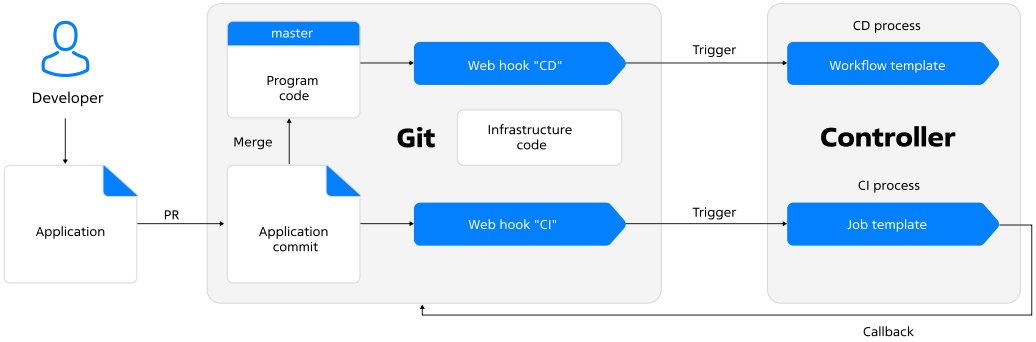
  При обнаружении ошибки в программе разработчик имеет возможность исправить код и повторить процесс CI.
- После слияния рабочей ветки с веткой `master` GitHub вызывает процесс CD, в котором контроллер проверяет, что объектом изменений является ветка `master`. В этом случае он запрашивает разрешение на обновление продукта и после одобрения выполняет обновление продукта.

18.3.2 Структура проекта

Проект реализации сценария включает разработку структуры и инфраструктурного кода.

Взаимодействие компонентов

Структурная схема сценария представляется в следующем виде:



Взаимодействие компонентов в процессе CI происходит следующим образом:

1. В репозитории GitHub разработчик вносит изменения в демонстрационную программу и создает запрос на внесение изменений в ветку `master` (PR).
2. GitHub перехватывает PR с помощью соответствующего объекта webhook и направляет запрос на выполнение процесса CI в контроллере. Запрос PR переходит в состояние ожидания ответа от контроллера.
3. Контроллер получает запрос на запуск задания из шаблона, реализующего процесс CI. В данном сценарии задание выводит содержимое программы в выходной поток и запускает программу для проверки корректности ее выполнения.
4. Контроллер возвращает результат обработки запроса в GitHub.
5. GitHub получает результат запроса и сообщает о готовности внести изменения в `master`.

CD является продолжением предыдущего процесса. Его можно выполнить в случае успешного завершения последнего:

1. Разработчик принимает решение о внесении изменений, представленных PR, в ветку `master` и начинает процесс слияния веток (Git merge).
2. После завершения обновления ветки `master` GitHub перехватывает запрос с помощью соответствующего объекта webhook и направляет запрос на выполнение процесса CD в контроллере.
3. Контроллер получает запрос на запуск заданий из шаблона потока, реализующего процесс CD. Он запускает выполнение цепочки заданий, включая получение одобрения от ответственного лица.

> **Примечание**
>
> В отличие от процесса CI, контроллер не возвращает результат обработки запроса в GitHub. Поэтому результаты процесса CD необходимо анализировать средствами контроллера.

Репозиторий проекта

Для простоты исполняемый код обслуживаемой демонстрационной программы разработчика (`project.sh`) и инфраструктурный код для реализации сценария находятся в одном репозитории в виде следующей структуры каталогов и файлов:

```
./
├── aac
│   ├── cd_check.yml
│   ├── cd_deploy.yml
│   └── ci.yml
└── project.sh
```

Назначение и состав файлов представлены в следующих секциях. Для воспроизведения сценария загрузите эти файлы в свой репозиторий GitHub в соответствии с представленной структурой.

Демонстрационная программа

Демонстрационная программа, к которой применены процессы CI/CD, состоит из одного скрипта shell `project.sh` (`download`):

Demo program

```
#!/bin/bash

# Use "exit 0" to simulate correct code
# Use "exit 1" to "exit 255" to simulate incorrect code
exit 1
```

Сценарии Ansible для CI/CD

Каждая часть процесса использует отдельные файлы сценариев.

Процесс CI состоит из одного задания, которое выполняется с помощью сценария из файла `aac/ci.yml` (`download`) на локальном узле, то есть на самом контроллере:

Сценарий CI

```
---
- hosts: localhost
  become: false
  gather_facts: false
  tasks:
    - name: Get code revision "{{ awx_webhook_event_ref }}"
      ansible.builtin.git:
        repo: "{{ awx_webhook_payload.repository.clone_url }}"
        dest: /tmp/project
        version: "{{ awx_webhook_event_ref }}"

    - name: Show main project
      ansible.builtin.shell:
        chdir: /tmp/project
        cmd: cat ./project.sh

    - name: Check main project
      ansible.builtin.shell:
        chdir: /tmp/project
        cmd: ./project.sh
```

Сценарий выполняет следующие действия:

1. Создает клон репозитория, адрес которого содержит переменная `awx_webhook_payload.repository.clone_url`, в каталоге `/tmp/project/`.
2. Выводит содержимое файла проекта в выходной поток командой `cat`.
3. Запускает на выполнение скрипт программы для тестирования ее работоспособности. По результатам выполнения контроллер определяет корректность внесенных изменений.

Процесс CD состоит из потока заданий, в который вовлечены следующие файлы сценариев:

1. `aac/cd_check.yml` (`download`) – проверка того, что вызов задания связан с обновлением ветки `master` в репозитории GitHub:

   ### Сценарий проверки CD

   ```
   ---
   - hosts: localhost
     become: false
     gather_facts: false
   ```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  tasks:
    - name: Check if push is into master branch
      ansible.builtin.fail:
        msg: This push is not into master branch
      when: awx_webhook_payload.ref != "refs/heads/master"
```

Сценарий завершает аварийно процесс CD, если обновление было не в ветке `master`. В противном случае процесс продолжается.

2. `aac/cd_deploy.yml (download)` – демонстрационное (фиктивное) развертывание продукта:

Сценарий развертывания

```
---
- hosts: localhost
  become: false
  gather_facts: false
  tasks:
    - name: Deploy project
      ansible.builtin.debug:
        msg: The project is being deployed.
```

Сценарий только выводит сообщение о выполнении развертывания.

18.3.3 Настройка процесса

После создания публичного репозитория необходимо выполнить настройки основных компонентов.

Настройка прав доступа

Для обеспечения взаимодействия компонентов проекта необходимо предоставить контроллеру полномочия на доступ к репозиторию GitHub. Это позволит контроллеру возвращать состояние процесса (callback) в GitHub. Полномочия необходимо предоставить через персональный токен доступа к репозиторию GitHub. Для его создания в GitHub существует следующая последовательность шагов в настройках:

profile > settings > developer settings > personal access tokens > tokens(classic) > generate new token

Примечание

Сгенерированный токен показывается только один раз. Для повторного использования необходимо сохранить его в надежном месте.

Контроллер

В контроллере необходимо произвести следующие настройки:

1. Задайте базовый URL сервиса, который для контроллера, установленном на одном узле, совпадает с URL этого контроллера. Шаги настройки:

   Настройки (Settings) > Система (System) > Общие системные настройки (Miscellaneous System settings) > Базовый URL сервиса (Base URL of the service)

   Примечание

   Базовый URL будет передан в GitHub для использования его в качестве ссылки на соответствующее задание контроллера.

2. Создайте организацию. Шаги настройки:

   Access > Organizations > Add

   Параметры настройки:

   - **Название**: *cicd*.

3. Создайте полномочие на доступ к репозиторию GitHub. Шаги настройки:

   Resources > Credentials > Add

   Параметры настройки:

   - **Название** (Name): *cicd*;
   - **Тип полномочия** (Credential Type): **Личный токен доступа к GitHub** (GitHub Personal Access Token);
   - **Сведения о типе** (Type Details) > **Токен** (Token): <содержимое токена, созданного для организации или учетной записи в GitHub>;
   - **Организация** (Organization): *cicd*.

4. Создайте проект на основе репозитория, созданного в GitHub. Шаги настройки:

   Resources > Projects > Add

   Параметры настройки:

   - **Название**: *cicd*;
   - **Организация** (Organization): *cicd*;
   - **Среда исполнения** (Execution Environment): *Default execution environment*;
   - **Тип системы управления исходными данными** (Source Control Type): *Git*;
   - **URL системы управления исходными данными** (Source Control URL): *https://github.com/<GitHub account or organization>/<repo-name>.git*.

5. Создайте шаблоны заданий. Шаги настройки:

   Resources > Templates > Add > Add job template

   Параметры настройки для шаблона CI:

   - **Название** (Name): *ci*;
   - **Тип задания** (Job Type): **Выполнение** (Run);
   - **Инвентарь** (Inventory): *Demo Inventory*;
   - **Проект** (Project): *cicd*;
   - **Playbook**: `aac/ci.yml`.
   - В секции **Опции** (Options) включите следующие флаги:
     - **Включить webhook** (Enable Webhook);
     - **Параллельные задания** (Enable Concurrent Jobs).
   - В секции **Подробности о webhook** (Webhook details) настройте следующие параметры:
     - **Сервис webhook** (Webhook Service): **GitHub**;
     - **Полномочия webhook** (Webhook Credential): *cicd*.

   Сохраните настройки.

   Параметры настройки для шаблона задания на шаге проверки процесса CD:

   - **Название** (Name): *cd_check*;
   - **Тип задания** (Job Type): **Выполнение** (Run);
   - **Инвентарь** (Inventory): *Demo Inventory*;

- **Проект** (Project): *cicd*;
- **Playbook**: `aac/cd_check.yml`.

Сохраните настройки.

Параметры настройки для шаблона задания на шаге развертывания процесса CD:

- **Название** (Name): *cd_deploy*;
- **Тип задания** (Job Type): **Выполнение** (Run);
- **Инвентарь** (Inventory): *Demo Inventory*;
- **Проект** (Project): *cicd*;
- **Playbook**: `aac/cd_deploy.yml`.

Сохраните настройки.

6. Создайте шаблон потока заданий для процесса CD. Шаги настройки:

   Resources > Templates > Add > Add workflow template

   Параметры настройки:

   - **Название** (Name): *cd*;
   - **Организация** (Organization): *cicd*;
   - **Инвентарь** (Inventory): *Demo Inventory*.
   - В секции **Опции** (Options) выберите следующие опции:
     - **Включить webhook** (Enable Webhook);
     - **Параллельные задания** (Enable Concurrent Jobs).
   - В секции **Подробности о webhook** (Webhook details) настройте следующие параметры:
     - **Сервис webhook** (Webhook Service): **GitHub**;
     - **Полномочия webhook** (Webhook Credential): *cicd*.

   Сохраните настройки.

   Во всплывающем окне **Пожалуйста, нажмите кнопку «Пуск», чтобы начать** (Please click the Start button to begin) нажмите кнопку **Пуск** (Start) для настройки потока с помощью визуализатора.

   - Создайте узел на основе шаблона `cd_check` со следующими параметрами:
     - **Тип узла** (Node Type): **Шаблон задания** (Job Template);
     - **Название** (Name): *cd_check*.
   - Создайте узел согласования со следующими параметрами:
     - **Тип узла** (Node Type): **Согласование** (Approval);
     - **Название** (Name): *Согласование развертывания продукта*.
   - Создайте узел на основе шаблона `cd_deploy` со следующими параметрами:
     - **Тип узла** (Node Type): **Job Template**;
     - **Название** (Name): *cd_deploy*.

После сохранения каждого из двух шаблонов контроллер сгенерирует параметры доступа для шаблона в следующем виде:

- **Webhook URL**: <webhook URL>;
- **Ключ webhook** (Webhook key): <hash code>.

Параметры доступа необходимы для настройки объектов webhook в GitHub.

GitHub

В репозитории проекта необходимо настроить перехват событий с помощью объектов webhook.

Шаги настройки:

Settings > Webhooks > Add webhook

Создайте объекты перехвата событий со следующими параметрами:

- Webhook для CI:
  - **Payload URL**: <URL, использованный в настройках шаблона задания CI в контроллере>;
  - **Content type**: **application/json**;
  - **Secret**: <значение Webhook key из настроек шаблона задания CI в контроллере>;
  - **SSL verification**: **Disable**.

    Примечание

    Здесь и далее необходимо отключить проверку сертификата, если не установлен корректный сертификат TLS/SSL на сервере, которому вы доверяете.

  - В секции **Which events would you like to trigger this webhook** выберите опцию **Let me select individual events**.
  - В открывшемся списке выберите **Pull requests**.
  - Выберите опцию **Active**.
- Webhook для CD:
  - **Payload URL**: <URL, использованный в настройках шаблона потока заданий CD в контроллере>;
  - **Content type**: **application/json**;
  - **Secret**: <значение Webhook key из настроек шаблона задания CI в контроллере>;
  - **SSL verification**: **Disable**.
  - В секции **Which events would you like to trigger this webhook** выберите опцию **Let me select individual events**.
  - В открывшемся списке выберите **Pushes**.
  - Выберите опцию **Active**.

18.3.4 Проверка работоспособности

Проверке подлежат отдельно два процесса.

Проверка CI

Инициируйте процесс CI следующими действиями:

1. В репозитории GitHub внесите изменение в файл `project.sh`, так чтобы в нем оставалась команда `exit 1`, имитирующая некорректный код.
2. Нажмите кнопку **Commit changes…**.
3. Выберите опцию **Create a new branch…** и введите комментарий в поле сообщения.
4. Нажмите кнопку **Propose changes**.
5. Нажмите кнопку **Create pull request**.

Сервис GitHub выведет сообщения о начале обработке события. Эта обработка завершается сообщением об ошибке «All checks have failed, 1 failing check».

Проверьте запуск процесса CI в контроллере:

1. В панели навигации откройте *Views > Jobs*. В верху списка вы видите появление новой записи о запуске задания из шаблона CI.
2. Откройте задание и изучите результат его работы. Поскольку скрипт `project.sh` возвращает код 1, то последняя задача в сценарии из файла `ci.yml` возвращает ошибку.

Повторите весь процесс, обеспечив в скрипте `project.sh` выполнение команды `exit 0`, сообщающей об успешном выполнении программы. Отличия от предыдущей попытки:

- Процесс следует начать с редактирования последней ветки, созданной в предыдущей попытке.
- Не надо создавать еще одну ветку, а выберите ту, в которой вы внесли изменения.

В этот раз процесс CI завершается успешно. Для того, чтобы убедиться в этом, откройте в вкладку **Pull requests** и затем откройте новейший PR.

Проверка CD

Процесс CD является продолжением процесса CI. После успешного завершения CI сервис GitHub предоставит возможность начать процесс обновления ветки `master` с последующим обновлением продукта.

После появления зеленой кнопки **Merge pull request** выполните обновление ветки `master`. Для этого нажмите кнопку **Merge pull request** и затем **Confirm merge**.

На этом работа в GitHub завершена.

Проверьте запуск процесса CD в контроллере:

1. В панели навигации откройте *Views > Jobs*. В верху списка вы видите появление новой записи о запуске потока заданий из шаблона CD.
2. Откройте задание и изучите результат ее работы.
3. Используйте один из следующих способов для определения узла графа, требующего разрешения на выполнение дальнейших действий:
   - визуализатор открытого потока заданий;
   - иконка уведомлений (в виде колокольчика);
   - список согласований, который можно открыть с помощью панели навигации: *Автоматизация процессов ▸ Администрирование ▸ Согласования потоков заданий* (*Automation Execution ▸ Administration ▸ Workflow Approvals*).
4. Выберите соответствующий узел и нажмите кнопку **Согласовать** (Approve). Убедитесь, что выполнение потока заданий (процесс CD) завершено успешно.

18.4 Пример настройки GitFlic webhook

В Automation Controller есть возможность запускать задания по запросу от внешних систем, в том числе по событиям, обнаруживаемым с помощью специальных программных модулей, называемых webhook.

Важно

Для обработки запросов от webhook Automation Controller должен быть доступен по HTTPS с доверенным сертификатом.

18.4.1 Получение токена

Для получения токена выполните следующие действия:

1. Создайте токен OAuth в настройках GitFlic.
2. Скопируйте созданный токен, поскольку он понадобится позже для настройки контроллера.

18.4.2 Настройка контроллера

Для настройки контроллера выполните следующие действия:

1. В графическом интерфейсе перейдите в раздел *Полномочия*.
2. Нажмите кнопку *Добавить* (Add).
3. Заполните форму **Создать новые учетные данные** (Create New Credential):
   - **Название** (Name) – название полномочия, например, *Доступ к GitFlic*.
   - **Описание** (Description) – дополнительная информация о полномочии, например, *Полномочие создано для настройки webhook GitFlic*.
   - **Организация** (Organization) – организация, которой принадлежит полномочие, например, *Jupiter*.

     Если поле не заполнено, полномочие принадлежит всем организациям контроллера.
   - **Тип полномочия** (Credential Type) – тип полномочия.

     Выберите **Личный токен доступа к GitFlic** (GitFlic Personal Access Token).
   - **Токен** (Token) – введите токен, созданный ранее в GitFlic.
4. Откройте в режиме редактирования шаблон задания, который необходимо связать с webhook, и измените значения следующих настроек:
   1. В секции **Опции** (Options) включите флаг **Включить webhook** (Enable Webhook).
   2. В секции **Подробности о webhook** (Webhook details) настройте следующие параметры:
      - **Сервис webhook** (Webhook Service): **GitFlic**.
      - **Полномочия webhook** (Webhook Credential): *Доступ к GitFlic*.
   3. Сохраните настройки.
5. После сохранения шаблона контроллер сгенерирует параметры доступа для шаблона в следующем виде:
   - **Webhook URL**: <webhook URL>;
   - **Ключ webhook** (Webhook key): <hash code>.
6. Скопируйте сгенерированные параметры, они понадобятся для настройки webhook в GitFlic.

18.4.3 Настройка GitFlic

Для настройки GitFlic выполните следующие действия:

1. В настройках проекта в GitFlic выберите вкладку **Вебхуки**.
2. Нажмите кнопку *Создать*.
3. В полях **URL для отправки данных** и **Секрет** введите значения, полученные ранее. Прочие поля оставьте пустыми.
4. Выберите события, при наступлении которых по указанному URL будут передаваться данные, например, **Новый запрос на слияние**.

18.4.4 Проверка работоспособности

Для проверки корректности настройки webhook выполните следующие действия:

1. Создайте запрос на слияние в GitFlic.
2. Убедитесь, что в Automation Controller стартовало задание.
3. После выполнения задания откройте его и выберите вкладку **Подробности** (Details).
4. Убедитесь, что переменная `awx_webhook_status_api` имеет значение `null`.
5. Убедитесь, что в файле `/var/log/tower/tower.log` нет ошибок.

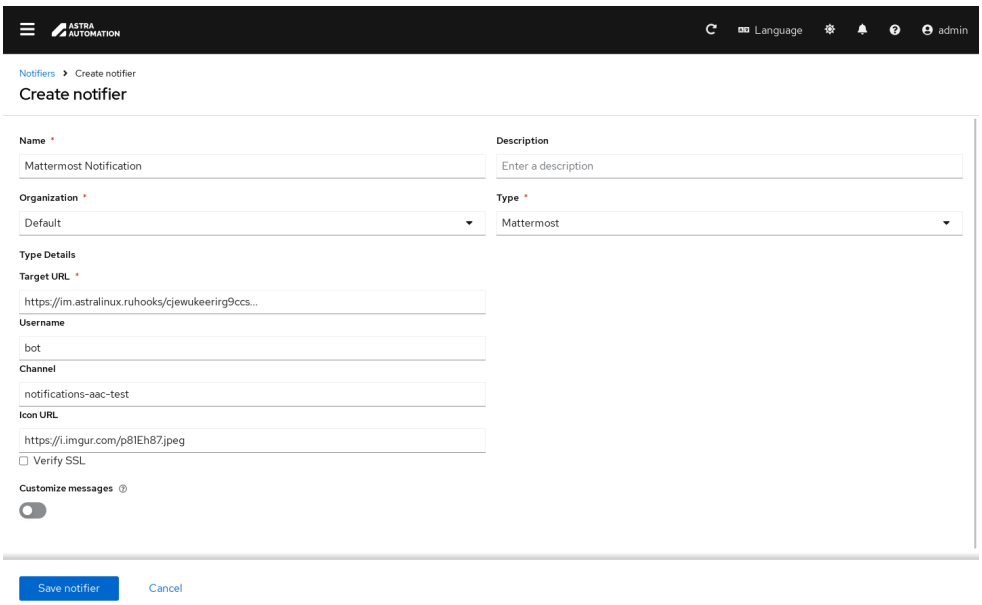
18.5 Отправка уведомлений в Mattermost

В данном документе представлена инструкция по настройке уведомлений из Automation Controller на примере Mattermost.

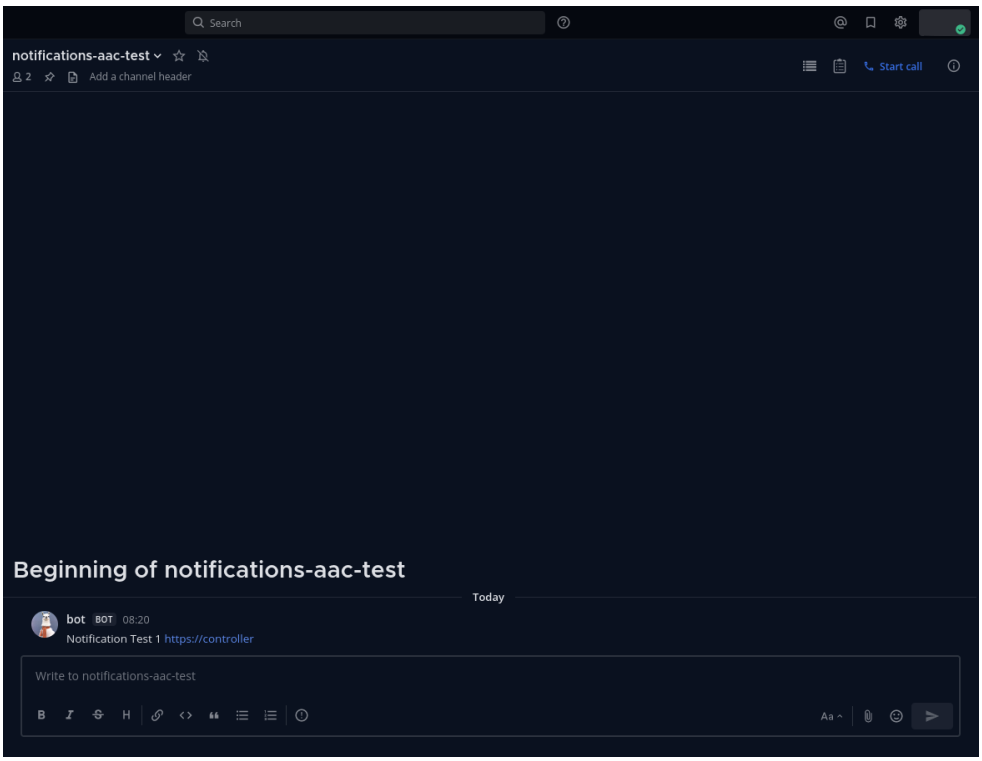
18.5.1 Создание уведомления

Для создания уведомления выполните следующие действия:

1. Перейдите в раздел *Автоматизация процессов* ▸ *Администрирование* ▸ *Источники уведомлений* (*Automation Execution* ▸ *Administration* ▸ *Notifiers*) и нажмите кнопку *Создать источник уведомлений* (Create notifier).



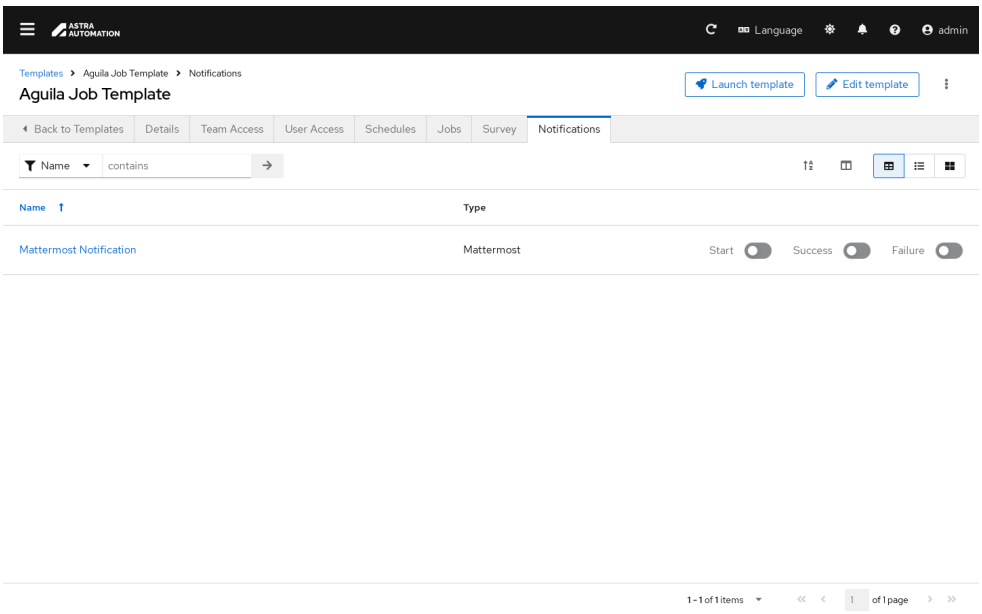
2. *Создайте* новый шаблон уведомления.
3. После сохранения откроется окно с созданным уведомлением, в котором можно проверить его работоспособность, нажав кнопку *Тестовое уведомление* (Test notifier). В канал Mattermost направляется уведомление следующего вида:



18.5.2 Добавление уведомления к шаблонам заданий и потоков заданий

Для добавления уведомлений к шаблонам заданий и потоков заданий выполните следующие действия:

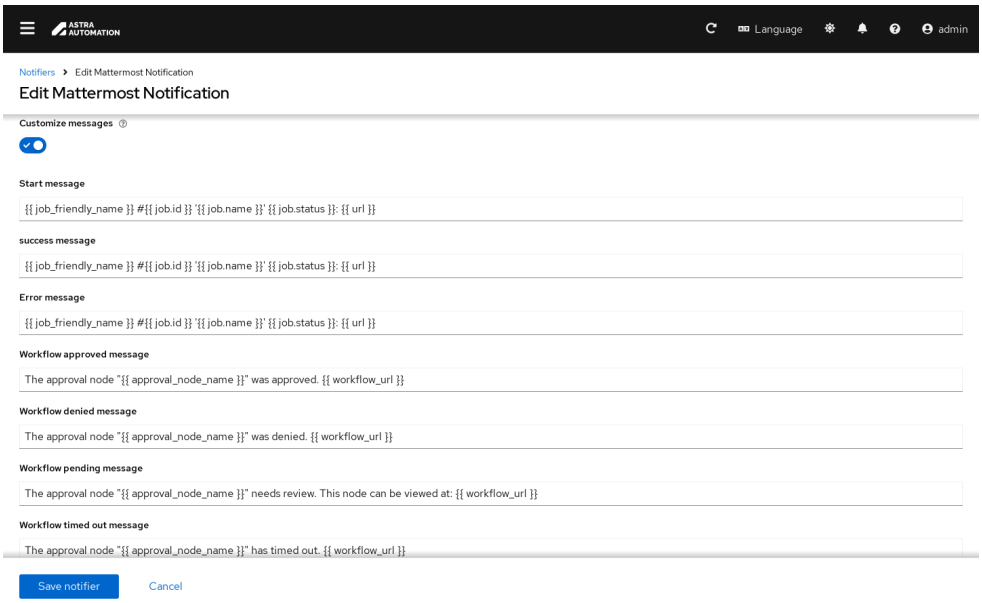
1. Перейдите в раздел *Автоматизация процессов* ▸ *Шаблоны* (*Automation Execution* ▸ *Templates*).
2. Нажмите на название шаблона к которому необходимо добавить уведомления и выберите вкладку **Уведомления** (Notifications).
3. В строке созданного уведомления включите необходимые флаги для отправки уведомления:
   - **Согласование** (Approval) – подтверждение или отмена действия, связанного с заданиями.
   - **Начать** (Start) – запуск заданий.
   - **Успешно** (Success) – успешное выполнение заданий.
   - **Сбой** (Failure) – возникновение ошибок при выполнении заданий.



18.5.3 Изменение текста сообщений

Для изменения текста сообщений в уведомлениях выполните следующие действия:

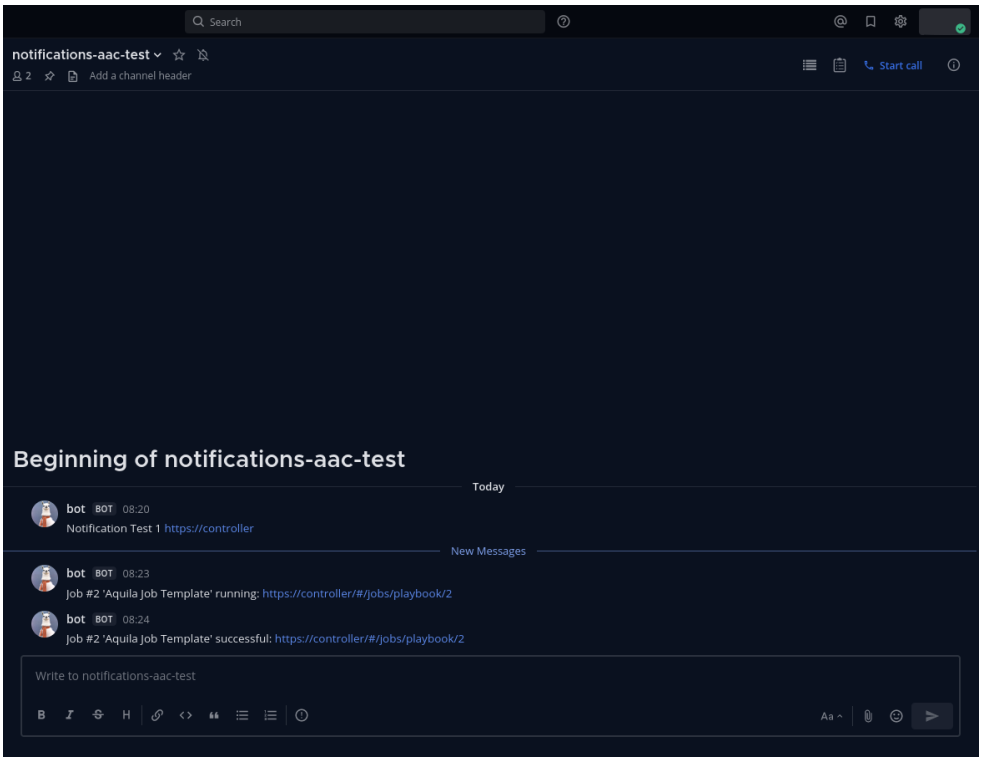
1. В настройках созданного ранее уведомления переведите переключатель **Настроить сообщения** (Customize messages) во включенное состояние:



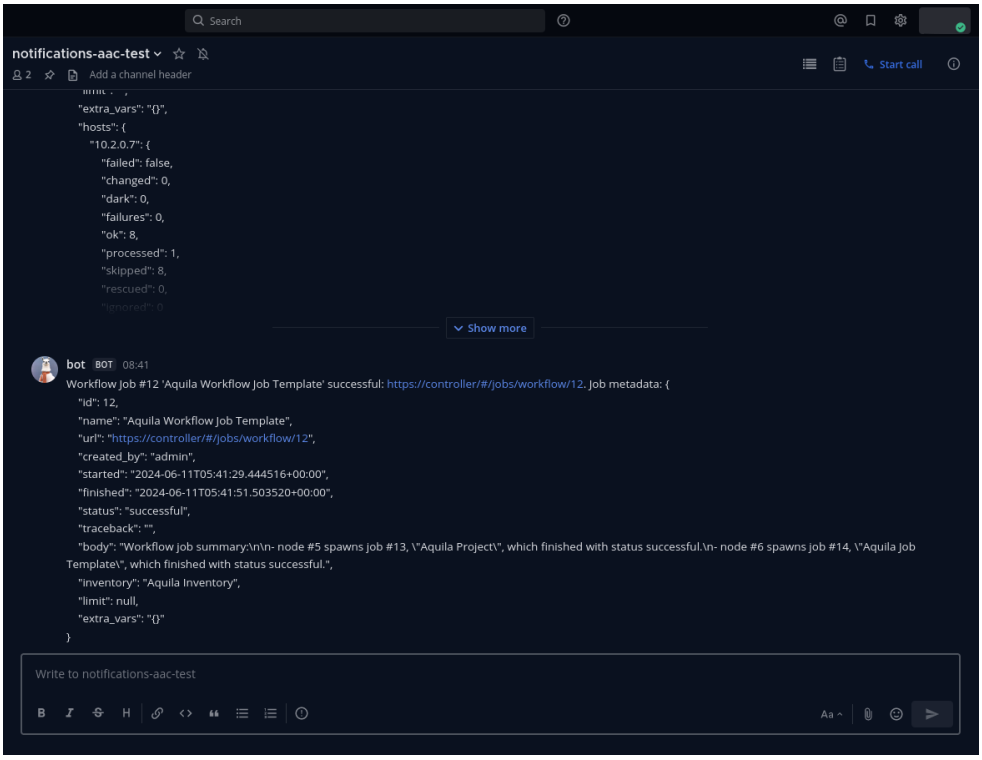
2. Выберите событие, для которого необходимо изменить сообщение.
3. Если необходимо добавить основную информацию о задании или потоке заданий, используйте переменную Ansible `{{ job_metadata }}`. Это словарь, который содержит следующие поля:
   - `inventory`;
   - `project`;
   - `playbook`;
   - `credential`;
   - `limit`;
   - `extra_vars`;
   - `hosts`;

- body (отображается только в потоке заданий).

Пример вывода для шаблона задания:



Пример вывода для потока заданий:



18.6 Настройка SSO в Automation Controller с использованием ALD Pro

Automation Controller может использовать контроллер домена ALD Pro в качестве поставщика услуг аутентификации (SSO) с применением метода LDAP.

18.6.1 Описание сценария

Процесс настройки SSO для Automation Controller с использованием ALD Pro состоит из следующих этапов:

1. Настройка домена ALD Pro.

   Параметры домена:

Параметр	Значение
Версия ALD Pro	`2.4.0`
Название домена	`domain.example`
FQDN контроллера домена	`dc.domain.example`

   В домене необходимо создать следующие группы:

   - `network_admins`;
   - `gitflic_admins`;
   - `controller_admins`.

   Пользователи этих групп при авторизации в Automation Controller будут автоматически получать необходимые привилегии.

2. Изменение настроек Automation Controller.

   В организации `Default` необходимо создать команды `Network Admins` и `GitFlic Admins` с необходимыми привилегиями. Пользователи доменных групп `network_admins` и `gitflic_admins` автоматически становятся участниками этих команд.

   Пользователи доменной группы `controller_admins` автоматически получают привилегии администратора Automation Controller.

3. Проверка работоспособности SSO.

18.6.2 Подготовка к работе

Подготовьте окружение к настройке SSO:

1. Авторизуйтесь в панели управления контроллером домена ALD Pro.
2. Чтобы Automation Controller мог получать данные для авторизации из ALD Pro, создайте учетную запись пользователя `acadmin` и добавьте ее в группу `admin`.
3. Создайте группы `network_admins`, `gitflic_admins` и `controller_admins`.
4. Добавьте в созданные группы нужных пользователей.

18.6.3 Настройка Automation Controller

Авторизуйтесь в Automation Controller от имени администратора и выполните следующие действия:

1. В организации `Default` создайте команды `Network Admins` и `GitFlic Admins`, следуя *инструкции*.
2. По своему усмотрению настройте доступ для каждой команды к различным ресурсам. Порядок настройки описан в *инструкции*.
3. На панели навигации выберите *Настройки* ▸ *Аутентификация* (*Settings* ▸ *Authentication*).
4. Нажмите на ссылку *Настройки LDAP* (LDAP Settings).
5. В окне **Подробности** (Details) нажмите кнопку *Редактировать* (Edit).
6. Заполните поля формы **Редактировать детали** (Edit Details):
   - **URI сервера LDAP** (LDAP Server URI):

```
ldap://dc.domain.example:389
```

- **Пароль привязки к LDAP** (LDAP Bind Password): пароль учетной записи, от имени которой Automation Controller будет обращаться к ALD Pro для получения данных авторизации.
- **Тип группы LDAP** (LDAP Group Type): **NestedMemberDNGroupType**.
- **Запуск LDAP с TLS** (LDAP Start TLS): если для защиты подключения к контроллеру домена используется сертификат, выданный удостоверяющим центром, переведите этот переключатель во включенное положение.
- **LDAP Bind DN**:

```
UID=acadmin,CN=users,CN=accounts,DC=domain,DC=example
```

- **Шаблон уникального наименования (DN) пользователя в LDAP** (LDAP User DN Template):

```
UID=%(user)s,CN=users,CN=accounts,DC=domain,DC=example
```

- **Групповой поиск по LDAP** (LDAP Group Search):

```
[
    "cn=groups,cn=accounts,dc=domain,dc=example",
    "SCOPE_SUBTREE",
    "(objectClass=groupOfNames)"
]
```

- **Ассоциация пользовательских атрибутов с LDAP** (LDAP User Attribute Map):

```
{
  "email": "email",
  "last_name": "sn",
  "first_name": "givenName"
}
```

- **Параметры типа группы LDAP** (LDAP Group Type Parameters):

```
{
  "name_attr": "cn",
  "member_attr": "member"
}
```

- **Пользовательские флаги LDAP по группам** (LDAP User Flags By Group):

```
{
  "is_superuser": [
    "CN=controller_admins,CN=groups,CN=accounts,DC=domain,DC=example"
  ]
}
```

- **Ассоциация организации с LDAP** (LDAP Organization Map):

```
{
  "Default": {
    "remove_admins": false,
    "remove_users": false,
    "users": [
      "CN=network_admins,CN=groups,CN=accounts,DC=domain,DC=example",
      "CN=gitflic_admins,CN=groups,CN=accounts,DC=domain,DC=example"
    ]
  }
}
```

- **Ассоциация группы с LDAP** (LDAP Team Map):

```
{
  "Network Admins": {
    "organization": "Default",
    "remove": true,
    "users": "CN=network_admins,CN=groups,CN=accounts,DC=domain,DC=example"
  },
  "GitFlic Admins": {
    "organization": "Default",
    "remove": true,
    "users": "CN=gitflic_admins,CN=groups,CN=accounts,DC=domain,DC=example"
  }
}
```

7. Нажмите кнопку *Сохранить* (Save).

18.6.4 Проверка работоспособности SSO

Чтобы проверить корректность настроек SSO, выполните следующие действия:

1. Завершите сессию администратора Automation Controller или откройте страницу авторизации в приватной вкладке.
2. На странице авторизации введите данные доменного пользователя:
   - **Имя пользователя** (Username): краткое название доменной учетной записи.

     Например, если полное название доменной учетной записи `johndow@domain.example`, введите только `johndow`.
   - **Пароль** (Password): пароль доменной учетной записи.
3. Нажмите кнопку *Вход* (Login).
4. Убедитесь, что авторизация прошла успешно и доступны ресурсы, разрешения для которых были настроены на уровне команд.

18.6.5 Особенности проекта

Обратите внимание на следующие особенности выполненного проекта:

- создание отдельного доменного пользователя для настройки SSO;
- настройка сопоставления доменных групп и команд Automation Controller;
- особенности заполнения полей при настройке параметров LDAP.

18.6.6 Заключение

В этом сценарии вы познакомились с основными шагами по настройке SSO в Automation Controller с использованием в качестве поставщика услуг единого входа контроллера домена ALD Pro. Из всей последовательности шагов важно выделить следующие действия:

- подготовка окружения;
- настройка параметров Automation Controller;
- проверка корректности работы SSO.

18.7 Защита информации

В ОС Astra Linux Special Edition встроено множество механизмов безопасности, но некоторые из них по умолчанию выключены с целью повышения быстродействия, удобства администрирования или использования ОС. Однако, при обработке определенных типов информации к информационной системе могут предъявляться различные требования по обеспечению безопасности.

Эти требования могут быть заданы сторонними регуляторами либо собственной службой безопасности. ОС Astra Linux Special Edition позволяет удовлетворить их без использования

сторонних (наложенных) *СЗИ*. Для этого необходимо выполнить настройку ОС и входящего в ее состав программного обеспечения.

Коллекция Ansible `astra.hardening` позволяет автоматизировать настройку механизмов безопасности Astra Linux Special Edition и входящего в ее состав программного обеспечения для выполнения необходимых требований. Все роли, входящие в состав коллекции, адаптированы под Astra Linux Special Edition.

18.7.1 CIS Benchmarks

Center for Internet Security (CIS)[262] – некоммерческая организация, специализирующая на повышении общего уровня кибербезопасности как в частном, так и государственном секторах.

CIS разрабатывает:

- наборы стандартных технических контрольных мер (CIS Controls[263]), которые обеспечивают конкретные и актуализированные рекомендации по безопасности ИТ-систем;
- детальные руководства по настройке безопасности (CIS Benchmarks[264]) для обширного спектра информационных технологий и систем.

Указанные меры и руководства широко приняты в отрасли как стандарты безопасности.

Руководство CIS Debian Family Linux Benchmark[265] описывает настройку параметров безопасности операционных систем, основанных на Debian Linux[266]. Это руководство проверено и адаптировано для ОС Astra Linux Special Edition, основанной на Debian Linux.

Роль Ansible `astra.hardening.cis` используется для автоматизации настройки Astra Linux Special Edition согласно рекомендаций и инструкций, приведенных в CIS Debian Family Linux Benchmark.

Особенности роли `astra.hardening.cis`:

- Правило «1.4.1 Ensure AIDE is installed» заменено правилом «1.4.1 Ensure afick is installed».

  В состав Astra Linux Special Edition не входит AIDE (Advanced Intrusion Detection System). Вместо этого используется утилита `afick` (Another File Integrity Checker).

- Правило из раздела «1.7 Mandatory Access Control» переработано таким образом, чтобы вместо системы мандатного контроля доступа AppArmor использовался встроенный в ядро ОС механизм PARSEC.

- Правило «1.9 Ensure GDM is removed or login is configured» заменено правилом «1.9 Ensure Fly DM is removed or login is configured».

  В ОС Astra Linux Special Edition вместо окружения рабочего стола GNOME используется окружение рабочего стола Fly.

- Для выполнения правила «3.1.1 Disable IPv6» рекомендуется отключать протокол IPv6 путем изменения параметров ядра с помощью `sysctl`, а не через настройки загрузчика GRUB2.

  Подробности см. в статье справочного центра IPv6: включение и выключение[267].

- Изначально в CIS Debian Family Linux Benchmark раздел «3.4 TCP Wrappers» отсутствует, однако роль `astra.hardening.cis` реализует все необходимые правила:
  - «3.4.1 Ensure TCP Wrappers is installed»;
  - «3.4.2 Ensure /etc/hosts.allow is configured»;
  - «3.4.3 Ensure /etc/hosts.deny is configured»;

[262] https://www.cisecurity.org/
[263] https://www.cisecurity.org/controls
[264] https://www.cisecurity.org/cis-benchmarks
[265] https://www.cisecurity.org/benchmark/debian_family
[266] https://www.debian.org/
[267] https://wiki.astralinux.ru/pages/viewpage.action?pageId=44892637

  - «3.4.4 Ensure permissions on /etc/hosts.allow are configured»;
  - «3.4.5 Ensure permissions on /etc/hosts.deny are configured».

- Правило «3.6.3.1.2 Ensure nftables is not installed» переработано в «3.6.3.1.2 Ensure nftables service is disabled».

  В Astra Linux Special Edition пакет `nftables` не удаляется, так как он является зависимостью пакета `iptables`.

- Правило «5.3.4 Ensure password hashing algorithm is SHA-512» заменено правилом «5.3.4 Ensure password hashing algorithm is GOST R 34.10-2012 (512)».

  Для хеширования паролей пользователей в файле `/etc/shadow` в ОС Astra Linux Special Edition вместо алгоритма SHA-512 используется 512-битный вариант алгоритма, описанного в ГОСТ Р 34.11-2012[268].

  Подробности см. в статье справочного центра Генерация хеша пароля по алгоритму ГОСТ Р 34.11-2012[269].

- Использование системы журналирования `rsyslog` в Astra Linux Special Edition x.7 и более новых версиях не рекомендуется. Поддержка пакета `rsyslog` в репозиториях Astra Linux Special Edition не осуществляется, а его установка нарушает работоспособность окружения рабочего стола Fly. Вместо `rsyslog` рекомендуется использовать систему журналирования `syslog-ng`.

  Подробности см. в статье справочного центра Журналы работы системных служб[270].

Названия переменных, управляющих применением правил CIS Debian Family Linux Benchmark, формируются по следующему шаблону:

```
cis_alse_rule_<number>
```

где <number> – номер правила, например, `6.2.1`, `3.6.1.1` или `3.6.3.3.4`.

Пример сценария, выполняющего настройку управляемых узлов с применением правил 6.2.1, 3.6.1.1 и 3.6.3.3.4:

```
- name: Configure security settings
  hosts: all
  vars:
    cis_alse_rule_6.2.1: true
    cis_alse_rule_3.6.1.1: true
    cis_alse_rule_3.6.3.3.4: true
  roles:
    - role: astra.hardening.cis
```

Полный список переменных роли `astra.hardening.cis` и их значения по умолчанию представлены в файле `hardening/roles/cis/defaults.main`.

Пример использования роли см. в документе *Применение рекомендаций CIS*.

18.7.2 DevSec Hardening Framework

DevSec Hardening Framework[271] содержит рекомендации по проверке настроек параметров безопасности ОС и различного ПО с помощью платформы для аудита и автоматического тестирования безопасности и соответствия требованиям Chef InSpec[272], а также по автоматизированной настройке этих параметров с помощью популярных систем управления конфигурацией (Ansible, Chef[273], Puppet[274]).

[268] https://protect.gost.ru/document.aspx?control=7&id=180209
[269] https://wiki.astralinux.ru/pages/viewpage.action?pageId=216540367
[270] https://wiki.astralinux.ru/pages/viewpage.action?pageId=9011231
[271] https://dev-sec.io/
[272] https://community.chef.io/tools/chef-inspec
[273] https://community.chef.io/
[274] https://www.puppet.com/community/open-source/open-source-puppet

Роль Ansible `astra.hardening.devsec` реализует настройки безопасности, описанные в следующих разделах DevSec Hardening Framework:

- DevSec Linux Security Baseline[275] v2.3.0;
- DevSec SSH Baseline[276] v2.4.0.

Особенности реализации роли `astra.hardening.devsec`:

- Упомянуто правило «package-04». Это правило пропущено в DevSec Linux Security Baseline.
- Реализовано правило «sysctl-34». При выполнении этого правила устанавливаются безопасные значения для следующих параметров ядра:
  - protected_fifos[277];
  - protected_hardlinks[278];
  - protected_regular[279];
  - protected_symlinks[280].

Названия переменных, управляющих применением правил, формируются по следующему шаблону:

```
devsec_alse_<software>_rule_<name>
```

где:

- <software> – настраиваемое ПО:
  - `linux` – Astra Linux Special Edition;
  - `ssh` – OpenSSH.
- <name> – название правила в соответствующем руководстве, но с заменой символов дефиса `-` на символы подчеркивания `_`.

Пример сценария, выполняющего настройку управляемых узлов с применением правил «sysctl-02» и «sshd-01»:

```
---
- name: DevSec security settings
  hosts: all
  vars:
    devsec_alse_linux_rule_sysctl_02: true
    devsec_alse_ssh_rule_sshd_01: true
  roles:
    - role: astra.hardening.devsec
```

Пример использования роли см. в документе *Применение рекомендаций DevSec*.

18.7.3 Требования и методические рекомендации ФСТЭК России

ОС Astra Linux Special Edition сертифицирована Федеральной службой по техническому и экспортному контролю (ФСТЭК России[281]) как *СЗИ* и реализует функции защиты информации, предусмотренные для 1 класса защиты следующими требованиями:

- Требования по безопасности информации к операционным системам (утв. приказом ФСТЭК России от 19.08.2016 г. № 119. ДСП);

[275] https://dev-sec.io/baselines/linux/
[276] https://dev-sec.io/baselines/ssh/
[277] https://docs.kernel.org/admin-guide/sysctl/fs.html#protected-fifos
[278] https://docs.kernel.org/admin-guide/sysctl/fs.html#protected-hardlinks
[279] https://docs.kernel.org/admin-guide/sysctl/fs.html#protected-regular
[280] https://docs.kernel.org/admin-guide/sysctl/fs.html#protected-symlinks
[281] https://fstec.ru/

- Требования по безопасности информации к средствам виртуализации (утв. приказом ФСТЭК России от 27.10.2022 г. № 187[282]);
- Требования по безопасности информации к средствам контейнеризации (утв. приказом ФСТЭК России от 04.07.2022 г. № 118[283]).

Кроме того, Astra Linux Special Edition реализует функции системы управления базами данных (СУБД), обеспечивающие выполнение ряда функций безопасности СУБД.

Функции защиты реализованы встроенным комплексом средств защиты информации Astra Linux Special Edition и могут применяться для реализации мер защиты информации ограниченного доступа в соответствии с требованиями следующих приказов ФСТЭК России:

- Приказ № 17 от 11 февраля 2013 г.[284] «Об утверждении требований о защите информации, не составляющей государственную тайну, содержащейся в государственных информационных системах»;
- Приказ № 21 от 18 февраля 2013 г.[285] «Об утверждении состава и содержания организационных и технических мер по обеспечению безопасности персональных данных при их обработке в информационных системах персональных данных»;
- Приказ № 31 от 14 марта 2014 г.[286] «Об утверждении требований к обеспечению защиты информации в автоматизированных системах управления производственными и технологическими процессами на критически важных объектах, потенциально опасных объектах, а также объектах, представляющих повышенную опасность для жизни и здоровья людей и для окружающей природной среды»;
- Приказ № 239 от 25 декабря 2017 г.[287] «Об утверждении требований по обеспечению безопасности значимых объектов критической информационной инфраструктуры российской федерации».

Важно

Требования вышеперечисленных приказов ФСТЭК России не регламентируют настройки средств защиты, а определяют базовый набор мер защиты, который для реализации на объектах информатизации адаптируется и уточняется в зависимости от структурно-функциональных характеристик и модели угроз.

С учетом того, что меры защиты представлены в требованиях приказов без конкретных параметров настроек, наши рекомендации по настройке операционной системы могут оказаться неподходящими для вашей информационной системы или недостаточными для предотвращения несанкционированного доступа. Решение о применении настроек остается за администратором безопасности вашей информационной системы.

В следующих нормативных документах приведены методические рекомендации ФСТЭК России по настройке систем безопасности информационных систем:

- Методический документ от 11 февраля 2014 г.[288] «Меры защиты информации в государственных информационных системах».
- Методический документ от 25 декабря 2022 г.[289] «Рекомендации по безопасной настройке операционных систем Linux».

В соответствии с требованиями ФСТЭК России в информационных системах, обрабатывающих информацию ограниченного доступа: в государственных информационных системах, информационных системах персональных данных, автоматизированных системах управления, в составе значимых объектов критической информационной инфраструктуры, –

[282] https://fstec.ru/dokumenty/vse-dokumenty/spetsialnye-normativnye-dokumenty/trebovaniya-po-bezopasnosti-informatsii-utverzh
[283] https://fstec.ru/dokumenty/vse-dokumenty/spetsialnye-normativnye-dokumenty/trebovaniya-po-bezopasnosti-informatsii-utverzh
[284] https://fstec.ru/dokumenty/vse-dokumenty/prikazy/prikaz-fstek-rossii-ot-11-fevralya-2013-g-n-17
[285] https://fstec.ru/dokumenty/vse-dokumenty/prikazy/prikaz-fstek-rossii-ot-18-fevralya-2013-g-n-21
[286] https://fstec.ru/dokumenty/vse-dokumenty/prikazy/prikaz-fstek-rossii-ot-14-marta-2014-g-n-31
[287] https://fstec.ru/dokumenty/vse-dokumenty/prikazy/prikaz-fstek-rossii-ot-25-dekabrya-2017-g-n-239
[288] https://fstec.ru/dokumenty/vse-dokumenty/spetsialnye-normativnye-dokumenty/metodicheskij-dokument-ot-11-fevralya-2014-g
[289] https://fstec.ru/dokumenty/vse-dokumenty/spetsialnye-normativnye-dokumenty/metodicheskij-dokument-ot-25-dekabrya-2022-g

должны быть реализованы меры защиты информации, состав которых определяется исходя из класса (уровня, категории) защищенности (значимости) системы и актуальной модели угроз. Соответствие системы защиты информации информационной системы требованиям по безопасности подтверждается на этапе аттестационных испытаний.

Возможности Astra Linux Special Edition для реализации мер защиты информации, предусмотренных указанными выше приказами и описание используемых встроенных средств защиты приведены в статье справочного центра Возможности реализации мер защиты информации в соответствии с приказами ФСТЭК России средствами Astra Linux Special Edition[290].

Для настройки систем безопасности в соответствии с указанными требованиями и рекомендациями можно использовать роль `astra.hardening.fstec`.

18.7.4 Payment Card Industry Data Security Standard

Стандарт Payment Card Industry Data Security Standard (PCI DSS)[291] содержит список требований, предъявляемых к информационным системам по обеспечению безопасности данных платежных карт.

Важно

Эта часть документации находится в стадии разработки.

18.7.5 Методические указания Банка России

Важно

Эта часть документации находится в стадии разработки.

Стенд

Приведенные в этом разделе руководства проверены на стендах, состоящих из узлов со следующими характеристиками:

Характеристика	Значение
Кол-во ядер CPU	≥ 2
Кол-во RAM, ГБ	≥ 4
Размер хранилища, ГБ	≥ 20
Тип ОС	Astra Linux Special Edition 1.8.2
Режим защищенности ОС	Выбирается в зависимости от типа сценария

Для подключения к стенду используется протокол SSH.

Если у вас нет подходящего стенда, вы можете создать его с помощью *Vagrant*, используя приложенный `Vagrantfile`.

Применение рекомендаций CIS

Использование роли `astra.hardening.cis` из реестра Automation Hub позволяет автоматизировать процесс настройки систем защиты ОС Astra Linux Special Edition в соответствии с рекомендациями, изложенными в руководстве «CIS Debian Family Linux Benchmark».

[290] https://wiki.astralinux.ru/pages/viewpage.action?pageId=181666117
[291] https://www.pcisecuritystandards.org/document_library/?document=pci_dss

Описание сценария

Процесс настройки систем безопасности ОС Astra Linux Special Edition с помощью роли `astra.hardening.cis` состоит из следующих этапов:

1. Подготовка проекта Ansible.

   Проект включает в себя следующие файлы:

   - `ansible.cfg` – настройки Ansible;
   - `inventory.yml` – описание инвентаря;
   - `playbooks/cis.yml` – набор сценариев с настройками управляемых узлов.

   С помощью переменных роли к ОС управляемых узлов применяются настройки, описанные в следующих разделах «CIS Debian Family Linux Benchmark»:

   - 1.8.5 Ensure remote login warning banner is configured properly

     Содержимое файла `/etc/issue.net` отображается при попытке удаленного подключения к узлу. Эта информация может быть полезна для разработчиков ПО, но также она может быть использована злоумышленником для выбора инструментальных средств, эксплуатирующих уязвимости ОС определенной версии.

     Рекомендуется отключить вывод информации о версии ОС при попытках удаленного подключения к узлу.

   - 1.8.6 Ensure local login warning banner is configured properly

     Содержимое файла `/etc/issue` отображается при попытке локального входа в систему. Эта информация может быть полезна для разработчиков ПО, но также может быть использована злоумышленником для выбора инструментальных средств, эксплуатирующих уязвимости ОС определенной версии.

     Рекомендуется отключить вывод информации о версии ОС на экране локального входа в систему.

   - 3.2.1 Ensure packet redirect sending is disabled

     Перенаправление ICMP (ICMP Redirects) используется для передачи информации о маршрутизации на другие узлы. Если узел не используется в качестве маршрутизатора, он не должен пересылать пакеты на другие узлы.

2. Запуск набора сценариев.
3. Проверка поведения ОС после изменения настроек безопасности.

Подготовка к работе

Подготовьте окружение к управлению задачами автоматизации:

1. Настройте управляемые узлы согласно *инструкции*.
2. Настройте Private Automation Hub на синхронизацию с Automation Hub.
3. Загрузите в Private Automation Hub образы среды исполнения `aa-1.2/aa-full-ee`.
4. Изучите описание[292] коллекции `astra.hardening`.
5. Подготовьте каталог для хранения файлов проекта.

Automation Controller

Если для управления задачами автоматизации используется Automation Controller, выполните следующие действия:

1. В одном из сервисов хранения репозиториев Git создайте пустой репозиторий для проекта.
2. Клонируйте этот репозиторий на локальный компьютер.

[292] https://hub.astra-automation.ru/ui/repo/validated/astra/hardening/docs/

Ansible Navigator

Если для управления задачами автоматизации используется Ansible Navigator, создайте в любом удобном месте пустой каталог, например:

```
mkdir ~/ansible-cis/
```

Проверка настроек систем защиты по умолчанию

Для проверки действующих значений параметров систем защиты выполните следующие действия:

1. Подключитесь к любому управляемому узлу по SSH.
2. Выполните команды, соответствующие выбранным правилам «CIS Debian Family Linux Benchmark»:
   - 1.8.5 Ensure remote login warning banner is configured properly

     ```
     grep -E -i "(\\v|\\r|\\m|\\s|$(grep '^ID=' /etc/os-release | cut -d= -f2 |␣
     ↪sed -e 's/"//g'))" /etc/issue.net
     ```

     Эта команда выполняет в файле `/etc/issue.net` поиск строки, позволяющей определить версию ОС. При настройках ОС по умолчанию эта команда возвращает строку:

     ```
     Astra Linux 1.8.2
     ```

   - 1.8.6 Ensure local login warning banner is configured properly

     ```
     grep -E -i "(\\v|\\r|\\m|\\s|$(grep '^ID=' /etc/os-release | cut -d= -f2 |␣
     ↪sed -e 's/"//g'))" /etc/issue
     ```

     Эта команда выполняет в файле `/etc/issue` поиск строки, позволяющей определить версию ОС. При настройках ОС по умолчанию эта команда возвращает строку:

     ```
     Astra Linux 1.8.2 \n \l
     ```

   - 3.2.1 Ensure packet redirect sending is disabled

     Для получения действующих значений параметров ядра выполните команды:

     ```
     sudo sysctl net.ipv4.conf.all.send_redirects
     sudo sysctl net.ipv4.conf.default.send_redirects
     ```

     При настройках ОС по умолчанию они возвращают следующий набор строк:

     ```
     net.ipv4.conf.all.send_redirects = 1
     net.ipv4.conf.default.send_redirects = 1
     ```

     Значение 1 означает, что соответствующая функциональность ядра включена.

     Для поиска значений настроек, хранящихся в конфигурационных файлах, выполните команды:

     ```
     grep "net\.ipv4\.conf\.all\.send_redirects" /etc/sysctl.conf /etc/sysctl.d/*
     grep "net\.ipv4\.conf\.default\.send_redirects" /etc/sysctl.conf /etc/sysctl.d/*
     ```

     При настройках ОС по умолчанию они возвращают следующий набор строк:

     ```
     /etc/sysctl.conf:#net.ipv4.conf.all.send_redirects = 0
     /etc/sysctl.d/99-sysctl.conf:#net.ipv4.conf.all.send_redirects = 0
     ```

     В конфигурационных файлах `/etc/sysctl.conf` и `/etc/systctl.d/99-sysctl.conf` присутствуют строки, отвечающие за отключение перенаправления пакетов, но они закомментированы.

3. Для отключения от управляемого узла выполните команду:

```
exit
```

Подготовка проекта Ansible

Подготовьте ресурсы, необходимые для использования Ansible.

1. Создайте файл `ansible.cfg` со следующим содержимым:

Список 5: `ansible.cfg`

```
[defaults]
host_key_checking = False
inventory = inventory.yml
```

2. Создайте файл инвентаря `inventory.yml`, например:

Список 6: `inventory.yml`

```
---
all:
  hosts:
    node1.example.com:
      ansible_host: 192.168.56.11
      ansible_user: vagrant
```

   В этом примере показаны настройки подключения к одному управляющему узлу.

3. Создайте каталог `playbooks/`, а в нем – набор сценариев `cis.yml`:

Список 7: `playbooks/cis.yml`

```
---
- name: Apply CIS requirements
  hosts: all
  become: true

  roles:
    - role: astra.hardening.cis
      vars:
        cis_alse_rule_1_8_5: true
        cis_alse_rule_1_8_6: true
        cis_alse_rule_3_2_1: true
```

   В этом файле выбор правил «CID Debian Family Linux Benchmark» выполняется через присваивание значения `true` переменным роли `astra.hardening.cis`.

4. Если для запуска заданий автоматизации используется Automation Controller, зафиксируйте сделанные в проекте изменения и опубликуйте их в репозитории Git.

Запуск сценария

Способ запуска сценария зависит от инструмента, используемого для управления задачами автоматизации.

Automation Controller

Для запуска задания автоматизации в Automation Controller выполните следующие действия:

1. Для доступа к управляемым узлам создайте необходимое количество полномочий типа *Машина* (Machine).
2. Если для доступа к репозиторию с кодом проекта требуется авторизация, создайте полномочие типа *Система управления версиями* (Source Control).

3. Создайте проект со следующими свойствами:
   - **Название**: *CIS*.
   - **Среда исполнения**: выберите среду исполнения, использующую образ `aa-full-ee`.

     Примечание

     Среда исполнения на основе образа `aa-full-ee` используется в Automation Controller по умолчанию. Вы можете добавить среду исполнения самостоятельно, следуя *инструкции*.

   - **Тип системы управления исходными данными**: **Git**.
   - **URL системы управления исходными данными**: укажите ссылку на репозиторий с кодом проекта.
   - **Полномочия на систему управления исходными данными**: если для доступа к репозиторию с кодом проекта требуется авторизация, выберите созданное ранее полномочие типа *Система управления версиями* (Source Control).
4. Создайте обычный инвентарь и добавьте в него сведения об управляемых узлах.

   Совет

   Вы можете использовать файл `inventory.yml` в качестве источника сведений об управляемых узлах.

5. Создайте шаблон задания со следующими свойствами:
   - **Тип задания**: **Выполнение**.
   - **Инвентарь**: выберите инвентарь, созданный на предыдущем шаге.
   - **Проект**: *CIS*.
   - **Playbook**: `playbooks/cis.yml`.
6. Запустите задание на основе созданного шаблона.

Ansible Navigator

При использовании Ansible Navigator для запуска сценария выполните команду:

```
ansible-navigator run playbooks/cis.yml \
   --eei private-hub.example.com/aa-1.2/aa-full-ee
```

При этом открывается псевдографический интерфейс, в котором показывается ход выполнения, например:

```
  Play name               Ok Changed Unreachable  Failed Skipped Ignored In progress ␣
↪Task count   Progress
0|Apply CIS requirements   5       0           0       0       0       0           1    ␣
↪      6   ██████

^b/PgUp page up     ^f/PgDn page down     ↑↓ scroll     esc back     [0-9] goto     ␣
↪:help help      Running
```

Для просмотра более подробной информации о ходе выполнения нажмите `0`.

Статус задания выводится в правом нижнем углу. Дождитесь перехода задания в статус `Successful`.

Проверка состояния систем защиты

Чтобы проверить состояние систем защиты после настройки, выполните следующие действия:

1. Подключитесь к любому из настроенных узлов по SSH.
2. Выполните команды, соответствующие выбранным правилам «CIS Debian Family Linux Benchmark»:
   - 1.8.5 Ensure remote login warning banner is configured properly

     ```
     grep -E -i "(\\v|\\r|\\m|\\s|$(grep '^ID=' /etc/os-release | cut -d= -f2 |␣
     ↪sed -e 's/"//g'))" /etc/issue.net
     ```

     После настройки эта команда возвращает пустую строку.
   - 1.8.6 Ensure local login warning banner is configured properly

     ```
     grep -E -i "(\\v|\\r|\\m|\\s|$(grep '^ID=' /etc/os-release | cut -d= -f2 |␣
     ↪sed -e 's/"//g'))" /etc/issue
     ```

     После настройки эта команда возвращает пустую строку.
   - 3.2.1 Ensure packet redirect sending is disabled

     Для определения значений действующих параметров ядра выполните команду:

     ```
     sudo sysctl net.ipv4.conf.all.send_redirects
     sudo sysctl net.ipv4.conf.default.send_redirects
     ```

     После настройки эти команды возвращают следующий набор строк:

     ```
     net.ipv4.conf.all.send_redirects = 0
     net.ipv4.conf.default.send_redirects = 0
     ```

     Значение параметров, равное `0`, свидетельствует о том, что пересылка пакетов выключена путем изменения параметров ядра.

     Для поиска значений настроек, хранящихся в конфигурационных файлах, выполните команды:

     ```
     grep "net\.ipv4\.conf\.all\.send_redirects" /etc/sysctl.conf /etc/sysctl.d/*
     grep "net\.ipv4\.conf\.default\.send_redirects" /etc/sysctl.conf /etc/sysctl.d/*
     ```

     После настройки эти команды возвращают следующий набор строк:

     ```
     /etc/sysctl.conf:#net.ipv4.conf.all.send_redirects = 0
     /etc/sysctl.d/999-astra.conf:net.ipv4.conf.all.send_redirects=0
     /etc/sysctl.d/99-sysctl.conf:#net.ipv4.conf.all.send_redirects = 0
     /etc/sysctl.d/999-astra.conf:net.ipv4.conf.default.send_redirects=0
     ```

     Отсюда следует, что для хранения необходимых параметров в каталоге `/etc/sysctl.d/` создан конфигурационный файл `999-astra.conf`, в который добавлены следующие строки:

     ```
     net.ipv4.conf.all.send_redirects=0
     net.ipv4.conf.default.send_redirects=0
     ```

     Содержимое файлов `/etc/sysctl.conf` и `/etc/sysctl.d/99-sysctl.conf` при этом не изменилось.
3. Для отключения выполните команду:

   ```
   exit
   ```

Особенности проекта

Обратите внимание на следующие особенности выполненного проекта:

- настройка параметров роли `astra.hardening.cis`;
- способы получения информации о параметрах систем безопасности ОС до и после настройки.

Заключение

В этом сценарии вы познакомились с основными шагами по настройке систем безопасности ОС Astra Linux Special Edition в соответствии с рекомендациями, изложенными в «CIS Debian Family Linux Benchmark». Из всей последовательности шагов важно выделить следующие действия:

- проверка состояния систем защиты ОС до и после настройки;
- создание и запуск набора сценариев Ansible, использующего коллекцию `astra.hardening`.

Применение рекомендаций DevSec

Использование роли `astra.hardening.devsec` из реестра Automation Hub позволяет автоматизировать процесс настройки систем защиты ОС Astra Linux Special Edition и входящих в ее состав клиента и сервера OpenSSH в соответствии с рекомендациями, изложенными в следующих руководствах:

- DevSec Linux Security Baseline[293];
- DevSec SSH Baseline[294].

Описание сценария

Процесс аудита и настройки систем безопасности ОС Astra Linux Special Edition и сервера OpenSSH с помощью роли `astra.hardening.devsec` состоит из следующих этапов:

1. Изучение поведения ОС и сервера OpenSSH с настройками по умолчанию.
2. Подготовка проекта Ansible.

   Проект включает в себя следующие файлы:

   - `ansible.cfg` – настройки Ansible.
   - `inventory.yml` – описание инвентаря.
   - `playbooks/devsec-audit.yml` – сценарий проверки параметров безопасности в соответствии с указанными выше руководствами DevSec. Этот сценарий выводит информацию об обнаруженных проблемах, но не устраняет их.
   - `playbooks/devsec-remediation.yml` – сценарий настройки параметров безопасности в соответствии с указанными выше руководствами DevSec. Этот сценарий изменяет настройки ОС и сервера OpenSSH в соответствии с указанными правилами.
3. Запуск набора сценариев.
4. Проверка поведения ОС и сервера OpenSSH после изменения настроек.

Подготовка к работе

Подготовьте окружение к управлению задачами автоматизации:

1. Разверните несколько управляемых узлов под управлением ОС Astra Linux Special Edition, работающей в базовом режиме защищенности («Орел»).
2. Настройте управляемые узлы согласно *инструкции*.

[293] https://dev-sec.io/baselines/linux/
[294] https://dev-sec.io/baselines/ssh/

3. Изучите описание[295] коллекции `astra.hardening`.
4. На своей рабочей станции подготовьте каталог для хранения файлов проекта, например:

```
mkdir ~/astra-devsec/
```

Проверка настроек систем защиты по умолчанию

Полная проверка поведения ОС Astra Linux Special Edition 1.8 и сервера OpenSSH с настройками систем безопасности по умолчанию занимает длительное время. В демонстрационных целях в этом руководстве проверяется только часть из них.

1. На любом управляемом узле авторизуйтесь с привилегиями администратора.
2. Установите пакеты `telnet` и `tftpd-hpa`.
   1. Обновите список доступных пакетов:

      ```
      sudo apt update
      ```

   2. Установите пакеты:

      ```
      sudo apt install telnet tftpd-hpa --yes
      ```

   Это ПО не удовлетворяет актуальным требованиям к безопасности, и его установка выполняется только в демонстрационных целях.

   В рекомендациях DevSec это правила `package-02 | Do not install Telnet server` и `package-06 | Do not install tftp server`. Они требуют отсутствия ПО Telnet и TFTP на узле.
3. Проверьте поддержку протокола IPv6 на уровне ядра ОС:

   ```
   sudo sysctl --all | grep net.ipv6.conf.all.disable_ipv6
   ```

   Значение `0` указывает на то, что поддержка IPv6 включена.

   В рекомендациях DevSec это правило `sysctl-18 | Disable IPv6 if it is needed`. Оно требует отключения протокола IPv6, если он не используется.
4. Получите список алгоритмов шифрования, с которыми работает сервер OpenSSH:

   ```
   sudo sshd -T | grep ciphers
   ```

   В Astra Linux Special Edition 1.8.2.UU1 эта настройка по умолчанию имеет следующее значение:

   ```
   ciphers grasshopper-ctr128,chacha20-poly1305@openssh.com,aes128-ctr,aes192-ctr,
   ↪aes256-ctr,aes128-gcm@openssh.com,aes256-gcm@openssh.com,grasshopper-cbc
   ```

   В рекомендациях DevSec это правило `sshd-01 | Check for secure ssh ciphers`. Оно требует отключения в настройках сервера OpenSSH использования алгоритмов, считающихся небезопасными.
5. Проверьте состояние перенаправления X-сервера в настройках сервера OpenSSH:

   ```
   sudo sshd -T | grep x11forwarding
   ```

   Значение yes указывает на то, что перенаправление включено.

   В рекомендациях DevSec это правило `sshd-42 | Disable X11Forwarding`. Оно требует отключения перенаправления X-сервера в настройках сервера OpenSSH.

[295] https://hub.astra-automation.ru/ui/repo/validated/astra/hardening/docs/

Подготовка проекта Ansible

На управляющем узле подготовьте ресурсы, необходимые для использования Ansible.

1. Создайте файл `ansible.cfg` со следующим содержимым:

Список 8: `ansible.cfg`

```
[defaults]
host_key_checking = False
inventory = inventory.yml
```

2. Создайте файл инвентаря `inventory.yml`, например:

Список 9: `inventory.yml`

```
---
all:
  hosts:
    node1.example.com:
      ansible_host: 192.168.0.101
    node2.example.com:
      ansible_host: 192.168.0.102
    node3.example.com:
      ansible_host: 192.168.0.103
    # ...
  vars:
    ansible_user: astra
```

3. Создайте каталог `playbooks/`.
4. Создайте в каталоге `playbooks/` файл сценария `devsec-audit.yml`:

Список 10: `playbooks/devsec-audit.yml`

```
---
- name: Audit system with astra.hardening.devsec role
  hosts: all
  become: true
  collections:
    - astra.hardening
  roles:
    - role: devsec
      vars:
        devsec_action: audit
        devsec_selected_rules:
          - package-02
          - package-06
          - sysctl-18
          - sshd-01
          - sshd-42
```

Сценарий автоматизации содержит задачу, которая проверяет действующие настройки ОС и сервера OpenSSH на соответствие перечисленным выше правилам DevSec.

5. Создайте в каталоге `playbooks` файл сценария `devsec-remediation.yml`:

Список 11: `playbooks/devsec-remediation.yml`

```
---
- name: Fix security issues with astra.hardening.devsec role
  hosts: all
  become: true
  collections:
    - astra.hardening
  roles:
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  - role: devsec
    vars:
      devsec_action: remediation
      devsec_selected_rules:
        - package-02
        - package-06
        - sysctl-18
        - sshd-01
        - sshd-42
```

Сценарий автоматизации содержит задачу, которая приводит настройки ОС и сервера OpenSSH в соответствие требованиям перечисленных выше правил DevSec.

6. Если для запуска заданий автоматизации используется Automation Controller, зафиксируйте сделанные в проекте изменения и опубликуйте их в репозитории Git.

Запуск сценариев

Способ запуска сценариев зависит от инструмента, используемого для управления задачами автоматизации.

Automation Controller

1. Для доступа к управляемым узлам создайте полномочие типа *Машина* (Machine).
2. Если для доступа к репозиторию с кодом проекта требуется авторизация, создайте полномочие типа *Система управления версиями* (Source Control).
3. Создайте проект со следующими свойствами:
   - **Название**: *DevSec*.
   - **Среда исполнения**: выберите среду исполнения, использующую образ `aa-full-ee`.

     Примечание

     Среда исполнения на основе образа `aa-full-ee` используется в Automation Controller по умолчанию. Вы можете добавить среду исполнения самостоятельно, следуя *инструкции*.

   - **Тип системы управления исходными данными**: **Git**.
   - **URL системы управления исходными данными**: укажите ссылку на репозиторий с кодом проекта.
   - **Полномочия на систему управления исходными данными**: если для доступа к репозиторию с кодом проекта требуется авторизация, выберите созданное ранее полномочие типа *Система управления версиями* (Source Control).
4. Опишите инвентарь, состоящий из развернутых ранее управляемых узлов.

   Совет

   Вы можете использовать файл `inventory.yml` в качестве источника сведений об управляемых узлах.

5. Создайте шаблон задания со следующими свойствами:
   - **Тип задания**: **Выполнение**.
   - **Инвентарь**: выберите созданный ранее инвентарь.
   - **Проект**: *DevSec*.

- **Playbook**: `playbooks/devsec-audit.yml`.

6. Запустите задание на основе созданного шаблона.
7. Проверьте вывод задания. Обратите внимание на найденные проблемы безопасности.
8. Для устранения обнаруженных проблем безопасности создайте шаблон задания со следующими свойствами:
   - **Тип задания**: **Выполнение**.
   - **Инвентарь**: выберите созданный ранее инвентарь.
   - **Проект**: *DevSec*.
   - **Playbook**: `playbooks/devsec-remediation.yml`.
9. Запустите задание на основе созданного шаблона.

Ansible Navigator

При использовании Ansible Navigator для запуска сценария аудита безопасности выполните команду:

```
ansible-navigator run playbooks/devsec-audit.yml \
   --eei private-hub.example.com/aa-1.2/aa-full-ee
```

При этом открывается псевдографический интерфейс, в котором показывается ход выполнения задания, например:

```
  Play name                                 Ok Changed Unreachable Failed Skipped Ignored␣
↪In progress Task count  Progress
0|Prepare system to astra.hardening.devsec  2       0           0      0       0       0␣
↪          0          2  Complete
1|Audit system with astra.hardening.devsec 40       0           0      0       8       0␣
↪          0         48  Complete
```

Статус задания выводится в правом нижнем углу. Дождитесь перехода задания в статус `Successful`.

Для просмотра результатов аудита перейдите к результатам выполнения задания:

1. Введите `:1` и нажмите `Enter`, чтобы перейти в данные задания «Audit system with astra.hardening.devsec role».
2. Введите двоеточие и номер строки с задачей «Show DevSec compliance report» и нажмите `Enter`.

   Ansible Navigator отображает сведения о задаче в формате YAML (часть вывода опущена с целью сокращения):

```
---
duration: 0.024929
end: "2025-07-30T11:50:47.073370+00:00"
event_loop: null
host: node1.example.com
play: Audit system with astra.hardening.devsec role
play_pattern: all
playbook: /home/***/project/playbooks/devsec-audit.yml
remote_addr: node1.example.com
res:
  _ansible_no_log: false
  _ansible_verbose_always: true
  changed: false
  msg:
    package-02:
      compliant: true
      rationale: ""
    package-06:
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
      compliant: false
      rationale:
        "The following TFTP server packages are installed in the target system:
        tftpd-hpa."
    sshd-01:
      compliant: false
      rationale: |-
        In the selected SSH server configuration file /etc/ssh/sshd_config
        the following parameters are absent or different from the values specified:␣
↪Ciphers.
    sshd-42:
      compliant: false
      rationale: |-
        In the selected SSH server configuration file /etc/ssh/sshd_config
        the following parameters are absent or different from the values specified:␣
↪X11Forwarding.
    sysctl-18:
      compliant: false
      rationale:
        "The following Linux kernel parameters are different from the values
        specified: net.ipv6.conf.all.disable_ipv6."
```

Из этого вывода следует, что указанные выше правила не выполняются, а значит, безопасность системы находится под угрозой.

Для устранения обнаруженных проблем безопасности запустите сценарий `playbooks/devsec-remediation.yml`:

```
ansible-navigator run playbooks/devsec-remediation.yml \
   --eei private-hub.example.com/aa-1.2/aa-full-ee
```

Проверка состояния систем защиты

Чтобы проверить состояние систем защиты после настройки, выполните следующие действия на любом из настроенных узлов:

1. Проверьте наличие `telnet` и `tftpd-hpa` в списке установленных пакетов:

   ```
   dpkg -l | grep -e "telnet\|tftpd-hpa"
   ```

   Если вывод пустой, значит, пакеты удалены из системы.

2. Проверьте разрешение на использование протокола IPv6:

   ```
   sudo sysctl --all | grep net.ipv6.conf.all.disable_ipv6
   ```

   Значение 1 указывает на то, что поддержка IPv6 выключена.

3. Получите список алгоритмов шифрования, с которыми работает сервер OpenSSH:

   ```
   sudo sshd -T | grep ciphers
   ```

   Текущая настройка отличается от того, что было первоначально:

   ```
   Ciphers chacha20-poly1305@openssh.com,aes256-gcm@openssh.com,aes128-gcm@openssh.com,
   ↪aes256-ctr,aes192-ctr,aes128-ctr
   ```

4. Проверьте состояние перенаправления X-сервера в настройках сервера OpenSSH:

   ```
   sudo sshd -T | grep x11forwarding
   ```

   Значение `no` указывает на то, что перенаправление выключено.

Особенности проекта

Обратите внимание на следующие особенности выполненного проекта:

- настройка параметров роли `astra.hardening.devsec`;
- правила, исключенные из руководств DevSec, но возвращенные в роль `astra.hardening.devsec`;
- способы получения информации о параметрах систем безопасности ОС и сервера OpenSSH до и после настройки;
- использование роли `astra.hardening.devsec` в двух режимах: аудит и устранение проблем безопасности.

Заключение

В этом сценарии вы познакомились с основными шагами по настройке систем безопасности ОС Astra Linux Special Edition в соответствии с требованиями, изложенными в руководствах DevSec Linux Security Baseline и DevSec SSH Baseline.

Из всей последовательности шагов важно выделить следующие действия:

- проверка состояния систем защиты ОС и сервера OpenSSH до и после настройки;
- создание и запуск набора сценариев Ansible, использующих роль `astra.hardening.devsec`.

Выполнение требований и рекомендаций ФСТЭК России

Использование роли `astra.hardening.fstec` из реестра Automation Hub позволяет автоматизировать процесс аудита и настройки систем защиты ОС Astra Linux Special Edition в соответствии с требованиями, изложенными в следующих приказах ФСТЭК России:

- Приказ № 17 от 11 февраля 2013 г.[296] «Об утверждении требований по защите информации, не составляющей государственную тайну, содержащейся в государственных информационных системах».
- Приказ № 21 от 18 февраля 2013 г.[297] «Об утверждении состава и содержания организационных и технических мер по обеспечению безопасности персональных данных при их обработке в информационных системах персональных данных».
- Приказ № 239 от 25 декабря 2017 г.[298] «Об утверждении требований по обеспечению безопасности значимых объектов критической информационной инфраструктуры Российской Федерации».
- Приказ № 31 от 14 марта 2014 г.[299] «Об утверждении требований к обеспечению защиты информации в автоматизированных системах управления производственными и технологическими процессами на критически важных объектах, потенциально опасных объектах, а также объектах, представляющих повышенную опасность для жизни и здоровья людей и для окружающей природной среды».

Описание сценария

Процесс аудита и настройки систем безопасности ОС Astra Linux Special Edition с помощью роли `astra.hardening.fstec` состоит из следующих этапов:

1. Изучение поведения ОС с настройками по умолчанию.
2. Подготовка проекта Ansible.

   Проект включает в себя следующие файлы:

   - `ansible.cfg` – настройки Ansible;

[296] https://fstec.ru/dokumenty/vse-dokumenty/prikazy/prikaz-fstek-rossii-ot-11-fevralya-2013-g-n-17
[297] https://fstec.ru/dokumenty/vse-dokumenty/prikazy/prikaz-fstek-rossii-ot-18-fevralya-2013-g-n-21
[298] https://fstec.ru/dokumenty/vse-dokumenty/prikazy/prikaz-fstek-rossii-ot-25-dekabrya-2017-g-n-239
[299] https://fstec.ru/dokumenty/vse-dokumenty/prikazy/prikaz-fstek-rossii-ot-14-marta-2014-g-n-31

- `inventory.yml` – описание инвентаря;
- `playbooks/fstec.yml` – сценарий настройки параметров безопасности ОС в соответствии с требованиями приказа ФСТЭК России.

3. Запуск набора сценариев.
4. Проверка поведения ОС после изменения настроек безопасности.

Подготовка к работе

Подготовьте окружение к управлению задачами автоматизации:

1. Разверните управляемый узел (например, виртуальную машину) под управлением ОС Astra Linux Special Edition, работающей в базовом (Орел) режиме защищенности.

> **Важно**
>
> Для демонстрации настройки узла в соответствии с требованиями ФСТЭК России на нем обязательно наличие графического интерфейса.

2. Настройте управляемый узел согласно *инструкции*.
3. Изучите описание[300] коллекции `astra.hardening`.
4. На своей рабочей станции подготовьте каталог для хранения файлов проекта, например:

```
mkdir ~/astra-fstec/
```

Проверка настроек систем защиты по умолчанию

Полная проверка поведения ОС Astra Linux Special Edition 1.8 с настройками систем безопасности по умолчанию занимает длительное время. В демонстрационных целях проверьте только часть из них:

1. На управляемом узле авторизуйтесь с привилегиями администратора.
2. Запустите утилиту «Монитор безопасности»: *Пуск* ▸ *Параметры* ▸ *Параметры системы*.
3. В дереве настроек выберите *Безопасность* ▸ *Монитор безопасности*.
4. Нажмите на ссылку **Нажмите сюда, для запроса аутентификации**.
5. Введите пароль.
6. Убедитесь, что в списке **Подсистемы безопасности** отмечены красным следующие строки:
   - блокировка системных команд (df, chattr, arp, ip, и т.д.);
   - блокировка консоли для пользователей;
   - блокировка интерпретаторов;
   - блокировка макросов;
   - запрет установки бита исполнения;
   - межсетевой экран UFW — UFW не найден;
   - системные ограничения ulimits;
   - блокировка выключения/перезагрузки ПК для пользователей;
   - запрет монтирования носителей непривилегированным пользователям;
   - блокировка интерпретатора bash;

[300] https://hub.astra-automation.ru/ui/repo/validated/astra/hardening/docs/

- режим работы файловой системы ОС - только чтение;
- ввод пароля для sudo;
- блокировка доступа пользователя root по протоколу SSH;
- защита SSH fail2ban;
- графический киоск.

7. В дереве настроек выберите *Безопасность* ▸ *Пользователи и группы* ▸ *Пользователи*.
8. Создайте непривилегированного пользователя.
9. Завершите работу с утилитой «Параметры системы».
10. Завершите активную сессию и авторизуйтесь в системе как непривилегированный пользователь.
11. Запустите утилиту «Терминал»: *Пуск* ▸ *Программы* ▸ *Инструменты* ▸ *Терминал*.
12. Убедитесь, что пользователю разрешено использование системной утилиты `ip`:

```
ip a
```

Когда использование утилиты разрешено, она выводит в терминал сведения об активных сетевых подключениях.

Подготовка проекта Ansible

На управляющем узле подготовьте ресурсы, необходимые для использования Ansible.

1. Создайте файл `ansible.cfg` со следующим содержимым:

Список 12: `ansible.cfg`

```
[defaults]
host_key_checking = False
inventory = inventory.yml
```

2. Создайте файл инвентаря `inventory.yml`, например:

Список 13: `inventory.yml`

```
---
all:
  hosts:
    node1.example.com:
      ansible_host: 192.168.56.11
      ansible_user: vagrant
```

3. Создайте каталог `playbooks/`, а в нем – файл сценариев `fstec.yml`:

Список 14: `playbooks/fstec.yml`

```
---
- name: Prepare for FSTEC hardening
  hosts: all
  become: true
  gather_facts: false
  tasks:
    - name: Install astra-admin-events
      ansible.builtin.apt:
        name:
          - astra-admin-events
        state: present

- name: Apply FSTEC requirements
  hosts: all
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
become: true
gather_facts: false
vars:
  fstec_order: 21
  fstec_security_level: sl1
  disabled_rules:
    - user_mac_4
    - user_mic_4
    - limit_memory_1
    - limit_memory_2
    - limit_safepolicy_1
    - limit_safepolicy_2
collections:
  - astra.hardening
roles:
  - role: fstec
    become: true
    vars:
      fstec_action: remediation
```

В этом файле описаны два сценария:

- `Prepare for FSTEC hardening` – подготовка ОС к применению роли `astra.hardening.fstec`.

  Сценарий содержит задачу, устанавливающую пакет `astra-admin-events`, необходимый для работы роли.

- `Apply FSTEC requirements` – приведение параметров систем безопасности в соответствие требованиям приказа ФСТЭК России № 21 от 18 февраля 2013 г. к информационным системам, защищенных по первому уровню защищенности, за исключением настроек, перечисленных в значении переменной `disabled_rules`.

4. Если для запуска заданий автоматизации используется Automation Controller, зафиксируйте сделанные в проекте изменения и опубликуйте их в репозитории Git.

Запуск сценария

Способ запуска сценария зависит от инструмента, используемого для управления задачами автоматизации.

Automation Controller

Для запуска задания автоматизации в Automation Controller выполните следующие действия:

1. Для доступа к управляемому узлу создайте полномочие типа *Машина* (Machine).
2. Если для доступа к репозиторию с кодом проекта требуется авторизация, создайте полномочие типа *Система управления версиями* (Source Control).
3. Создайте проект со следующими свойствами:
   - **Название**: *ФСТЭК*.
   - **Среда исполнения**: выберите среду исполнения, использующую образ `aa-full-ee`.

     Примечание

     Среда исполнения на основе образа `aa-full-ee` используется в Automation Controller по умолчанию. Вы можете добавить среду исполнения самостоятельно, следуя *инструкции*.

   - **Тип системы управления исходными данными**: **Git**.

- **URL системы управления исходными данными**: укажите ссылку на репозиторий с кодом проекта.
- **Полномочия на систему управления исходными данными**: если для доступа к репозиторию с кодом проекта требуется авторизация, выберите созданное ранее полномочие типа *Система управления версиями* (Source Control).

4. Создайте обычный инвентарь и добавьте в него сведения об управляемом узле.

> **Совет**
>
> Вы можете использовать файл `inventory.yml` в качестве источника сведений об управляемом узле.

5. Создайте шаблон задания со следующими свойствами:
   - **Тип задания**: **Выполнение**.
   - **Инвентарь**: выберите инвентарь, созданный на предыдущем шаге.
   - **Проект**: *ФСТЭК*.
   - **Playbook**: `playbooks/fstec.yml`.
6. Запустите задание на основе созданного шаблона.

Ansible Navigator

При использовании Ansible Navigator для запуска сценария выполните команду:

```
ansible-navigator run playbooks/fstec.yml \
   --eei private-hub.example.com/aa-1.2/aa-full-ee
```

При этом открывается псевдографический интерфейс, в котором показывается ход выполнения, например:

```
  Play name                 Ok Changed Unreachable  Failed Skipped Ignored In progress ␣
↪Task count   Progress
0|Apply FSTEC requirements   5       0           0       0       0       0           1  ␣
↪        6   ▉▉▉▉▉

^b/PgUp page up     ^f/PgDn page down     ↑↓ scroll     esc back     [0-9] goto     ␣
↪:help help      Running
```

Для просмотра более подробной информации о ходе выполнения нажмите `0`.

Статус задания выводится в правом нижнем углу. Дождитесь перехода задания в статус `Successful`.

Проверка состояния систем защиты

Чтобы проверить состояние систем защиты после настройки, выполните следующие действия на настроенном узле:

1. Перезагрузите компьютер.
2. Авторизуйтесь от имени учетной записи с привилегиями администратора.
3. Запустите утилиту «Монитор безопасности»: *Пуск* ▸ *Параметры* ▸ *Параметры системы*.
4. В дереве настроек выберите *Безопасность* ▸ *Монитор безопасности*.
5. Нажмите на ссылку **Нажмите сюда, для запроса аутентификации**.
6. Введите пароль.
7. Убедитесь, что в списке **Подсистемы безопасности** уменьшилось количество строк, отмеченных красным.

8. Завершите работу с утилитой «Параметры системы».
9. Завершите активную сессию и авторизуйтесь в системе с учетными данными непривилегированного пользователя.
10. Запустите утилиту «Терминал»: *Пуск* ▸ *Программы* ▸ *Инструменты* ▸ *Терминал*.
11. Убедитесь, что теперь пользователю запрещено использование системной утилиты `ip`:

    ```
    ip a
    ```

    Когда использование утилиты запрещено, она выводит в терминал следующее сообщение:

    ```
    bash: /usr/bin/ip: Отказано в доступе
    ```

Особенности проекта

Обратите внимание на следующие особенности выполненного проекта:

- настройка параметров роли `astra.hardening.fstec`;
- правила, необходимые для соответствия требованиям ФСТЭК России, но исключенные из роли `astra.hardening.fstec`;
- способы получения информации о параметрах систем безопасности ОС до и после настройки.

Заключение

В этом сценарии вы познакомились с основными шагами по настройке систем безопасности ОС Astra Linux Special Edition в соответствии с требованиями, изложенными в приказах ФСТЭК России:

- Приказ № 17 от 11 февраля 2013 г.
- Приказ № 21 от 18 февраля 2013 г.
- Приказ № 239 от 25 декабря 2017 г.
- Приказ № 31 от 14 марта 2014 г.

Из всей последовательности шагов важно выделить следующие действия:

- проверка состояния систем защиты ОС до и после настройки;
- создание и запуск набора сценариев Ansible, использующего роль `astra.hardening.fstec`.

Выполнение требований PCI DSS

Важно

Эта часть документации находится в стадии разработки.

Выполнение требований Банка России

Важно

Эта часть документации находится в стадии разработки.

ГЛАВА 19

Дополнительные материалы

Раздел содержит дополнительные материалы по продуктам, сервисам и инструментам для работы с Astra Automation.

19.1 Приложение 1. Устанавливаемые продукты

Важно

Эта часть документации находится в стадии разработки.

Astra Automation содержит ряд коллекций, направленных на автоматизацию развертывания продуктов ПАО Группа Астра и управления ими.

19.1.1 ALD Pro

ALD Pro – это набор сетевых служб сервера Astra Linux для организации централизованного управления ИТ-инфраструктурой. Развертывание домена ALD Pro с использованием коллекции `astra.ald_pro` из реестра Automation Hub позволяет автоматизировать большую часть рутинных операций.

Структура

Домен ALD Pro может состоять из контроллера домена, его реплик, узлов с дополнительными службами домена и клиентов домена. Для настройки каждого компонента существует отдельная роль. Также в состав коллекции входит служебная роль `astra.ald_pro.common`, которая используется другими ролями.

Далее приводится пример использования следующих ролей:

- `astra.ald_pro.controller`;
- `astra.ald_pro.replica`;
- `astra.ald_pro.client`.

Параметры настройки

Для настройки параметров домена необходимо присвоить соответствующие значения переменным роли.

Роль astra.ald_pro.controller

Роль astra.ald_pro.controller используется для настройки контроллера домена ALD Pro.

Переменные роли:

- aldpro_global_forwarders – IP-адреса DNS-серверов, используемых для разрешения имен за пределами домена.

  Значение по умолчанию:

  - 8.8.8.8;
  - 8.8.4.4.

- aldpro_server_minimal_memory_mb – минимальное количество мегабайт оперативной памяти, необходимое для развертывания служб контроллера домена ALD Pro.

  Развертывание контроллера на узле запускается только в том случае, когда на нем установлено количество оперативной памяти большее или равное указанному.

  Значение по умолчанию: 4096 (4 ГБ).

- aldpro_enable_syncer – установка модуля синхронизации aldpro-syncer.

  Этот модуль необходим для использования расширенных функций интеграции с доменом Microsoft Active Directory.

  Значение по умолчанию: true.

- aldpro_enable_gc – установка модуля глобального каталога aldpro-gc.

  Этот модуль необходим, если используется топология из контроллера домена и нескольких реплик. Службы, предоставляемые этим модулем, выполняют синхронизацию данных пользователей между контроллером домена и его репликами.

  Значение по умолчанию: true.

- aldpro_start_upgrade – флаг, указывающий, что выполняется обновление версии ALD Pro, а не развертывание контроллера домена.

  Значение по умолчанию: false.

Роль astra.ald_pro.replica

Роль astra.ald_pro.replica используется для настройки реплик контроллера домена ALD Pro.

Переменные роли:

- aldpro_pdc_ip – IP-адрес контроллера домена, например, 192.168.56.10.
- aldpro_pdc_name – краткое имя контроллера домена.

  Например, если FQDN контроллера домена – dc1.moscow.example.com, то краткое имя – dc1.

- aldpro_dc_site_name – название сайта ALD Pro.

  Значение по умолчанию: Головной офис.

Развертывание реплики происходит в два этапа:

1. развертывание служб клиента домена;
2. преобразование клиента домена в реплику.

Если службы клиента домена уже настроены, этот этап развертывания реплики можно пропустить. Для этого необходимо выполнить следующие действия:

1. Добавить в сценарий для соответствующего задания тег replica, например:

```
---
- name: Enrolling ALD Pro replica
  hosts: replicas
  become: true
  tags: replica
  roles:
    - astra.ald_pro.replica
```

2. Запустить набор сценариев с тегом replica.

```
ansible-navigator run playbook.yml -i inventory.yml -t replica -m stdout
```

Роль astra.ald_pro.client

Роль astra.ald_pro.client используется для настройки клиентов домена ALD Pro.

Переменные роли:

- aldpro_pdc_ip – IP-адрес контроллера домена.

  Значение по умолчанию: 192.168.56.10.

- aldpro_pdc_name – имя контроллера домена.

  Значение по умолчанию: dc01.

Роль astra.ald_pro.common

Переменные роли astra.ald_pro.common:

- aldpro_domain – название домена.
- aldpro_admin_password – пароль администратора домена.

  Длина пароля – не менее 8 символов.

- aldpro_version – используемая версия ALD Pro.

  По умолчанию используется самая новая из доступных версий ALD Pro.

- aldpro_repo – список ссылок на репозитории, которые следует использовать для установки пакетов ALD Pro.

  Например, для развертывания ALD Pro версии 2.4.1 эта ссылка может иметь следующий вид:

```
deb https://dl.astralinux.ru/aldpro/frozen/01/2.4.1 1.7_x86-64 main base
```

- aldpro_start_upgrade – если значение этой переменной равно true, производится обновление ALD Pro до более новой версии. При значении false происходит развертывание контроллера и реплик, ввод клиентов в домен.
- aldpro_main_repo_upgrade – список ссылок на репозитории, которые следует использовать для обновления пакетов ALD Pro.

  Например, для обновления ALD Pro до версии 2.4.1 эта ссылка может иметь следующий вид:

```
deb https://dl.astralinux.ru/aldpro/frozen/01/2.4.1 1.7_x86-64 main base
```

- aldpro_upgrade_hosts_list – список узлов, на которых необходимо обновить версию ALD Pro.

Обязательные переменные

При использовании ролей обязательны для заполнения следующие переменные:

- `astra.ald_pro.controller`:
  - `aldpro_admin_password`;
  - `aldpro_domain`;
  - `aldpro_main_repo_upgrade`;
  - `aldpro_repo`;
- `astra.ald_pro.replica`:
  - `aldpro_admin_password`;
  - `aldpro_dc_site_name`;
  - `aldpro_domain`;
  - `aldpro_main_repo_upgrade`;
  - `aldpro_pdc_ip`;
  - `aldpro_pdc_name`;
  - `aldpro_repo`;
- `astra.ald_pro.client`:
  - `aldpro_admin_password`;
  - `aldpro_domain`;
  - `aldpro_main_repo_upgrade`;
  - `aldpro_pdc_ip`;
  - `aldpro_pdc_name`;
  - `aldpro_repo`.

Последовательность применения

Для применения коллекции `astra.ald_pro` выполните следующие действия:

1. Подготовьте узлы с необходимыми техническими характеристиками.

   Требования к техническим характеристикам узлов и методики расчета приведены в документе РДЦП.10101-01 95 01 «Программный комплекс «ALD Pro». Руководство администратора. Инструкция по развертыванию и обновлению»[301], п. 1.4 «Рекомендации по планированию ресурсов службы каталога».

2. Настройте управляемые узлы согласно *инструкции*.

3. Создайте каталог проекта, например:

   ```
   mkdir ald-pro-enrolling
   ```

   Все упомянутые ниже файлы следует создавать в каталоге проекта, если только явно не указано иное.

4. Создайте файл с настройками Ansible `ansible.cfg`:

   ```
   [defaults]
   ansible_ssh_private_key_file = /path/to/private/ssh/key
   host_key_checking = False
   interpreter_python = /usr/bin/python3
   inventory = inventory.ini
   ```

[301] https://www.aldpro.ru/docs/manual_admin.pdf

5. Создайте файл инвентаря `inventory.ini` и добавьте в него сведения об управляемых узлах, например:

```
[all]
dc1       ansible_host=192.168.56.12
dc2       ansible_host=192.168.56.13
client1   ansible_host=192.168.56.14
client2   ansible_host=192.168.56.15

[all:vars]
ansible_user='vagrant'

[clients]
client1
client2
```

6. Создайте файл с переменными ролей `variables.yml`, например:

```
---
aldpro_admin_password: "aldpropass123456"
aldpro_dc_site_name: "Центральный офис"
aldpro_domain: "domain.example.com"
aldpro_enable_gc: true
aldpro_enable_syncer: true
aldpro_global_forwarders:
  - "8.8.8.8"
  - "1.1.1.1"
aldpro_pdc_ip: "192.168.56.12"
aldpro_pdc_name: "dc1"
aldpro_repo:
  - "deb https://dl.astralinux.ru/aldpro/frozen/01/2.4.1 1.7_x86-64 main base"
aldpro_start_upgrade: false
```

7. Создайте файл сценария `playbook.yml` и добавьте в него соответствующие роли для узлов домена, например:

```
---
- name: Enrolling ALD Pro domain controller
  hosts: dc1
  become: true
  vars_files:
    - variables.yml
  roles:
    - astra.ald_pro.controller

- name: Enrolling ALD Pro domain replica
  hosts: dc2
  become: true
  vars_files:
    - variables.yml
  roles:
    - astra.ald_pro.replica

- name: Enrolling ALD Pro clients
  hosts: clients
  become: true
  vars_files:
    - variables.yml
  roles:
    - astra.ald_pro.client
```

8. Выполните сценарий с использованием образа среды исполнения `aa-full-ee`:

```
ansible-navigator run playbook.yml \
  --inventory inventory.ini \
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
--eei hub.astra-automation.ru/aa-1.2/aa-full-ee:latest \
--inventory inventory.ini
```

19.1.2 RuBackup

RuBackup – клиент-серверная система резервного копирования. Развертывание RuBackup с использованием коллекции `astra.rubackup` из реестра Automation Hub позволяет автоматизировать большую часть рутинных операций.

Структура

Система резервного копирования RuBackup может состоять из основного сервера резервного копирования, резервного сервера, медиа-серверов и клиентов.

Существуют роли для настройки следующих компонентов RuBackup:

- `astra.rubackup.rubackup_server` – основной сервер резервного копирования;
- `astra.rubackup.rubackup_media_server` – медиасерверы;
- `astra.rubackup.rubackup_client` – клиенты.

Также в состав коллекции входит роль `astra.rubackup.rubackup_common`, однако, ее непосредственное использование не допускается. Это служебная роль, которая используется другими ролями.

Для работы RuBackup необходим сервер PostgreSQL. Для его развертывания и настройки можно использовать коллекцию `astra.postgresql` из реестра Automation Hub.

Параметры настройки

Для настройки параметров RuBackup необходимо присвоить соответствующие значения переменным роли.

Роль `astra.rubackup.rubackup_server`

Роль `astra.rubackup.rubackup_server` используется для настройки основного сервера резервного копирования.

Переменные роли:

- `rubackup_def_pool` – путь к пулу хранилища по умолчанию.
- `rubackup_server_pgsql_user` – название учетной записи пользователя-администратора PostgreSQL.
- `rubackup_server_psql_fqdn` – *FQDN* сервера PostgreSQL.
- `rubackup_server_psql_password` – пароль учетной записи пользователя-администратора PostgreSQL.
- `rubackup_server_psql_password_rubackup` – пароль учетной записи пользователя PostgreSQL, от имени которого RuBackup подключается к PostgreSQL.

Важно

Требования к паролю:

- длина не менее 12 символов;
- содержит хотя бы по одному символу следующих типов:
  - символ верхнего регистра;
  - цифра;
  - специальный символ.

- rubackup_server_users – учетные данные пользователей PostgreSQL, от имени которых RuBackup подключается к СУБД.

Роль astra.rubackup.rubackup_media_server

Роль astra.rubackup.rubackup_media_server используется для настройки медиа-серверов.

Переменные роли:

- rubackup_media_server_db_host – FQDN сервера PostgreSQL.
- rubackup_media_server_db_pass – пароль пользователя-администратора сервера PostgreSQL.
- rubackup_pri_fqdn – FQDN основного сервера резервного копирования.

Роль astra.rubackup.rubackup_client

Роль astra.rubackup.rubackup_client используется для настройки клиентов.

Переменные роли:

- rubackup_pri_fqdn – FQDN основного сервера резервного копирования.
- rubackup_servers_group_name – название группы, узлы которой необходимо зарегистрировать как клиенты.
- rubackup_users – список названий учетных записей пользователей PostgreSQL.

Обязательные переменные

При использовании ролей обязательные для заполнения следующие переменные:

- astra.rubackup.rubackup_server:
  - rubackup_packages_version;
  - rubackup_ram;
  - rubackup_server_psql_fqdn;
  - rubackup_server_psql_password_rubackup;
- astra.rubackup.rubackup_media_server:
  - rubackup_media_server_db_pass;
  - rubackup_packages_version;
  - rubackup_pri_fqdn;
  - rubackup_ram;
- astra.rubackup.rubackup_client:
  - rubackup_packages_version;
  - rubackup_pri_fqdn;
  - rubackup_ram;
  - rubackup_servers_group_name.

Последовательность применения

Для применения указанной коллекции выполните следующие действия:

1. Разверните сервер PostgreSQL любым удобным способом.
2. Подготовьте узлы с необходимыми техническими характеристиками.
3. Подготовьте узлы к настройке с помощью Ansible, следуя *инструкции*.
4. Создайте каталог проекта, например:

```
mkdir rubackup-enrolling
```

Все упомянутые ниже файлы следует создавать в каталоге проекта, если только явно не указано иное.

5. Создайте файл с настройками Ansible `ansible.cfg`:

```
[defaults]
collections_path = ./collections
host_key_checking = False
interpreter_python = /usr/bin/python3
inventory = inventory.ini
gathering = smart
timeout = 60

[ssh_connection]
pipelining = True
retries = 10
```

6. Создайте файл инвентаря `inventory.ini` и добавьте в него сведения об управляемых узлах, например:

```
[all]
server.example.com         ansible_host=192.168.56.101
media-server.example.com   ansible_host=192.168.56.102

[all:vars]
ansible_user='vagrant'
ansible_ssh_private_key='./ssh/key'
```

7. Создайте файл с переменными `variables.yml`, например:

```
---
# Settings for astra.rubackup roles
rubackup_user: "rubackup"
rubackup_user_password: "p@ssword!1"
rubackup_default_repository: >-
  deb https://dl.astralinux.ru/rubackup/repository-deb-main/ astra_1.7 stable
rubackup_pri_fqdn: "server.example.com"
rubackup_users:
  - rubackup
rubackup_media_server_db_pass: "SecretPassw@rd1"
rubackup_server_psql_password_rubackup: "SecretPassw@rd1"
rubackup_server_psql_fqdn: "server.example.com"
rubackup_server: "server.example.com"

postgresql_users:
  - name: "{{ rubackup_server_pgsql_user }}"
    password: "{{ rubackup_server_psql_password }}"
    state: present
postgresql_hba_entries:
  - type: local
    database: all
    user: "{{ rubackup_server_pgsql_user }}"
    auth_method: peer
  - type: local
    database: all
    user: all
    auth_method: peer
  - type: host
    database: all
    user: all
    address: "127.0.0.1/32"
    auth_method: md5
  - type: host
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
    database: all
    user: all
    address: "192.168.56.31/32"
    auth_method: md5
  - type: host
    database: all
    user: all
    address: "192.168.56.32/32"
    auth_method: md5
  - type: host
    database: all
    user: all
    address: "192.168.56.33/32"
    auth_method: md5
rubackup_server_users:
  - rubackup
```

8. Создайте файл зависимостей Ansible `requirements.yml`, в который добавьте ссылку на загрузку коллекции `astra.ald_pro`, например:

```
---
collections:
  - name: astra.postgresql
    version: 2.2.3

  - name: astra.rubackup
    version: 0.6.0
```

9. Установите зависимости:

```
ansible-navigator exec -- ansible-galaxy install -r requirements.yml
```

10. Создайте файл сценария `playbook.yml` и используйте в нем роли для всех узлов домена, например:

```
---
# Create rubackup user and join it to the rubackup group
- hosts: server.example.com
  become: true
  tasks:
    - name: Create RuBackup system user (on Server)
      ansible.builtin.user:
        name: "{{ rubackup_user }}"
        password: "{{ rubackup_user_password | password_hash('sha512') }}"
        state: present

  vars_files:
    - variables.yml

# SERVER
- hosts: server.example.com
  become: true
  gather_facts: true
  roles:
    - astra.postgresql.postgresql
    - astra.rubackup.rubackup_server

  vars_files:
    - variables.yml

# MEDIA SERVER
- hosts: media-server.example.com
  become: true # may be not necessary
  roles:
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
    - astra.rubackup.rubackup_media_server
  vars_files:
    - variables.yml
```

11. Запустите выполнение сценария.

19.2 Приложение 2. Облачные сервисы

19.2.1 ПК СВ «Брест»

Astra Automation предоставляет множество коллекций и модулей для автоматизации установки и настройки продуктов ПАО Группа Астра в ПК СВ «Брест»[302]. Перечень ПО, которое можно установить в этом комплексе средств виртуализации, представлен в разделе *Приложение 1. Устанавливаемые продукты*.

Ресурсная модель

В рамках Astra Automation чаще всего используются следующие ресурсы ПК СВ «Брест»:

- виртуальные сети;
- образы дисков;
- шаблоны виртуальных машин;
- экземпляры виртуальных машин.

Все создаваемые ресурсы размещаются внутри кластера ПК СВ «Брест».

Способы управления

В Astra Automation используются следующие способы управления ресурсами ПК СВ «Брест»:

- Веб-интерфейс.

  Этот способ удобен для получения информации о существующих ресурсах ПК СВ «Брест» и управления ими вручную.

- CLI.

  Этот способ удобен для создания сценариев автоматизации, однако, требует доступа к фронтальной машине от имени учетной записи администратора в сессии с высоким уровнем целостности.

Необходимые привилегии

Для использования ПК СВ «Брест» при работе с Astra Automation необходимо иметь привилегии, позволяющие управлять следующими объектами:

- Виртуальные сети.
- Образы дисков.
- Шаблоны виртуальных машин.
- Экземпляры виртуальных машин.

Необходимые ресурсы

Для использования Astra Automation совместно с ПК СВ «Брест» необходимы:

- URI точки доступа (endpoint) к API;
- образы Astra Linux Special Edition;
- шаблоны ВМ;

[302] https://astra.ru/software-services/application-software-astra-group/brest/

- идентификатор виртуальной сети;
- идентификатор группы ВМ;
- токен входа;
- ключи SSH для доступа к стендам.

Пошаговые инструкции по получению данных и созданию необходимых ресурсов приведены далее.

Настройка ПК СВ «Брест»

Для работы с ресурсами ПК СВ «Брест» необходимо подготовить следующие данные:

1. URI точки доступа к API.
2. Образы Astra Linux Special Edition.
3. Шаблоны ВМ.
4. Идентификатор виртуальной сети.
5. Токен входа.
6. Ключи SSH для доступа к создаваемым стендам.

Все описанные ниже действия следует выполнять в веб-интерфейсе ПК СВ «Брест».

Предполагается, что комплекс средств виртуализации настроен согласно инструкций:

- РДЦП.10001-02 95 01-1 Программный комплекс «Средства виртуализации «БРЕСТ». Руководство администратора. Часть 1»
- РДЦП.10001-02 95 01-2 Программный комплекс «Средства виртуализации «БРЕСТ». Руководство администратора. Часть 2»

Определение точки доступа к API

URI точки доступа к API формируется следующим образом:

```
https://<brest>:8443/RPC2/
```

где <brest> – FQDN или IP-адрес фронтальной машины ПК СВ «Брест».

Загрузка установочных образов в хранилище ПК СВ «Брест»

Если у вас нет подготовленных образов Astra Linux Special Edition, выполните следующие действия для загрузки в ПК СВ «Брест» установочных образов Astra Linux Special Edition:

1. Из Личного кабинета[303] загрузите на фронтальную машину установочные образы Astra Linux Special Edition в формате ISO.

   Примечание

   Образы доступны для загрузки при наличии действующей лицензии на продукт.

2. На панели навигации ПК СВ «Брест» выберите *Хранилище* ▸ *Образы*.
3. Нажмите кнопку + и в выпадающем меню выберите **Создать**.
4. Заполните форму **Образ**:
   - **Название**: произвольное название образа.

     Допускается использование букв латинского алфавита, цифр и дефиса.
   - **Тип**: **CD-ROM только для чтения**.

[303] https://lk.astra.ru

- **Расположение образа**:
  a. Выберите значение **Закачать**.
  b. Нажмите кнопку *Browse* и укажите путь к файлу образа.

5. Нажмите кнопку *Создать*.
6. Дождитесь загрузки образа в хранилище. Загрузка считается выполненной, когда статус образа принимает значение `ГОТОВО`.

Создание образа диска

Образ диска необходим для создания шаблона ВМ. Чтобы создать образ диска, выполните следующие действия:

1. На панели навигации выберите *Хранилище* ▸ *Образы*.
2. Нажмите кнопку + и в выпадающем меню выберите **Создать**.
3. Заполните форму **Образ**:
   a. **Название**: произвольное название образа.

      Допускается использование букв латинского алфавита, цифр и дефиса.
   b. **Тип**: **Общий блок данных хранилища**.
   c. **Этот образ является постоянным**: **Да**.
   d. **Расположение образа**: **Пустой образ диска**.
   e. **Размер**: размер диска по умолчанию.

      Рекомендуемое значение – не менее 20 ГБ.
4. Нажмите кнопку *Создать*.

Дождитесь создания образа. Создание считается завершенным, когда статус образа принимает значение `ГОТОВО`.

Создание шаблона ВМ

Чтобы создать шаблон ВМ, выполните следующие действия:

1. На панели навигации выберите *Шаблоны* ▸ *ВМ*.
2. Нажмите кнопку + и в выпадающем меню выберите **Создать**.
3. Заполните форму **Создать шаблон ВМ**:
   a. Заполните вкладку **Общие**:
      - **Название**: произвольное название шаблона ВМ.

        Допускается использование букв латинского алфавита, цифр и дефиса.
      - **Гипервизор**: `KVM`.
      - **Память**: объем памяти, доступный создаваемым ВМ.

        Рекомендуемое значение – не менее 2 ГБ.
      - **Физический CPU**: укажите значение не ниже `0.5`.
      - **Виртуальный CPU**: количество ядер, доступных ВМ.

        Рекомендуемое значение – не менее 2.
   b. Заполните вкладку **Хранилище**:
      - **ДИСК0**:
        1. Выберите значение **Образ**.
        2. Из списка образов выберите созданный ранее образ диска типа «Блок данных».

- В списке дисков нажмите кнопку +.
- **ДИСК1**:
  1. Выберите значение **Образ**.
  2. Из списка образов выберите образ установочного диска Astra Linux Special Edition.

4. Нажмите кнопку *Создать*.

Определение идентификатора виртуальной сети

По умолчанию при развертывании ПК СВ «Брест» создается виртуальная сеть `virtnetwork` с идентификатором `0`. Если для Astra Automation планируется использовать другую виртуальную сеть, для получения ее идентификатора на панели навигации выберите *Сеть* ▸ *Вирт. сети*. Идентификаторы виртуальных сетей указаны в колонке **ID**.

Создание токена входа

Чтобы сформировать токен входа, выполните следующие действия:

1. Нажмите кнопку с именем активного пользователя и в выпадающем меню выберите **Настройки**.
2. Выберите вкладку **Аутентификация**.
3. Нажмите кнопку *Управление токенами входа*.
4. В форме **Токен входа** заполните поле **Истечение, в секундах**, в котором укажите срок действия токена.
5. Нажмите кнопку *Получить новый токен*.
6. Значение из колонки **Токен** сохраните в надежном месте.
7. Закройте форму **Токен входа**.

Настройка SSH

ПК СВ «Брест» позволяет подключаться к созданным ВМ по протоколам RDP и VNC, однако, для использования Astra Automation требуется доступ к ВМ по SSH. Чтобы настроить его, выполните следующие действия:

1. Нажмите кнопку с именем активного пользователя и в выпадающем меню выберите **Настройки**.
2. Выберите вкладку **Аутентификация**.
3. В разделе **Публичный SSH ключ** нажмите кнопку редактирования.
4. В открывшемся поле введите содержимое публичного ключа, *используемого для доступа к стендам*.
5. Нажмите кнопку редактирования еще раз.

19.2.2 Cloud.ru

Astra Automation позволяет развертывать продукты ПАО Группа Астра в Cloud.ru. Перечень ПО, поддерживающего развертывание в Cloud.ru, представлен в документе *Приложение 1. Устанавливаемые продукты*.

Ресурсная модель

В рамках Astra Automation чаще всего используются следующие ресурсы Cloud.ru:

- виртуальные машины;
- сети;

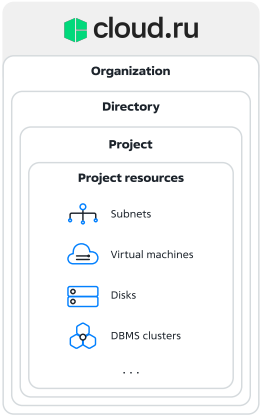
- подсети.

Все создаваемые ресурсы размещаются внутри проектов. Каждый проект размещается в определенном каталоге. Каждый каталог принадлежит определенной организации.

Примечание

При необходимости можно создать дополнительные проекты.

Упрощенная структура Cloud.ru показана на схеме:



Способы управления

В Astra Automation для управления ресурсами Cloud.ru следует использовать личный кабинет. Он доступен по адресу https://console.cloud.ru/ и предоставляет веб-интерфейс для управления ресурсами облака. Этот способ удобен для получения информации о существующих ресурсах Cloud.ru и управления ими вручную.

Необходимые привилегии

Для использования Cloud.ru при работе с Astra Automation необходимо иметь роли, назначенные на организацию или проект и обеспечивающие выполнение следующих операций:

- Создание сервисных аккаунтов и назначение им ролей.
- Создание ключей доступа.

  Ключи доступа используются для управления ресурсами Cloud.ru от имени сервисного аккаунта.
- Управление сетями и подсетями.
- Управление виртуальными машинами.
- Создание объектного хранилища S3, загрузка и выгрузка объектов из хранилища.
- Импорт ключей SSH.

Предупреждение

Если в организации или проекте не существует сервисный аккаунт с необходимыми ролями, для его создания и назначения ему ролей потребуется учетная запись администратора или пользователя, входящего в группу `admin`.

Необходимые ресурсы

Для использования Astra Automation совместно с Cloud.ru необходимы:

- Права на управление ресурсами Cloud.ru от имени любой из указанных учетных записей:

  - собственная учетная запись;
  - учетная запись организации;
  - сервисный аккаунт[304].
- Организация и ее идентификатор.
- Каталог и его идентификатор.
- Проект и его идентификатор.

  В этом проекте размещаются ВМ, используемые для развертывания необходимого ПО.

Пошаговые инструкции по получению данных и созданию необходимых ресурсов приведены далее.

Настройка сервисного аккаунта

Для работы с ресурсами Cloud.ru можно использовать учетную запись обычного пользователя, однако, для средств автоматизации лучше подходит сервисный аккаунт.

Для подготовки сервисного аккаунта Cloud.ru к работе с Astra Automation выполните следующие действия:

1. Убедитесь, что сервисный аккаунт существует и ему назначены необходимые роли.
2. Назначьте созданному сервисному аккаунту необходимые роли.
3. Создайте ключ доступа к ресурсам проекта.

Проверка наличия необходимых ролей

Для управления ресурсами сервисному аккаунту должны быть назначены роли, дающие необходимые привилегии.

Примечание

Роль Администратор дает привилегии на создание, изменение и удаление любых типов ресурсов в проекте, однако, в большинстве случае столь широкие полномочия избыточны. Для более тонкой настройки ознакомьтесь со списком стандартных ролей[305], предоставляемых Cloud.ru, и назначьте сервисному аккаунту только те роли, которые действительно необходимы.

Если сервисный аккаунт уже существует, для проверки наличия у него необходимых ролей используйте Личный кабинет пользователя Cloud.ru.

Создание сервисного аккаунта

Если сервисный аккаунт не существует, создайте его согласно инструкции[306].

Назначение ролей сервисному аккаунту

Чтобы назначить сервисному аккаунту роль на проект, воспользуйтесь инструкцией[307].

[304] https://cloud.ru/docs/console/ug/topics/guides__service_accounts.html
[305] https://cloud.ru/docs/iam/ug/topics/overview__roles-and-policies-list.html
[306] https://cloud.ru/docs/console/ug/topics/guides__service_accounts_create.html
[307] https://cloud.ru/docs/console/ug/topics/guides__employees_edit.html

Создание ключа доступа

Чтобы создать для сервисного аккаунта ключ доступа, воспользуйтесь инструкцией[308].

19.2.3 Yandex Cloud

Astra Automation позволяет развертывать продукты ПАО Группа Астра в Yandex Cloud. Перечень ПО, поддерживающего развертывание в Yandex Cloud, представлен в разделе *Приложение 1. Устанавливаемые продукты*.

Ресурсная модель

В рамках Astra Automation чаще всего используются следующие ресурсы Yandex Cloud:

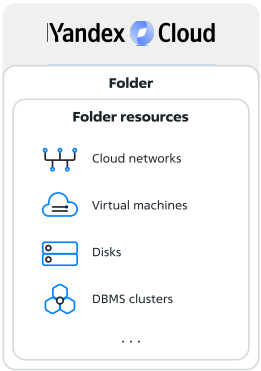
- облачные сети и подсети (Virtual Private Cloud)[309];
- виртуальные машины (Cloud Compute)[310];
- объектное хранилище S3 (Object Storage)[311].

Все создаваемые ресурсы размещаются внутри облачных каталогов. Каждый облачный каталог является частью облака. При создании облака в нем автоматически создается облачный каталог с именем `default`.

Примечание

При необходимости можно создать дополнительные облачные каталоги.

Упрощенная структура Yandex Cloud показана на схеме:



Способы управления

В Astra Automation используются следующие способы управления ресурсами Yandex Cloud:

- Консоль управления.

  Консоль управления доступна по адресу https://console.yandex.cloud/ru/ и предоставляет веб-интерфейс для управления ресурсами облака. Этот способ удобен для получения информации о существующих ресурсах Yandex Cloud и управления ими вручную.

- Клиент командной строки.

  YC CLI позволяет управлять облачными ресурсами с помощью утилиты командной строки. Этот способ удобен при использовании сценариев оболочки командной строки.

  Все дальнейшие инструкции предполагают использование YC CLI для выполнения необходимых настроек.

[308] https://cloud.ru/docs/console/ug/topics/guides__service_accounts_key.html
[309] https://yandex.cloud/ru/docs/vpc/concepts/network/
[310] https://yandex.cloud/ru/docs/compute/concepts/vm/
[311] https://yandex.cloud/ru/docs/storage/

Необходимые привилегии

Для использования Yandex Cloud при работе с Astra Automation необходимо иметь роль editor[312] на облачный каталог, либо роли, обеспечивающие выполнение следующих операций:

- Создание сервисных аккаунтов и назначение им ролей.

  Эта привилегия необходима, если сервисные аккаунты с нужными ролями на каталог не существуют.
- Создание авторизованных ключей доступа к ресурсам Yandex Cloud и объектным хранилищам S3.
- Управление облачными сетями и подсетями.
- Управление виртуальными машинами Yandex Compute Cloud.
- Создание объектного хранилища S3 (Yandex Object Storage), загрузка и выгрузка объектов.

Предупреждение

Если в облачном каталоге не существует сервисный аккаунт с необходимыми ролями, для его создания и назначения ролей потребуется роль admin[313].

Необходимые ресурсы

Для использования Astra Automation совместно с Yandex Cloud необходимы:

- Права на управление ресурсами Yandex Cloud от имени любой из указанных учетных записей:
  - собственная учетная запись;
  - учетная запись организации;
  - сервисный аккаунт[314].
- Облако и его идентификатор.
- Облачный каталог и его идентификатор. В этом каталоге размещаются ВМ, используемые для развертывания необходимого ПО.
- Сервисный аккаунт Yandex Cloud. Этот аккаунт позволяет средствам Astra Automation иметь доступ к Yandex Cloud.
- Файл формата JSON, содержащий авторизованные ключи. Используется при управлении облачными ресурсами с использованием сервисного аккаунта.

Пошаговые инструкции по получению данных и созданию необходимых ресурсов приведены далее.

Настройка интерфейса командной строки (YC CLI)

Платформа Yandex Cloud предоставляет возможность управления облачными ресурсами с помощью *YC CLI*. Основу клиентской части составляет утилита yc. Все дальнейшие инструкции по использованию Yandex Cloud совместно с Astra Automation предполагают использование этого инструмента для работы с облаком.

[312] https://yandex.cloud/ru/docs/iam/concepts/access-control/roles#editor
[313] https://yandex.cloud/ru/docs/iam/concepts/access-control/roles#admin
[314] https://yandex.cloud/ru/docs/iam/concepts/users/service-accounts

Установка и настройка интерфейса командной строки YC CLI

Детальная информация о начале работы в командной строке приведена в документации Yandex Cloud[315].

Чтобы начать пользоваться YC CLI, выполните следующие действия:

1. Установите клиентскую часть:

```
curl -sSL https://storage.yandexcloud.net/yandexcloud-yc/install.sh | bash
```

   Эта команда выполняет следующие действия:

   - создает каталог `yandex-cloud/` в домашнем каталоге пользователя;
   - дополняет файл инициализации `.bashrc` вызовом скриптов настройки клиента – `~/yandex-cloud/path.bash.inc` и `~/yandex-cloud/completion.bash.inc`.

2. Перезапустите терминал или выполните команду:

```
source ~/.bashrc
```

3. Создайте профиль пользователя для работы с облачным каталогом.

   Примечание

   Профиль определяет набор параметров окружения, в котором пользователь работает в каждый конкретный момент.

   При использовании федеративного метода идентификации на базе SAML выполните команду:

```
yc init --federation-id=<federation_ID>
```

   где <federation_ID> – идентификатор федерации[316] для вашей организации или подразделения.

   Необходимо выбрать – использовать существующий каталог или создать новый.

4. (Опционально) Настройте зону доступности по умолчанию.

5. Проверьте полученные настройки с помощью команды:

```
yc config list
```

   В терминал выводятся идентификаторы федерации, облака и каталога, используемых в текущий момент.

6. Проверьте список профилей:

```
yc config profile list
```

   В терминал выводится список существующих профилей YC CLI. Активный профиль будет отмечен меткой `ACTIVE`.

Управление профилями

Дополнительные профили YC CLI могут понадобиться, например:

- для работы с разными облаками и каталогами Yandex Cloud;
- для управления ресурсами одного и того же каталога от имени разных пользователей, в том числе сервисных аккаунтов.

Чтобы создать дополнительный профиль YC CLI, выполните следующие действия:

[315] https://yandex.cloud/ru/docs/cli/
[316] https://yandex.cloud/ru/docs/organization/concepts/add-federation#saml-authentication

1. Получите идентификаторы нужного облака и облачного каталога любым удобным способом. Чтобы получить сведения об облаке и каталоге, к которым привязан активный профиль YC CLI, выполните команду:

   ```
   yc config profile get <profile>
   ```

   В терминал выводится следующая информация о профиле:

   - `cloud-id` – идентификатор облака;
   - `folder-id` – идентификатор облачного каталога;
   - `compute-default-zone` – зона доступности по умолчанию (если была указана при настройке профиля).

2. Выполните команду создания нового профиля:

   ```
   yc config profile create <profile_name>
   ```

   Профиль автоматически активируется после создания.

3. Привяжите профиль к нужному облаку и каталогу:

   ```
   yc config set cloud-id <cloud_ID>
   yc config set folder-id <folder_ID>
   ```

   где <cloud_ID> и <folder_ID> – полученные ранее идентификаторы облака и облачного каталога соответственно.

Совет

Полный перечень команд YC CLI приведен в документации Yandex Cloud[317].

Примеры использования

Получение списка доступных профилей:

```
yc config profile list
```

Переключение на другой профиль:

```
yc config profile activate <profile>
```

Назначение зоны доступности по умолчанию:

```
yc config set compute-default-zone <zone>
```

где <zone> – название зоны доступности, например:

```
yc config set compute-default-zone ru-central1-a
```

Подробности о зонах доступности см. в документации Yandex Cloud[318].

Настройка сервисного аккаунта

Для работы с ресурсами Yandex Cloud можно использовать учетную запись обычного пользователя, однако, для средств автоматизации лучше подходит сервисный аккаунт.

Для подготовки сервисного аккаунта Yandex Cloud к работе с Astra Automation выполните следующие действия:

1. Если клиент командной строки Yandex Cloud не настроен, следуйте инструкциям из секции *Установка и настройка интерфейса командной строки YC CLI*.

[317] https://yandex.cloud/ru/docs/cli/cli-ref/
[318] https://yandex.cloud/ru/docs/overview/concepts/geo-scope

2. Убедитесь, что сервисный аккаунт существует и ему назначены необходимые роли.
3. Создайте ключ доступа к ресурсам каталога.

Проверка наличия необходимых ролей

Для управления ресурсами сервисному аккаунту должны быть назначены роли, дающие необходимые привилегии.

Примечание

Роль `editor` дает привилегии на создание, изменение и удаление любых типов ресурсов в каталоге, однако, в большинстве случаев столь широкие полномочия избыточны. Для более тонкой настройки ознакомьтесь со списком всех ролей[319], предоставляемых Yandex Cloud, и назначьте сервисному аккаунту только те роли, которые действительно необходимы.

Если сервисный аккаунт уже существует, убедитесь, что ему назначены роли, позволяющие управлять необходимыми типами ресурсов:

```
yc resource-manager folder list-access-bindings <folder_name|folder_ID> | grep <sa_ID>
```

В терминал выводится таблица, отображающая список ролей, назначенных сервисному аккаунту с указанным идентификатором <sa_ID>.

```
+------------------+----------------+----------------------+
|     ROLE ID      |  SUBJECT TYPE  |      SUBJECT ID      |
+------------------+----------------+----------------------+
| storage.uploader | serviceAccount | a..................1 |
| editor           | serviceAccount | a..................1 |
+------------------+----------------+----------------------+
```

Создание сервисного аккаунта

Если сервисный аккаунт не существует, создайте его:

```
yc iam service-account create \
   --name=<sa_name> \
   --description="<description>"
```

где

- <sa_name> – имя сервисного аккаунта. Должно удовлетворять следующим требованиям:
  - длина от 3 до 63 символов;
  - может содержать только строчные буквы латинского алфавита, цифры и дефисы;
  - первый символ – буква;
  - последний символ – не дефис;
  - имя должно быть уникальным в рамках облака.
- <description> – опциональное описание сервисного аккаунта.

Подробности о создании сервисных аккаунтов см. в документации Yandex Cloud[320].

[319] https://yandex.cloud/ru/docs/iam/concepts/access-control/roles
[320] https://yandex.cloud/ru/docs/iam/quickstart-sa#create-sa

Назначение ролей сервисному аккаунту

Чтобы назначить сервисному аккаунту роль на каталог, выполните команду:

```
yc resource-manager folder add-access-binding \
   --name <folder_name> \
   --role <role_name> \
   --service-account-name <sa_name>
```

где

- <folder_name> – имя облачного каталога;
- <role_name> – имя роли;
- <sa_name> – имя сервисного аккаунта.

Подробности о назначении ролей сервисным аккаунтам см. в документации Yandex Cloud[321].

Создание пары авторизованных ключей доступа

Чтобы создать пару авторизованных ключей доступа к облачному каталогу и сохранить ее в файл формата JSON, выполните команду:

```
yc iam key create \
   --service-account-name <sa_name> \
   --output key.json
```

где

- <sa_name> – имя сервисного аккаунта;
- `key.json` – имя файла для сохранения пары ключей.

При успешном выполнении команды в терминал выводится информация о созданной паре ключей, например:

```
id: ajonetwothreefoursp63
service_account_id: aj**************o3
created_at: "2023-03-21T08:13:19.002945823Z"
key_algorithm: RSA_2048
```

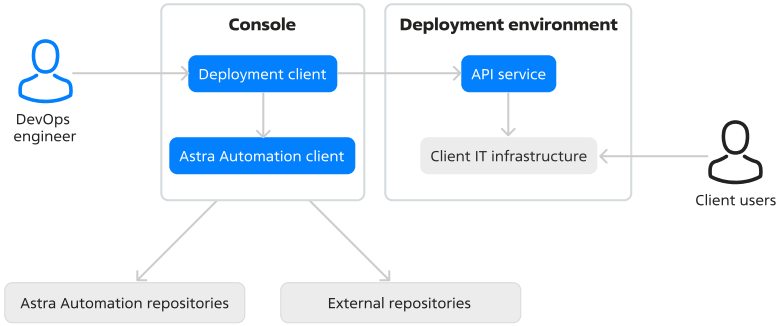
Пример содержимого файла `key.json` (ключи представлены в сокращенном виде):

```
{
   "id": "ajonetwothreefoursp63",
   "service_account_id": "aj**************o3",
   "created_at": "2023-03-21T08:13:19.002945823Z",
   "key_algorithm": "RSA_2048",
   "public_key": "-----BEGIN PUBLIC KEY-----\nMIIBI...DAQAB\n-----END PUBLIC KEY-----\n",
   "private_key": "-----BEGIN...END PRIVATE KEY-----\n"
}
```

Подробности о создании авторизованных ключей доступа см. в документации Yandex Cloud[322].

Типовая схема развертывания облачной инфраструктуры:

[321] https://yandex.cloud/ru/docs/iam/operations/sa/assign-role-for-sa
[322] https://yandex.cloud/ru/docs/iam/operations/authorized-key/create



19.3 Приложение 3. Инструментарий

Для реализации своих функций Astra Automation использует различные методы, инструменты и системы, описания которых приводятся в следующих разделах.

19.3.1 Docker

Docker[323] – это инструмент для создания и запуска контейнеризованных приложений.

Назначение

В Astra Automation утилиты, входящие в состав ПО Docker, используются для следующих задач:

- запуск заданий Ansible в *среде исполнения*;
- *сборка собственного образа* среды исполнения Automation Controller.

Установка

Для установки Docker выполните следующие действия:

1. Обновите список доступных пакетов:

```
sudo apt update
```

2. Установите пакеты:

```
sudo apt install docker.io docker-compose containerd runc --yes
```

3. Добавьте активного пользователя в группу docker:

```
sudo gpasswd --add ${USER} docker
```

4. Для применения привилегий новой группы без перезапуска сессии выполните команду:

```
newgrp docker
```

Настройка размера разделяемой памяти

Разделяемая память – часть оперативной памяти, смонтированная в файловую систему. Операции ввода-вывода в разделяемую память выполняются значительно быстрее чем при работе с реальным блочным устройством. Это позволяет ускорить выполнение задач, требующих частого обращения к файловой системе.

Подробности об использовании разделяемой памяти при сборке образов см. в документации Docker[324].

Для настройки размера разделяемой памяти выполните следующие действия:

[323] https://www.docker.com/
[324] https://docs.docker.com/reference/compose-file/build/#shm_size

1. В конфигурационном файле `/etc/docker/daemon.json` укажите нужное значение для параметра `default-shm-size`. Если такой файл отсутствует, создайте его.

   Список 1: `/etc/docker/daemon.json`

   ```
   { "default-shm-size": "2g" }
   ```

   Значение 2g устанавливает размер разделяемой памяти в 2 гигабайта. Необходимое значение подбирается экспериментально с учетом доступного размера оперативной памяти.

2. Перезапустите службу Docker:

   ```
   sudo systemctl restart docker
   ```

19.3.2 Git

Для работы с репозиториями необходим клиент системы управления версиями Git[325]. Для его установки выполните команду:

```
sudo apt update && sudo apt install git --yes
```

19.3.3 Podman

Podman[326] – это инструмент с открытым исходным кодом, используемый для создания и запуска приложений в контейнерах. В отличие от *Docker*, по умолчанию Podman запускает контейнеры в непривилегированном режиме.

Назначение

В Astra Automation утилиты, входящие в состав ПО Podman, используются для следующих задач:

- запуск заданий Ansible в *среде исполнения*;
- сборка собственного образа среды исполнения Automation Controller.

Установка

Важно

Пакеты Podman доступны в репозиториях Astra Linux Special Edition начиная с оперативного обновления 1.7.2 (БЮЛЛЕТЕНЬ № 2022-0819SE17). Инструкции по обновлению ОС приведены в разделе справочного центра Оперативные обновления для Astra Linux Special Edition РУСБ.10015-01 (очередное обновление 1.7), РУСБ.10015-10, РУСБ.10015-37[327].

Для установки пакетов Podman выполните следующие действия:

1. Получите номер установленного оперативного обновления ОС:

   ```
   cat /etc/astra_version
   ```

   Команда выводит в терминал строку вида:

   ```
   1.7.5
   ```

2. Добавьте в файл `/etc/apt/sources.list` ссылку на расширенный (extended) репозиторий Astra Linux Special Edition 1.7:

[325] https://git-scm.com/
[326] https://podman.io
[327] https://wiki.astralinux.ru/pages/viewpage.action?pageId=158612043

```
deb https://download.astralinux.ru/astra/frozen/1.7_x86-64/<version>/repository-
↪extended/ 1.7_x86-64 main contrib non-free
```

где <version> – версия установленного оперативного обновления Astra Linux Special Edition. Для примера выше указанная строка имеет вид:

```
deb https://download.astralinux.ru/astra/frozen/1.7_x86-64/1.7.5/repository-
↪extended/ 1.7_x86-64 main contrib non-free
```

3. Обновите список доступных пакетов:

```
sudo apt update
```

4. Установите пакеты Podman:

```
sudo apt install podman --yes
```

Настройка Podman

Для корректной работы Podman необходимо настроить группу управления и драйвер хранилища.

Настройка группы управления

Для настройки группы управления добавьте в секцию `[engine]` конфигурационного файла `.config/containers/containers.conf` строку:

```
[engine]
cgroup_manager = "cgroupfs"
```

Если такой файл отсутствует, создайте его.

Настройка драйвера хранилища

Примечание

Настройка необходима, если установлен пакет Podman версии 4.3.1.

Podman версии 4.3.1 настроен на использование драйвера хранилища `vfs`, который приводит к быстрому переполнению дискового пространства рабочей станции.

Проверьте версию Podman с помощью следующей команды:

```
podman --version
```

Для перевода Podman на драйвер `overlayfs` выполните следующие действия:

1. Создайте в домашнем каталоге пользователя файл `~/.config/containers/storage.conf` со следующим содержимым:

```
[storage]
driver = "overlay"
```

2. Обновите настройку Podman:

```
podman system reset -f
```

Проверка корректности установки

Для проверки корректности установки Podman выполните следующую команду:

```
podman run -it --rm hello-world
```

Здесь:

- `-it` – интерактивный режим вывода.
- `--rm` – автоматическое удаление контейнера после выполнения команды.
- `hello-world` – ссылка на образ.

Если Podman настроен корректно, в терминал выводится приветственное сообщение.

19.3.4 Vagrant

Vagrant[328] относится к инструментам для создания ВМ и объединения их в инфраструктуру в некотором виртуальном окружении, используя такие известные технологии, как VMware, VirtualBox и AWS. К такому виртуальному окружению относится, например, сервер с установленной на нем одной из упомянутых систем виртуализации.

Vagrant предоставляет развитый API, необходимый для автоматизации развертывания систем. Следуя парадигме *IAC*, указания для Vagrant создают в файле `Vagrantfile` в виде директив, объединенных в общий поток последовательных действий.

Применение

Приложение Vagrant удобно использовать для отладки развертывания пилотной инфраструктуры в локальной среде. Типовая последовательность использования Vagrant состоит из следующих шагов:

1. Создание и отладка потока управляющих директив на локальном узле с помощью VMware или VirtualBox.
2. Перенос полученного потока в производственную среду на базе облачных технологий или на собственные серверы.

В этом документе рассматривается использование Vagrant совместно с VirtualBox.

Примечание

Приведенные ниже инструкции проверены в ОС Astra Linux Special Edition 1.7.4, 1.7.5 и 1.7.5.UU1.

Установка

Приведенные ниже инструкции по установке и настройке ПО предназначены для ОС Astra Linux Special Edition 1.7. Пошаговые инструкции для других ОС доступны на сайтах разработчиков соответствующего ПО:

- Vagrant[329]
- VirtualBox[330]

Подготовка к установке

Подготовьте ОС к установке и настройке Vagrant и VirtualBox.

1. Если установка производится на ВМ, включите в ее настройках вложенную виртуализацию (nested virtualization). Для получения инструкций обратитесь к производителю ПО, используемого для виртуализации.

[328] https://developer.hashicorp.com/vagrant
[329] https://developer.hashicorp.com/vagrant/install
[330] https://www.virtualbox.org/wiki/Downloads

2. Получите номер установленного оперативного обновления ОС:

```
cat /etc/astra_version
```

Команда выводит в терминал строку вида:

```
1.7.5uu1
```

3. Добавьте в файл `/etc/apt/sources.list` ссылки на базовый (base) и расширенный (extended) репозитории Astra Linux Special Edition 1.7:

```
deb https://download.astralinux.ru/astra/frozen/1.7_x86-64/<version>/repository-
↪base/     1.7_x86-64 main contrib non-free
deb https://download.astralinux.ru/astra/frozen/1.7_x86-64/<version>/repository-
↪extended/ 1.7_x86-64 main contrib non-free
```

где <version> – версия установленного оперативного обновления Astra Linux Special Edition. Для примера выше указанные строки имеют вид:

```
deb https://download.astralinux.ru/astra/frozen/1.7_x86-64/1.7.5/uu/1/repository-
↪base/     1.7_x86-64 main contrib non-free
deb https://download.astralinux.ru/astra/frozen/1.7_x86-64/1.7.5/uu/1/repository-
↪extended/ 1.7_x86-64 main contrib non-free
```

4. Обновите список доступных пакетов:

```
sudo apt update
```

5. Получите номер версии и вариант сборки установленного ядра:

```
uname -r
```

В терминал выводится строка вида:

```
6.1.50-1-generic
```

Здесь `6.1.50-1` – версия ядра, `generic` – вариант сборки.

6. Установите менеджер загрузок `wget`, заголовочные файлы используемого ядра и утилиты, необходимые для сборки модулей ядра:

```
sudo apt install build-essential linux-headers-<version>-<build_variant> make wget -
↪-yes
```

где <version> и <build_variant> – версия ядра и вариант его сборки соответственно. Для примера выше нужная команда имеет вид:

```
sudo apt install build-essential linux-headers-6.1.50-1-generic make wget --yes
```

Совет

Шаги получения информации о версии ядра и установки его заголовочных файлов можно объединить в одну команду:

```
sudo apt install build-essential linux-headers-$(uname -r) make wget --yes
```

Установка VirtualBox

Для установки VirtualBox выполните следующие действия:

1. Загрузите пакет `libvpx5` из репозиториев Debian Linux, например:

```
wget http://security.debian.org/debian-security/pool/updates/main/libv/libvpx/
↪libvpx5_1.7.0-3+deb10u2_amd64.deb
```

Совет

Актуальная ссылка на загрузку доступна на странице пакета: https://packages.debian.org/buster/amd64/libvpx5/download

2. Установите загруженный пакет:

```
sudo dpkg -i libvpx5_1.7.0-3+deb10u2_amd64.deb
```

3. Импортируйте ключ репозитория VirtualBox:

```
wget https://www.virtualbox.org/download/oracle_vbox_2016.asc -O- |\
    sudo gpg --dearmor --yes --output /etc/apt/trusted.gpg.d/oracle.gpg
```

4. Создайте в каталоге `/etc/apt/sources.list.d/` файл `virtualbox.list` со следующим содержимым:

```
deb [arch=amd64] https://download.virtualbox.org/virtualbox/debian buster contrib
```

5. Обновите список доступных пакетов:

```
sudo apt update
```

6. Установите пакет `virtualbox-7.0`:

```
sudo apt install virtualbox-7.0 --yes
```

7. Получите номер установленной версии VirtualBox:

```
apt policy virtualbox-7.0
```

В терминал выводятся строки вида:

```
virtualbox-7.0:
  Установлен: 7.0.16-162802~Debian~buster
  Кандидат:   7.0.16-162802~Debian~buster
  Таблица версий:
 *** 7.0.16-162802~Debian~buster 500
        500 https://download.virtualbox.org/virtualbox/debian buster/contrib amd64␣
↪Packages
        100 /var/lib/dpkg/status
```

Здесь `7.0.16` – версия VirtualBox.

8. Загрузите Extension Pack[331] для установленной версии VirtualBox:

```
wget https://download.virtualbox.org/virtualbox/<version>/Oracle_VM_VirtualBox_
↪Extension_Pack-<version>.vbox-extpack
```

где <version> – версия VirtualBox.

Для примера выше указанная команда имеет вид:

```
wget https://download.virtualbox.org/virtualbox/<version>/Oracle_VM_VirtualBox_
↪Extension_Pack-7.0.16.vbox-extpack
```

9. Установите Extension Pack:

[331] https://www.virtualbox.org/wiki/Downloads

```
sudo vboxmanage extpack install --replace <path>
```

где <path> – путь к загруженному ранее файлу.

Для примера выше указанная команда имеет вид:

```
sudo vboxmanage extpack install --replace ./Oracle_VM_VirtualBox_Extension_Pack-7.0.
↪16.vbox-extpack
```

Установка Vagrant

Для установки Vagrant выполните следующие действия:

1. Загрузите DEB-пакет Vagrant с сайта разработчика, например:

```
wget https://releases.hashicorp.com/vagrant/2.4.1/vagrant_2.4.1-1_amd64.deb
```

Примечание

Если сайт разработчика по какой-либо причине недоступен, используйте для загрузки зеркало репозитория:

```
wget https://releases.comcloud.xyz/vagrant/2.4.1/vagrant_2.4.1-1_amd64.deb
```

2. Установите загруженный пакет:

```
sudo dpkg -i vagrant_2.4.1-1_amd64.deb
```

3. Определите путь к каталогу, в котором хранятся расширения Vagrant:

```
dpkg -L vagrant | egrep 'plugins/guests$'
```

В терминал выводится строка вида:

```
/opt/vagrant/embedded/gems/gems/vagrant-<version>/plugins/guests
```

где <version> – номер установленной версии Vagrant.

4. Загрузите архив с расширением для Vagrant, позволяющим ему работать с образами Astra Linux:

```
wget https://dl.astralinux.ru/files/astra-vagrant.tar.gz
```

5. Распакуйте загруженный архив в каталог с расширениями Vagrant, например:

```
sudo tar xf astra-vagrant.tar.gz -C <plugins_dir>
```

где <plugins_dir> – путь к каталогу с расширениями Vagrant.

Проверка корректности установки

Для проверки корректности установки Vagrant и VirtualBox выполните следующие действия:

1. В любом каталоге создайте `Vagrantfile` с простейшей конфигурацией ВМ:

```
# frozen_string_literal: true

Vagrant.configure("2") do |config|
  config.vm.box = "alse-vanilla-base/1.7.5uu1"
  config.vm.box_url = "https://dl.astralinux.ru/vagrant/alse-vanilla-base%2F1.7.5uu1
↪"
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
  config.vm.define "VG" do |conf|
    conf.vm.hostname = "VG"
  end
end
```

В этом файле описана ВМ с названием VG, использующая образ с Astra Linux Special Edition 1.7.5.UU1.

Подробности об именовании и составе образов см. в документации универсальных базовых образов Astra Linux[332].

2. Для создания и запуска ВМ выполните команду:

```
vagrant up
```

3. Подключитесь к созданной ВМ по SSH:

```
vagrant ssh
```

При успешном подключении приглашение командной строки меняется на следующее:

```
vagrant@VG:~$
```

4. Для отключения от созданной ВМ выполните команду:

```
exit
```

Подготовка инфраструктуры

Приведенные ниже инструкции описывают типовую последовательность действий для создания инфраструктуры ВМ, управляемой с помощью Vagrant.

ВМ на локальной машине

Для создания ВМ с установленной операционной системой Astra Linux выполните следующие действия:

1. Создайте пару ключей SSH, используемых для подключения к ВМ:

```
ssh-keygen -C "<comment>" -f ~/.ssh/deployment -N ""
```

В результате выполнения команды в каталоге `~/.ssh/` создаются два файла, содержащие приватный (`deployment`) и публичный (`deployment.pub`) ключи SSH соответственно.

2. Создайте файл `Vagrantfile` со следующим содержимым:

```
# -*- frozen_string_literal: true -*-

Vagrant.configure('2') do |config|
  config.vm.box = 'alse-vanilla-base/1.7.5uu1' # Название бокса

  # Ссылка на образ
  config.vm.box_url = 'https://dl.astralinux.ru/vagrant/alse-vanilla-base%2F1.7.5uu1
↪'

  config.vm.provision 'file',
                      source: '~/.ssh/deployment.pub',
                      destination: '/home/vagrant/.ssh/id_rsa.pub'

  config.vm.provision 'shell',
```

(продолжается на следующей странице)

[332] https://registry.astralinux.ru/latest/descriptions/local/vagrant/

(продолжение с предыдущей страницы)

```
                    inline: 'cat /home/vagrant/.ssh/id_rsa.pub >> /home/vagrant/.
↪ssh/authorized_keys'

  # Настройка параметров ВМ
  config.vm.define 'host01' do |node|
    node.vm.hostname = 'host01' # Короткое имя хоста

    # Настройки, специфичные для VirtualBox
    node.vm.provider 'virtualbox' do |vb|
      vb.cpus = 4 # Кол-во ядер CPU
      vb.memory = 4096 # Объем RAM, МБ
    end

    # Параметры сети
    node.vm.network 'private_network',
                    ip: '192.168.56.11', # Статический IP-адрес
                    netmask: '255.255.255.0', # Маска подсети
                    dhcp_enabled: false # Запрет использования DHCP
  end
end
```

Имя бокса Vagrant задается в строке:

```
config.vm.box = 'alse-vanilla-base/1.7.5uu1' # Название бокса
```

После этого указывается ссылка на образ:

```
config.vm.box_url = 'https://dl.astralinux.ru/vagrant/alse-vanilla-base%2F1.7.5uu1'
```

Чтобы получить доступ к ВМ по SSH, необходимо в подкаталоге `~/.ssh/` нужного пользователя ВМ разместить публичный ключ SSH, а также добавить его в список доверенных ключей:

```
config.vm.provision 'file',
                    source: '~/.ssh/deployment.pub',
                    destination: '/home/vagrant/.ssh/id_rsa.pub'

config.vm.provision 'shell',
                    inline: 'cat /home/vagrant/.ssh/id_rsa.pub >> /home/vagrant/.
↪ssh/authorized_keys'
```

Короткое имя хоста (`vm`) задается в параметре `node.vm.hostname`:

```
node.vm.hostname = 'vm' # Короткое имя хоста
```

В этих строках задается количество доступных ВМ ядер CPU и МБ RAM:

```
vb.cpus = 4 # Кол-во ядер CPU
vb.memory = 4096 # Объем RAM, МБ
```

Настройки подключения ВМ к сети:

```
node.vm.network 'private_network',
                ip: '192.168.56.11', # Статический IP-адрес
                netmask: '255.255.255.0', # Маска подсети
                dhcp_enabled: false # Запрет использования DHCP
```

3. Для проверки корректности синтаксиса созданного `Vagrantfile` выполните команду:

```
vagrant validate
```

При отсутствии ошибок в терминал выводится строка:

```
Vagrantfile validated successfully.
```

В случае наличия ошибок исправьте их и выполните повторную проверку синтаксиса `Vagrantfile`.

4. Для создания и запуска VM выполните команду:

```
vagrant up
```

Дождитесь завершения выполнения команды, это может занять некоторое время.

5. Проверьте состояние ВМ:

```
vagrant status
```

Она должна быть активна (состояние `running`):

```
Current machine states:

vm               running (virtualbox)

The VM is running. To stop this VM, you can run `vagrant halt` to
shut it down forcefully, or you can run `vagrant suspend` to simply
susptend the virtual machive. In either case, to restart it again,
simply run `vagrant up`.
```

Если в процессе создания или запуска ВМ будут выведены сообщения об ошибках, выполните следующие действия:

1. Проверьте содержимое `Vagrantfile` на наличие ошибок в параметрах настройки ВМ.
2. Удалите существующую ВМ и создайте ее заново:

```
vagrant destroy --force && vagrant up
```

Инфраструктура из нескольких ВМ

Следующий пример файла `Vagrantfile` демонстрирует задание на создание инфраструктуры из двух ВМ:

Содержимое файла `Vagrantfile`

```
# frozen_string_literal: true

boxes = {
  'dc01' => {
    box: 'alse-vanilla-base/1.7.5uu1',
    box_url: 'https://dl.astralinux.ru/vagrant/alse-vanilla-base%2F1.7.5uu1',
    cpu_cores: '2',
    memory: '2048',
    ipv4: '192.168.56.10'
  },
  'dc02' => {
    box: 'alse-vanilla-base/1.7.5uu1',
    box_url: 'https://dl.astralinux.ru/vagrant/alse-vanilla-base%2F1.7.5uu1',
    cpu_cores: '2',
    memory: '2048',
    ipv4: '192.168.56.11'
  }
}

Vagrant.configure(2) do |config|
  boxes.each do |hostname, cfg|
```

(продолжается на следующей странице)

(продолжение с предыдущей страницы)

```
    config.vm.define hostname do |host|
      host.vm.boot_timeout = 600
      host.vm.hostname = hostname
      host.vm.box = cfg[:box]
      host.vm.box_url = cfg[:box_url]
      host.vm.provider 'virtualbox' do |v|
        v.cpus = cfg[:cpu_cores]
        v.memory = cfg[:memory]
      end
      host.vm.network 'private_network', ip: cfg[:ipv4]
    end
  end
end
```

Настройка состоит из двух частей:

- Первая часть boxes задает параметры двух ВМ.
- Вторая часть описывает создание этих ВМ с использованием параметров, заданных в структуре boxes.

Удаление ВМ и образов

Для удаления неиспользуемых ресурсов выполните следующие действия:

1. Определите список существующих ВМ:

   ```
   vagrant global-status
   ```

2. Удалите ненужные ВМ с помощью команды:

   ```
   vagrant destroy <VM_name>
   ```

3. Получите список образов, загруженных на компьютер:

   ```
   vagrant box list
   ```

4. Удалите ненужные образы:

   ```
   vagrant box remove <box_name>
   ```

ГЛАВА 20

Термины и определения

CIDR
Classless Inter-Domain Routing (бесклассовая междоменная маршрутизация) – метод IP-адресации, который позволяет экономить адресное пространство путем использования сетевых масок переменной длины.

CLI
Command Line Interface – интерфейс командной строки.

Cloud-Init
Промышленный стандарт, позволяющий настраивать ОС виртуальных машин в облачных сервисах с помощью специального файла. Особенностью использования Cloud-Init является то, что настройка виртуальной машины выполняется только один раз – во время первой загрузки. (Источник[333])

DAG
Directed Acyclic graph (направленный ациклический граф, ориентированный ациклический граф) – орграф, в котором отсутствуют направленные циклы, но могут быть «параллельные» пути, выходящие из одного узла и разными путями приходящие в конечный узел. Направленный ациклический граф является обобщением дерева, точнее, их объединения – леса. (Источник[334])

EE
Execution Environment (среда исполнения) – контейнер с образом на базе Astra Linux Special Edition, который рекомендуется использовать для развертывания и тестирования кода, запускаемого с помощью Ansible.

FQCN
Fully Qualified Content Name – полностью определенное название контента.

FQDN
Fully Qualified Domain Name – полностью определенное имя домена.

HA
High Availability (высокая доступность) – подход к развертыванию веб-сайтов или приложений, обеспечивающий их доступность для пользователей даже при отказе отдельных частей инфраструктуры. (Источник[335])

HCL
HashiCorp Configuration Language – декларативный язык описания настроек программ-

[333] https://cloud-init.io/
[334] https://ru.wikipedia.org/wiki/Ориентированный_ациклический_граф
[335] https://ru.wikipedia.org/wiki/Высокая_доступность

ного обеспечения, разработанный для Terraform и используемый также другими системами. (Источник[336])

IaC

Infrastructure as Code – способ (подход) автоматического развертывания сетевой инфраструктуры от одиночных серверов до центров обработки данных (ЦОД) и облачных структур. Способ основан на определении требуемой инфраструктуры в файлах, которые интерпретирует и реализует система развертывания инфраструктур. (Источник[337])

Данный подход реализован в Ansible, Terraform, Astra Automation и других системах автоматизации.

IPA

Identity, Policy, Audit – три основных компонента системы безопасности, положенные в основу FreeIPA. (Источник[338])

OCFS2

Система управления файлами Oracle Cluster File System второй версии, обеспечивающая доступ к единому пространству файлов. Используется в основном для виртуализации (Oracle VM), кластерных баз данных (Oracle RAC), кластеров на промежуточном ПО (Oracle E-Business Suite) и подобных системах. (Источник[339])

RBAC

Role Based Access Control (управление доступом на основе ролей) – политика избирательного управления доступом, при которой права доступа субъектов системы на объекты группируются с учетом специфики их применения, образуя роли. (Источник[340])

SSO

Single Sign-On (технология единого входа) – технология, при использовании которой пользователь переходит из одного портала в другой, либо из одной системы в другую, не связанную с первой системой, без аутентификации.

(Источник[341])

TUI

Text-based User Interface (текстовый пользовательский интерфейс) – разновидность интерфейса пользователя, использующая при вводе-выводе и представлении информации исключительно набор буквенно-цифровых символов и символов псевдографики.

(Источник[342])

VCS

Version Control System (система контроля версий) – программное обеспечение для облегчения работы с изменяющейся информацией. Система управления версиями позволяет хранить несколько версий одного и того же документа, при необходимости возвращаться к более ранним версиям, определять, кто и когда сделал то или иное изменение, и многое другое. (Источник[343])

YC CLI

Yandex Cloud Command Line Interface – это интерфейс для управления ресурсами Yandex Cloud с помощью утилиты командной строки. (Источник[344])

ГИС

Государственная информационная система.

СЗИ

Средство защиты информации – техническое, программное, программно-техническое

[336] https://octopus.com/blog/introduction-to-hcl-and-hcl-tooling
[337] https://en.wikipedia.org/wiki/Infrastructure_as_code
[338] https://www.freeipa.org/page/About
[339] https://ru.wikipedia.org/wiki/OCFS
[340] https://ru.wikipedia.org/wiki/Управление_доступом_на_основе_ролей
[341] https://ru.wikipedia.org/wiki/Технология_единого_входа
[342] https://ru.wikipedia.org/wiki/Текстовый_интерфейс_пользователя
[343] https://ru.wikipedia.org/wiki/Система_управления_версиями
[344] https://yandex.cloud/ru/docs/cli/

средство, вещество и (или) материал, предназначенные или используемые для защиты информации. (ГОСТ Р 50922-2006. Защита информации. Основные термины и определения)

СЗИ Astra Linux Special Edition
Средства защиты информации в Astra Linux Special Edition – комплекс программных средств защиты информации, реализующих функции безопасности Astra Linux.

Установочный узел
Установочным называется узел, на котором запускается установщик платформы Astra Automation.

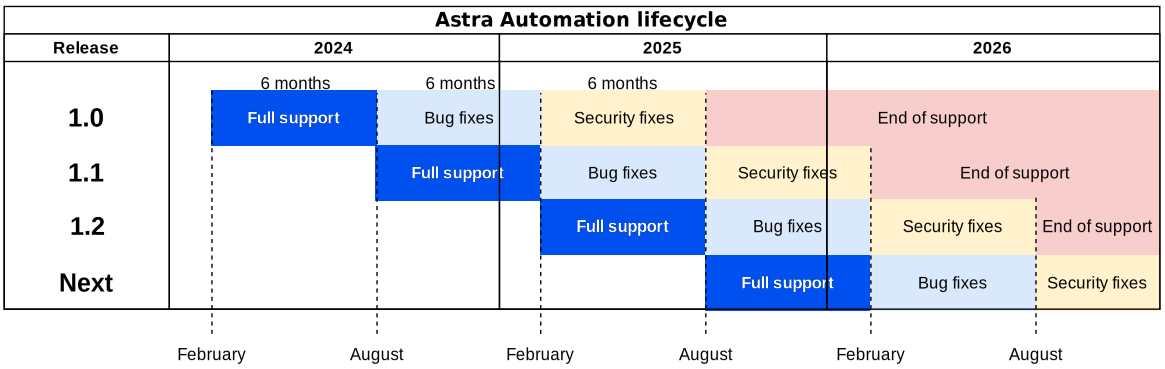
ГЛАВА 21

Обновления

Объявления о выпусках новых версий продукта Astra Automation, обновлениях его характеристик и исправлениях программного обеспечения выходят регулярно одновременно с появлением каждой версии.

21.1 Жизненный цикл продукта

Выпуск новой стабильной версии продукта происходит каждые 6 месяцев согласно следующей диаграмме:



Особенности:

- Одновременно поддерживаются три стабильные версии.
- В течение жизненного цикла каждой стабильной версии проводятся регулярные обновления.
- Жизненный цикл каждой стабильной версии содержит три стадии:
  - **Full support** – полная техническая поддержка с выпуском обновлений по изменению функциональности, исправлению ошибок и повышению безопасности.
  - **Bug fixes** – техническая поддержка, обеспечиваемая выпуском обновлений с исправлениями ошибок и повышением безопасности.
  - **Security fixes** – техническая поддержка заданного уровня безопасности, обеспечиваемая выпуском обновлений по устранению возможных уязвимостей.

Более подробно этот цикл представлен в *истории обновлений*. Отдельно представлен *состав платформы* для каждой версии.

21.2 Версионирование

Версия выпуска Astra Automation представляется в соответствии со следующей схемой:

```
<field1>.<field2>-<update>
```

где:

- <field1> и <field2> – целые числа, вместе составляющие номер стабильной версии.

  Примечание

  Обновления не основаны на семантическом версионировании.

- <update> – строка вида upd1, upd2 и так далее, означающая внесение изменений и исправление ошибок. При выпуске новой стабильной версии это поле отсутствует.

21.3 Что означает техническая предварительная версия (technical preview)

Некоторые возможности платформы могут быть предоставлены до проведения полного тестирования соответствующих компонентов. Такие возможности помечаются как «техническая предварительная версия», позволяют пользователю использовать их предварительно на свое усмотрение и не поддерживаются службой технической поддержки.

Предупреждение

На такие возможности не распространяется Положение о технической поддержке программы для Astra Automation[345].

21.4 Цикл разработки документации

Разработка документации синхронизирована с разработкой продукта и имеет следующие особенности:

- Версии документации совпадают со стабильными версиями продукта, то есть имеют значения 1.0, 1.1 и так далее с периодом выпуска очередной версии 6 месяцев.
- Одновременно поддерживаются до трех последних версий документации с возможностью переключения между ними.
- С выходом минорного обновления продукта, например, 1.0-upd1 версия документации не изменяется. Новые изменения, которые появляются в документации в связи с выходом такого обновления, помечаются в виде короткой заметки в тексте сразу после соответствующего заголовка или параграфа.

Документация и спецификация API доступны также для загрузки в формате PDF с помощью соответствующих кнопок в правом верхнем углу экрана. Файлы PDF обновляются с выпуском очередной версии продукта.

21.5 Состав платформы

Следующая таблица содержит состав Astra Automation и зависимость от операционной системы и ее версии. Данные представлены в обратном хронологическом порядке.

[345] https://astra.ru/support/support-docs/polozhenie-o-tekhnicheskoy-podderzhke-astra-automation/

Вер-сия	Состав	Операционная система	СУБД
2.0	Ansible Core 2.15.10 Platform Gateway X.Y.Z Automation Controller 2.1.1 Private Automation Hub 1.1.4 Event-Driven Automation 1.0.2 Execution Environment aa-minimal-ee 1.3.0 Execution Environment aa-full-ee 0.8.0 Execution Environment aa-control-ee 0.4.0 Decision Environment aa-minimal-de 1.2.0 Decision Environment aa-full-de 1.2.0 Development Container aa-cdk 0.3.0 Ansible Navigator 25.4.1+aa1.1.0 Ansible Builder 3.1.0+aa1.2.0 Ansible Creator 25.4.1+aa1.0.1 Ansible Lint 25.4.0+aa1.1.0 Ansible Pytest 25.4.1+aa1.1.0 Ansible Molecule 25.4.0+aa1.1.0 Ansible Tox 25.4.0+aa1.2.0 Ansible Sign 0.1.1+aa1.2.0	Astra Linux Special Edition 1.7.6+ Astra Linux Special Edition 1.7.7.UU2 Astra Linux Special Edition 1.7.8 Astra Linux Special Edition 1.8.1.UU1 Astra Linux Special Edition 1.8.2.UU1 Astra Linux Special Edition 1.8.3, 1.8.3.UU1	PostgreSQL 15
1.2-upd4	Ansible Core 2.15.10 Automation Controller 2.1.1 Private Automation Hub 1.1.4 Event-Driven Automation 1.0.2 Execution Environment aa-minimal-ee 1.3.0 Execution Environment aa-full-ee 0.8.0 Execution Environment aa-control-ee 0.4.0 Decision Environment aa-minimal-de 1.2.0 Decision Environment aa-full-de 1.2.0 Development Container aa-cdk 0.3.0 Ansible Navigator 24.9.0+aa1.1.2 Ansible Builder 3.1.0+aa1.1.1 Ansible Creator 24.10.1+aa2.0.1 Ansible Lint 24.9.2+aa1.1.1 Ansible Pytest 24.9.0+aa2 Ansible Molecule 24.9.0+aa1.1.3 Ansible Tox 25.4.0+aa1.1.1 Ansible Sign 0.1.1+aa1.1.1	Astra Linux Special Edition 1.7.6+ Astra Linux Special Edition 1.7.7.UU2 Astra Linux Special Edition 1.7.8 Astra Linux Special Edition 1.8.1.UU1 Astra Linux Special Edition 1.8.2.UU1 Astra Linux Special Edition 1.8.3, 1.8.3.UU1	PostgreSQL 15
1.2-upd3	Ansible Core 2.15.10	Astra Linux Special Edition 1.7.6+	PostgreSQL 15

21.6 История обновлений

Следующая таблица содержит перечень обновлений в обратном хронологическом порядке и основные данные по ним. В ней отсутствуют обновления, выпущенные после завершения стадии features конкретной версии продукта. Например, здесь не будет присутствовать выпуск 1.0-upd3, потому что он выходит позднее выпуска следующей стабильной версии продукта, то есть версии 1.1. Для каждой стабильной версии есть отдельная таблица с полным перечнем всех обновлений, например, *по версии 1.0*.

Версия	Дата выпуска	Обновление характеристик (главное)	Исправления (главные)
2.0	27.11.2025	Перевод платформы на унифицированный интерфейс. Добавление модели развертывания в Kubernetes.	Устранение критических уязвимостей в утилите `aa-setup`.
1.2-upd4	12.09.2025	–	Исправлено поведение, когда пароль к базе данных сервера PostgreSQL, созданного через `aa-setup`, был доступен в открытом виде.
1.2-upd3	02.09.2025	Улучшена логика работы с поврежденными базами данных.	Скорректирована конфигурация для сервиса `supervisor`: обработчик событий работает с привилегиями `awx` вместо `root`.
1.2-upd2	11.07.2025	Добавлена поддержка работы компонентов Astra Automation в режиме замкнутой программной среды (ЗПС). Разработано решение по оптимизации резервного копирования и восстановления данных.	Ошибки, связанные с обработкой сложных сценариев, требующих длительного времени.
1.2-upd1	28.03.2025	В графическом интерфейсе контроллера добавили ссылку на документ по активации подписки, необходимой после развертывания платформы. Упрощен перенос коллекций Ansible в приватный реестр для платформы, установленной а закрытом контуре без доступа к интернету.	Ошибки, связанные с некорректным восстановлением из резервной копии.
1.2	14.02.2025	Первый выпуск компонента Event-Driven Automation. Переход на новое семейство образов служебных контейнеров.	Устранены ошибки, из-за которых в исполнительном узле, установленном из пакета offline bundle, не запускались задания. Исправлено поведение, когда утилита `aa-setup` требовала указать переменную `ansible_host` даже при использовании FQDN в описании узла.
1.1-upd1	21.10.2024	Первый выпуск Private Automation Hub.	Устранена ошибка, препятствующая передаче и отображению переменных, заданных в шаблоне, во вкладке **Подробности** (Details) выполненного задания.
1.1	10.06.2024	Выпуск пакета Ansible Navigator и Ansible Builder в составе SDK	Устранена ошибка передачи

21.7 Стабильные версии

21.7.1 Версия 2.0

Дата выпуска: 27.11.2025
Тип выпуска: стабильная версия

Примечание

1. Данная версия является мажорной, то есть содержит существенные изменения, которые могут повлиять на текущие процессы после обновления платформы.
2. Все обновления представлены по отношению к версии 1.2-upd4, новейшей на момент выпуска данного обновления.

Предупреждение

Перед переходом на новую версию ознакомьтесь также с *известными проблемами*.

Состав платформы и совместимость с версиями операционной системы представлены в *таблице*.

Главное

Следующие обновления представляются наиболее важными:

- **Platform Gateway** – единая точка входа для веб-интерфейса и API всех компонентов платформы;
- **Унифицированная аутентификация и RBAC** – централизованное управление доступом и Single Sign-On для всех сервисов;
- **Поддержка Kubernetes** – развертывание в кластере Kubernetes через операторы (*technical preview*);
- **Архитектурные улучшения** – горизонтальное масштабирование и высокая доступность Event-Driven Automation;
- **Устранение критических уязвимостей** – устранены уязвимости CVE-2024-53899 и CVE-2024-39689.
- **Завершение технической поддержки версии 1.0** – жизненный цикл поддержки Astra Automation 1.0 истек согласно *правилам*.

Изменения в развертывании

Поддерживаются две модели развертывания:

- Установка каждого компонента платформы на одной или нескольких виртуальных машинах. Последнее применяют для обеспечения высокой доступности и распределения нагрузки. Это традиционная модель развертывания с использованием утилиты `aa-setup`.
- Установка в кластере Kubernetes через операторы. Центральная часть Astra Automation развертывается на подах Kubernetes с динамической репликацией и распределением их по рабочим узлам кластера. Плоскость исполнения состоит из узлов на виртуальных машинах или физических серверах.

Примечание

1. В этом выпуске модель представлена как *техническая предварительная версия*.
2. Эта модель представляется наиболее перспективной в дальнейшем развитии платформы.

Особенности развертывания в кластере Kubernetes:

- **Декларативное развертывание** – управление платформой через Kubernetes CustomResourceDefinitions:
  - Предназначено для администраторов Kubernetes.
  - Позволяет описывать состав и настройку платформы с помощью файлов-манифестов.
  - Обеспечивает автоматическое применение изменений через операторы.
  - Функциональность доступна в режиме Technical Preview.
- **Автоматическое масштабирование** – динамическое изменение количества реплик под нагрузкой:
  - Предназначено для администраторов высоконагруженных платформ.
  - Обеспечивает автоматическое масштабирование компонентов на основе метрик.
  - Повышает эффективность использования ресурсов кластера.
  - Функциональность доступна в режиме Technical Preview.
- **Встроенная отказоустойчивость** – автоматическое восстановление при сбоях:
  - Предназначено для администраторов, требующих высокой доступности.
  - Использует встроенные механизмы Kubernetes для обеспечения отказоустойчивости.
  - Автоматически перезапускает отказавшие компоненты.
  - Функциональность доступна в режиме Technical Preview.

Новые возможности и улучшения

Новая версия содержит несколько новых возможностей и улучшений существующих характеристик.

Утилита развертывания aa-setup

Следующие улучшения предназначены для администраторов, выполняющих установку и обновление платформы:

- Добавлена ранняя проверка используемых ресурсов и их описания (fast fail):
  - Предотвращает траты времени на развертывание, которое заведомо завершится неудачей.
  - Перед началом развертывания утилита автоматически проверяет готовность окружения, корректность его настройки и описание инвентаря. При обнаружении проблем развертывание прерывается с понятным сообщением об ошибке и ссылкой на документацию.
  - Улучшение доступно сразу после обновления утилиты `aa-setup`.

  Проверяемые параметры и ресурсы:
  - корректность описания инвентаря, в частности наличие обязательных параметров, например `automationgateway_admin_password`;
  - версия операционной системы Astra Linux (поддерживаемые версии, архитектура x86_64, отсутствие FLY desktop);

  - аппаратные ресурсы серверов (CPU, RAM, свободное место на диске);
  - доступность внешних ресурсов при развертывании с доступом в интернет (репозитории Debian, hub.astra-automation.ru, пользовательские реестры);
  - корректность параметров базы данных PostgreSQL (доступность по указанным параметрам, версия, отсутствие ручной установки);
  - доступность необходимых портов TCP у компонентов платформы.
- Оптимизирован процесс миграции больших баз данных.

Platform Gateway

В состав платформы добавлен новый компонент Platform Gateway, обеспечивающий единую точку входа для всех сервисов. Ключевые возможности:

- **Единая аутентификация** – централизованное управление пользователями и интеграция с внешними источниками аутентификации (LDAP, SAML, RADIUS, OAuth 2.0, OIDC) через единый интерфейс:
  - Предназначена для администраторов платформы.
  - Позволяет настроить аутентификацию один раз для всех компонентов.
  - Обеспечивает Single Sign-On для Automation Controller, Private Automation Hub и Event-Driven Automation.
  - Новая возможность доступна сразу после обновления платформы.
- **Унифицированная ролевая модель RBAC** (Role-Based Access Control) – единая модель доступа для всех компонентов:
  - Предназначена для администраторов безопасности.
  - Упрощает управление правами доступа через единый интерфейс.
  - Централизует управление организациями, пользователями и командами пользователей.
  - Новая возможность доступна сразу после обновления платформы.
- **Интеллектуальная маршрутизация** – автоматическое перенаправление запросов к соответствующим компонентам:
  - Предназначена для разработчиков интеграции.
  - Обеспечивает прозрачный доступ ко всем API через единую точку входа.
  - Автоматически добавляет токены аутентификации при перенаправлении запросов.
  - Новая возможность доступна сразу после обновления платформы.
- **Централизованный мониторинг** – единая точка доступа для проверки статуса всех сервисов:
  - Предназначен для операторов систем мониторинга.
  - Позволяет получить агрегированный статус всех компонентов одним запросом к `/api/gateway/v1/status/`.
  - Упрощает интеграцию с внешними системами мониторинга.
  - Новая возможность доступна сразу после обновления платформы.

Private Automation Hub

Внесены изменения в Private Automation Hub:

- Улучшена производительность API при работе с метаданными коллекций.
- Оптимизирована работа с container registry.

Automation Controller

Automation Controller получил значительные улучшения:

- Улучшена стабильность работы планировщика при переходе на летнее/зимнее время.
- Оптимизирована производительность при работе с большими шаблонами заданий.

Event-Driven Automation

Event-Driven Automation (EDA) получил значительные архитектурные улучшения:

- **Кластерный вариант** – полностью переработана архитектура для поддержки объединения в кластер при развертывании на BM.
- **Горизонтальное масштабирование** – поддержка множественных рабочих узлов Kubernetes:
  - Предназначено для администраторов высоконагруженных компонентов EDA.
  - Позволяет распределять обработку событий между несколькими узлами.
  - Повышает производительность обработки событий.
  - Улучшение доступно сразу после обновления EDA.
- **Высокая доступность** – автоматический failover при сбоях:
  - Предназначено для критически важных компонентов EDA.
  - Обеспечивает непрерывную обработку событий при отказе узлов.
  - Автоматически перераспределяет нагрузку между оставшимися узлами.
  - Улучшение доступно сразу после обновления EDA.
- **Поддержка дополнительных источников событий** – добавлена поддержка дополнительных типов источников событий.

Коллекции Ansible

В наборы коллекций, распространяемые через Automation Hub[346], внесены существенные изменения.

Разработаны новые коллекции:

- `astra.acm` – роли для автоматизации развертывания ACM (Astra Configuration Manager);
- `astra.vmmanager` – модули для базового управления виртуальными ресурсами;
- `orion.zvirt` – коллекция для управления инфраструктурой виртуализации zVirt.

Обновлены существующие коллекции:

- `astra.aa_controller`:
  - добавлена поддержка Astra Automation 2.0;
  - добавлены переменные для управления детализацией вывода `aa-setup`.
- `astra.ald_pro`:
  - добавлена поддержка новых версий ALD Pro;
  - добавлена опция `filter_nested_groups` в inventory plugin для управления поведением фильтрации вложенных групп;
  - добавлен LDAP inventory plugin.
- `astra.astralinux`:
  - добавлены модули `postgresql_mac_control`, `pdac_adm`.

[346] https://hub.astra-automation.ru/ui/

- `astra.ceph`:
  - добавлена поддержка Ceph 17 (Quincy) из состава ALSE 1.8.3.UU1.
- `astra.chrony`:
  - актуализирован список поддерживаемых операционных систем.
- `astra.dcimanager`:
  - добавлена поддержка DCI Manager 6;
  - добавлена поддержка Astra Linux Special Edition 1.8.1;
  - добавлена возможность установки в сегменте, изолированном от интернета.
- `astra.dhcp`:
  - актуализирован список поддерживаемых операционных систем.
- `astra.hardening`:

  Значительно улучшена роль `fstec`:
  - добавлены preflight-проверки;
  - добавлена установка пакетов, необходимых для управления функциями безопасности;
  - добавлена возможность определения собственного списка правил безопасности;
  - добавлены правила `login_13`, `digsig_1`;
  - внесены изменения в документации.
- `astra.iscsi`:
  - актуализирован список поддерживаемых операционных систем;
  - удалена временная поддержка неправильно написанных переменных `iscsi_inititator_dev_list`, `iscsi_inititator_wwid_list`.

    Примечание

    Последнее изменение выполнено без обеспечения обратной совместимости.
- `astra.nfs`:
  - актуализирован список поддерживаемых операционных систем.
- `astra.postgresql`:
  - актуализирован список поддерживаемых операционных систем.
- `astra.rupost`:
  - добавлена поддержка RuPost 3.4.0.
- `astra.termidesk`:
  - добавлена поддержка Termidesk 6.0.2 и ALSE 1.8.2.UU1;
  - прекращена поддержка Termidesk версий 6.0.0 и более ранних.

    Примечание

    Последнее изменение выполнено без сохранения обратной совместимости.

Утилиты разработчика

Следующие улучшения предназначены для разработчиков сценариев автоматизации и коллекций Ansible. Доступны сразу после обновления утилит из AA CDK:

`ansible-lint`:

- Улучшена совместимость с Ansible Core 2.18.
- Добавлены новые правила проверки безопасности playbooks.
- Оптимизирована скорость проверки больших проектов.

`ansible-navigator`:

- Улучшена работа с execution environments.

`ansible-creator`:

- Добавлен флаг `--no-overwrite` для предотвращения перезаписи существующих файлов. Защищает от случайной перезаписи важных файлов при генерации шаблонов.
- Улучшена проверка допустимости названий коллекций при их создании.

Образы служебных контейнеров

Выполнено обновление образов служебных контейнеров.

Общие обновления для всех образов:

- Ansible Core обновлен до версии 2.18.3.

Execution Environment `aa-full-ee`:

- Операционная система Astra Linux Special Edition обновлена до версии 1.8.3.
- Добавлена зависимость коллекции `astra.ald_pro` от модуля `python3-ldap3`.
- Исправлено название переменной для указания приватного реестра в скрипте выгрузки коллекций.

Decision Environment `aa-minimal-de`:

- Обновлен openjdk до версии 17.0.14.

Content Development Kit `aa-cdk`:

- Операционная система Astra Linux Special Edition обновлена до версии 1.8.3.
- Обновлен `podman` до версии 5.4.0.
- Обновлен `openjdk` до версии 17.0.14.

Измененные функции

При миграции на новую версию происходят следующие функциональные изменения:

- Изменяется процесс аутентификации пользователей и внешних приложений. Необходимо использовать единый интерфейс для создания токенов и настройки на внешние источников аутентификации (SAML, LDAP, RADIUS).
- Доступ к компонентам платформы происходит через единый интерфейс, что потребует изменения внешних приложений, интегрированных с Astra Automation.

Устаревшие сущности

При миграции на новую версию станут недействительными следующие сущности:

- Токены для Automation Controller и Private Automation Hub. Для генерации токенов используйте единый интерфейс.
- Все внешние механизмы аутентификации Automation Controller и Private Automation Hub станут недействительными. Необходимо перейти на настройку через единый интерфейс.

Устранение уязвимостей

В версии 2.0 устранены критические уязвимости безопасности:

- В утилите `aa-setup` устранена критическая уязвимость в модуле `virtualenv` (CVE-2024-53899[347], CVSS 9.8), позволяющая выполнение произвольного кода, путем использования другой версии модуля.
- В утилите `aa-setup` устранена уязвимость в модуле `certifi` (CVE-2024-39689[348], CVSS 7.5), связанная с недостаточной проверкой сертификатов, путем использования другой версии модуля. Процесс `failure-event-handler` в Automation Controller теперь запускается с привилегиями пользователя `awx` вместо `root`, что снижает потенциальные риски безопасности.

Исправления ошибок

Ниже перечислены наиболее значимые исправления.

Утилита aa-setup

- Исправлены критические проблемы с резервным копированием и восстановлением компонентов Redis и EDA. Улучшена логика остановки и запуска сервисов при выполнении операций `backup` и `restore`. Теперь корректно учитываются состояния рабочих процессов EDA для их автоматического возобновления после восстановления.
- Улучшена логика остановки и отключения рабочих процессов EDA во время установки и обновления.
- Исправлена установка платформы в изолированном сегменте сети из пакета offline bundle.

Automation Controller

- Исправлена ошибка, приводившая к некорректному подсчету хостов в инвентарях с большим количеством хостов.
- Устранены проблемы с кириллическими символами в файлах локализации интерфейса.
- Устранена утечка памяти при длительной работе служб контроллера.
- Устранены race conditions при параллельном запуске заданий.

Private Automation Hub

- Исправлены проблемы с синхронизацией коллекций большого размера.
- Устранены ошибки при импорте коллекций с большим количеством зависимостей.

Event-Driven Automation

- Исправлены проблемы с обработкой webhook’ов под высокой нагрузкой.
- Повышена стабильность при работе с большими rulebooks.

Коллекции Ansible

В коллекциях исправлены следующие проблемы:

- `astra.ald_pro`:
  - устранена уязвимость CVE-2025-4404.
- `astra.astralinux`:

[347] https://nvd.nist.gov/vuln/detail/CVE-2024-53899
[348] https://nvd.nist.gov/vuln/detail/CVE-2024-39689

  - исправлена документация модуля pdpl_user.

- astra.dhcp:
  - добавлено условие срабатывания обработчика (handler);
  - исправлен тип данных для параметра mode.
- astra.hardening:
  - улучшена логика обновления конфигурации для изменения параметров на месте вместо добавления их в конец файлов для auth_login_1 - auth_login_9 и auth_login_14;
  - усовершенствована логика аудита и исправлений для корректной идентификации и обновления только релевантных параметров для auth_login_10;
  - аудит и исправление обновлены для применения изменений в XML-конфигурацию только при наличии пакета fly-dm для limit_policy_1;
  - обработано наличие LibreOffice и исправлено поведение аудита и исправлений для статуса макросов в limit_safepolicy_6;
  - исправлены аудит и исправление для проверки и применения политики на основе уровня защищенности ФСТЭК вместо жестко заданного значения для auth_lockuser_1;
  - исправлен файл аудита для отчета о статусе соответствия и исправление для применения корректировок на основе уровня защищенности ФСТЭК для auth_passwdcomplexity_1;
  - задачи аудита обновлены для получения статуса пакета fly-dm и выполнения задач аудита для требований соответствия ФСТЭК limit_safepolicy_4;
  - исправлены ссылки на README, добавлены метаданные к ролям;
  - исправлена проверка и настройка параметров sysctl в основных settings_1;
  - исправлена логика аудита user_wireless_1;
  - исправлена логика аудита user_block_2;
  - исправлена логика аудита user_block_3;
  - удалено пустое правило lockuser_1;
  - исправлена логика аудита integrity_shedule_2;
  - исправлена логика аудита и исправления integrity_shedule_1;
  - исправлена логика аудита rotate_1;
  - исправлена логика аудита и исправления rotate_2, rotate_5;
  - исправлена логика аудита user_noauth_1;
  - исправлена логика аудита и исправления user_noauth_2;
  - добавлен шаблон для syslog-ng в audit_notify_1;
  - исправлена логика аудита в user_noauth_3, user_mic_2, user_mic_3, user_mic_4, user_maxlogins_1, user_mac_1, user_mac_4, user_mac_5, user_filter_1, user_devices_1, user_devices_3;
  - удалены исключенные правила из fstec_orders;
  - исправлены опечатки в названиях задач;
  - исправлена логика аудита auth_lockuser_4;
  - исправлена генерация названия отчета;
  - исправлена логика user_wireless_2;
  - исправлена логика limit_policy_1``;
  - исправлена логика auth_login_10;

  - добавлено правило auth_sudo_1 в список отключенных правил;
  - исправлена логика limit_safepolicy_9;
  - удалена задача обновления операционных систем.
- astra.nfs:
  - добавлено ожидание готовности сервера.

Утилиты разработчика

ansible-lint:

- Исправлена проблема с отображением цветов в терминале с черным фоном.

ansible-navigator:

- Исправлена проблема с отображением цветов в терминале с черным фоном.

ansible-creator:

- Добавлен флаг --no-overwrite для предотвращения перезаписи существующих файлов.
- Улучшена проверка допустимости имен коллекций при создании.
- Исправлены проблемы с генерацией шаблонов для различных типов проектов.

ansible-builder:

- Исправлены проблемы с отображением длинных строк в выходных данных команд.

Обновление на версию 2.0

Возможны следующие варианты перехода на новую версию с предыдущей версии 1.2:

- **Side-by-side migration** – *параллельная миграция*, обеспечивающая обновление версии с переносом компонентов платформы на другие узлы:
  - минимальный простой (минуты) для переключения на новую версию;
  - при необходимости – быстрый откат;
  - позволяет полностью протестировать новую версию перед переводом рабочей нагрузки на нее;
  - требуется временное удвоение ресурсов инфраструктуры.
- **Миграция в Kubernetes** – современный вариант, обеспечивающий масштабируемость, высокую доступность и автоматическое восстановление при сбоях.

  Требует наибольших инвестиций и квалификации команды, но обеспечивает долгосрочную готовность к будущим обновлениям платформы и эффективность использования ресурсов.

Критические изменения

Предупреждение

Версия 2.0 содержит существенные архитектурные изменения, которые в зависимости от текущих настроек требуют после обновления на новую версию внесения изменений с помощью ручных операций. Невыполнение указанных действий приведет к неработоспособной системе.

Ограничения автоматической миграции прав доступа

Механизм автоматической миграции имеет существенные ограничения, связанные с изменениями в модели RBAC версии 2.0.

Тип прав доступа	Автоматическая миграция	Требуемые действия
Роли пользователей на уровне организаций (Admin, Auditor, Member)	✅ Да	Не требуются
Права групп/команд на любые объекты	❌ **Нет**	**Ручная перенастройка**
Права пользователей на проекты	❌ Нет	Ручная перенастройка
Права пользователей на инвентари	❌ Нет	Ручная перенастройка
Права пользователей на шаблоны заданий	❌ Нет	Ручная перенастройка
Права пользователей на учетные данные	❌ Нет	Ручная перенастройка

Примечание

Рекомендации:

- До обновления задокументируйте все привилегии, назначенные на критически важных пользователей и команды пользователей.
- Запланируйте время на восстановление привилегий после миграции.
- Подробная инструкция по миграции привилегий доступна в разделе *Обновление на версию 2.0*.

Перенастройка внешней аутентификации

Управление внешними источниками аутентификации полностью перенесено в Platform Gateway.

Следующие настройки требуют изменений:

- LDAP/Active Directory – параметры подключения и ассоциация групп (mapping);
- SAML 2.0 – параметры IdP и атрибуты пользователей;
- RADIUS – параметры подключения;
- OAuth 2.0 – параметры провайдеров.

Последствия:

- Связи пользователей с группами LDAP будут утеряны до перенастройки аутентификации в Platform Gateway.
- Single Sign-On не будет работать до завершения настройки.
- Пользователи не смогут войти через внешние источники до перенастройки.

После корректной настройки внешней аутентификации в Platform Gateway членство пользователей в командах восстановится автоматически при следующем входе их в систему.

Изменения в API

Все URI точек доступа в API изменяются в связи с введением Platform Gateway как единой точки входа.

Компонент	Путь в версии 1.2	Путь в версии 2.0
Automation Controller	`/api/v2/...`	`/api/controller/v2/...`
Private Automation Hub	`/api/galaxy/...`	`/api/galaxy/...`
Event-Driven Automation	`/api/eda/...`	`/api/eda/...`
Platform Gateway (новое)	Не существовало	`/api/gateway/v1/...`

Недействительность токенов:

- Все токены API, созданные в версии 1.2 для Controller, Hub или EDA, становятся **недействительными**.
- Требуется генерация новых токенов через Platform Gateway API.
- Необходимо обновить все скрипты и настройки интегрированных систем для использования новых URI и токенов API.

Детальная информация о миграции API доступна в *описании миграции*.

В версии 2.0 сохранена возможность обращения к API компонентов платформы напрямую, минуя Platform Gateway. Это может потребоваться в следующих случаях:

- миграционный период при переходе с версии 1.2;
- специализированные интеграции, требующие прямого доступа к функциям бэкенда;
- legacy-приложения, которые не могут быть быстро адаптированы.

Предупреждение

При прямом доступе недоступны:

- централизованная аутентификация (SSO),
- единый RBAC,
- балансировка нагрузки через Gateway.

Использование прямого доступа не рекомендуется и должно рассматриваться как временная мера.

Отсутствие миграции Event-Driven Automation

В связи со значительными архитектурными улучшениями в EDA версии 2.0, автоматическая миграция данных **не поддерживается**. **Не мигрируют автоматически следующие ресурсы:**

- проекты EDA;
- правила (Rulebooks);
- источники событий;
- учетные данные EDA;
- история событий.

Перед обновлением необходимо экспортировать все настройки и rulebooks. После обновления потребуется пересоздать объекты вручную в новой версии EDA.

Изменения в документации

Произведены следующие наиболее существенные изменения в документации:

- Явно выделены основные фазы жизненного цикла продукта. Следуя лучшим мировым практикам, они обозначены как:
  - *Day 0* – планирование и подготовка к развертыванию платформы;

  - *Day 1* – развертывание платформы;
  - *Day 2* – техническое обслуживание.
- При описании платформы и в инструкциях учитываются модели развертывания – на виртуальных машинах и в кластере Kubernetes. Особенно явно это выделяется при описании упомянутых фаз жизненного цикла.

Контактная информация и поддержка

Для получения дополнительной информации, поиска решения проблем и обращения в службу поддержки используйте *официальные каналы Astra Automation*.

21.7.2 Версия 1.2

Дата выпуска: 14.02.2025
Тип выпуска: стабильная версия

Примечание

1. Все обновления представлены по отношению к версии 1.1-upd1, новейшей на момент выпуска данного обновления.
2. Если для баз данных Astra Automation вы используете или планируете использовать внешнюю СУБД, убедитесь, что она работает под управлением PostgreSQL версии 15.
3. Перед переходом на новую версию ознакомьтесь также с *известными проблемами*.

Главное

Следующие обновления представляются наиболее важными:

- В состав платформы Astra Automation добавлен новый компонент – *контроллер Event-Driven Automation (EDA)*, доступный как *техническая предварительная версия*.
- *Automation Controller* теперь базируется на AWX версии 24.6.1.
- Включен режим лицензирования.
- СУБД PostrgeSQL обновлена до версии 15.
- Python в узлах платформы обновлен до версии 3.11.
- Добавлена возможность развертывания платформы на операционной системе Astra Linux Special Edition версий 1.7.6 и 1.8.1.UU1.
- Прекращена техническая поддержка развертывания платформы на Astra Linux Special Edition версий 1.7.4.
- Все утилиты CDK переведены из состояния *technical preview* в состояние полноценной технической поддержки.

Новые возможности и улучшения

Новая версия содержит несколько новых возможностей и улучшений существующих характеристик.

Утилита aa-setup

Утилита `aa-setup` использует по умолчанию новые образы служебных контейнеров:

- Обновление предназначено для администраторов платформы.
- Позволяет использовать расширенный список коллекций Ansible, встроенных в новые образы.

- По умолчанию среды исполнения (EE, Execution Environment) Control Plane и Default в Automation Controller базируются на новых образах контейнеров.
- Изменения вступают в действие сразу в *процессе обновления платформы* или после прямой установки пакета развертывания платформы при подготовке установочного узла.

Обновление образов служебных контейнеров

Пересмотрен список и состав образов контейнеров, используемых в платформе. Образы собраны заново с добавлением множества компонентов и с использованием Python 3.11 из состава Astra Linux Special Edition 1.8.

Универсальный образ

Добавлен новый образ `aa-minimal-ee`, который является исходным образом для всех последующих и содержит минимальный набор утилит:

- Предназначен для разработчиков собственных образов служебных контейнеров и пользователей базовых возможностей Ansible.
- Максимально оптимизирован для создания собственных образов служебных контейнеров с необходимым набором коллекций и модулей Python.
- Доступен на Automation Hub[349].

Среда исполнения

Образ среды исполнения `aa-base-ee` заменен на следующие образы:

- `aa-control-ee` – образ для Control Plane в Automation Controller:
  - Предназначен для служебных задач Automation Controller.
  - Позволяет формировать инвентарные списки на основе данных из сторонних систем виртуализации.
  - Доступен на Automation Hub.
- `aa-full-ee` – образ EE по умолчанию в Automation Controller и Ansible Navigator:
  - Предназначен для пользователей Automation Controller и Ansible Navigator.
  - Позволяет выполнять сценарии автоматизации, использующие коллекции Astra Automation, без необходимости загружать эти коллекции.
  - Доступен на Automation Hub.

Среда принятия решений

Контроллер EDA требует использования специальный контейнеров для среды принятия решений (DE, Decision Environment). Понятие Decision Environment для Event-Driven Automation аналогично понятию Execution Environment для Automation Controller.

Добавлен образ `aa-minimal-de` с минимальными возможностями:

- Предназначен для разработчиков сводов правил (rulebooks) и собственных образов DE для них.
- Позволяет собирать собственные образы с необходимым набором коллекций и модулей Python.
- Доступен на Automation Hub.

Добавлен образ `aa-full-de` с расширенными возможностями:

- Предназначен для пользователей контроллера EDA.

[349] https://hub.astra-automation.ru/

- Позволяет использовать своды правил (rulebooks) со стандартными источниками событий (event sources).
- Доступен на Automation Hub.

Контейнер для разработчика

Образ `aa-creator-ee` заменен на `aa-cdk`:

- Предназначен для разработчиков инфраструктурного кода Ansible.
- Интегрируется с Visual Studio Code в качестве Dev Container и позволяет вести разработку и тестирование контента Ansible (сценарии, коллекции) с использованием инструментов из состава Astra Automation CDK.
- Доступен на Automation Hub.

> **Примечание**
>
> Образ `aa-cdk` не предназначен для EE или DE.

Обновление средств разработки

Утилита `ansible-navigator` теперь позволяет отключать передачу ключей и настроек SSH пользователя из рабочей станции в EE:

- Предназначено для пользователей Ansible Navigator.
- Обеспечивает гибкость в выборе способов предоставления полномочий для EE путем включения или отключения передачи ключей и настроек с помощью следующих аргументов:
  - `--ssh-forward-keys`;
  - `--ssh-forward-config`;
  - `--ssh-forward-agent`.
- Новая возможность доступна сразу после обновления CDK.

Обновления в Automation Controller

Изменения в работе служб контроллера:

- Добавлена возможность ограничивать доступы к Automation Controller API для неавторизованных пользователей:
  - Предназначено для администраторов системы.
  - Позволяет изменить конфигурации доступов к API для различных URL.
  - В разделе **Settings > Authentication** графического интерфейса Automation Controller добавлена секция **Allowed URLs for Anonymous Access** для указания списка URL, доступных для неавторизованных пользователей.
  - Новая возможность доступна сразу после обновления контроллера.
- Добавлена возможность определять состояние исполняющего узла:
  - Предназначена для команд devops, интегрирующих Astra Automation с различными системами.
  - Позволяет отслеживать состояние исполняющегося узла для своевременного обнаружения отказа.
  - В Automation Controller API добавлен параметр `status` в структуру данных, возвращаемых по запросу `/api/v2/metrics/`.
  - Новая возможность доступна сразу после обновления контроллера.

- Добавлены новые уровни регистрации информации в журналах:
  - Предназначено для администраторов системы.
  - Расширяет спектр доступной информации о системе.
  - Добавлена поддержка уровня регистрации ERROR для системы анализа и поиска данных OpenSearch.
  - Новая возможность доступна сразу после обновления контроллера.

Обновления в Private Automation Hub

Изменения в работе служб Private Automation Hub:

- Добавлена подсказка о символах, разрешенных в названиях пространства имен:
  - Предназначено для администраторов, создающих и настраивающих пространства имен.
  - Ускоряет процесс создания пространств, минимизируя возможность ввода недопустимых символов в названии.
  - Добавлена подсказка о допустимых символах при заполнении поля «Название» в окне создания пространства имен.
  - Новая возможность доступна сразу после обновления Private Automation Hub.
- Добавлена поддержка создания резервной копии хранилища S3:
  - Предназначена для администраторов и службы технической поддержки.
  - Повышает надежность хранения данных.
  - Обеспечена возможность клонирования данных S3 в файлы резервной копии.
  - Новая возможность доступна сразу после обновления Private Automation Hub.

Обновление коллекций Ansible

В коллекции, распространяемые через Automation Hub[350], добавлены существенные возможности:

- Обновлена сертифицированная коллекция `astra.ald_pro`:
  - Добавлена возможность развертывания всех подсистем домена.
  - Добавлены модули для управления доменом.
  - Добавлена поддержка ALD Pro 2.4.0.
  - Удалена поддержка развертывания предыдущих версий ALD Pro. Развертывание этих версий ALD Pro поддерживается в LTE-версии коллекции 1.0.x.
  - Добавлена поддержка клиентов Astra Linux Special Edition 1.8.1, 1.8.1.UU1.
- Обновлена коллекция `astra.aa_controller`:
  - Добавлена возможность развертывания EDA.
  - Добавлена возможность расширенного журналирования при установке.
- Обновлена коллекция `astra.astralinux`:
  - Добавлена возможность установки и обновления утилиты `astra-update`.
  - Добавлена поддержка снимков (snapshot) при обновлении.
- Обновлена коллекция `astra.freeipa`:
  - Добавлена возможность указания опций в файле `resolv.conf` клиента.
- Обновлена коллекция `astra.keycloak`:

[350] https://hub.astra-automation.ru/ui/

  - Добавлена совместимость с образом `aa-full-ee`.
- Обновлена коллекция `astra.termidesk`:
  - Изменен порт по умолчанию для VMmanager.
- Обновлена коллекция `astra.yandexcloud`:
  - Добавлены модули управления VPC (Virtual Private Cloud) и NLB (Network Load Balancer).

Исправление ошибок

В новой версии исправлены ошибки в поведении различных компонентов платформы.

Утилита развертывания платформы

Исправлены следующие ошибки в поведении утилиты `aa-setup`:

- Устранена ошибка, из-за которой утилита `aa-setup` игнорировала переменную `pg_password` после переопределения последней.
- Исправлено поведение, когда утилита `aa-setup` требовала указать переменную `ansible_host` даже при использовании FQDN.
- Устранена ошибка развертывания Automation Controller при попытке установки сертификата PostgreSQL.
- Устранена ошибка логики селектора выбора внешней или внутренней базы данных при развертывании платформы.

Automation Controller

Исправлены следующие ошибки в Automation Controller:

- Исправлена некорректная обработка сообщений, передаваемых из Automation Controller в `logstash-oss-with-opensearch-output-plugin`.
- Исправлен фильтр по организациям, выдававший ошибку в некоторых меню.
- Исправлена верстка у опции **Enable privilege escalation** в модальном окне **Run command**.
- Устранена ошибка, из-за которой предопределенные переменные шаблона не передавались в задание при использовании этого шаблона в Workflow.
- Внесены правки в текст пользовательского интерфейса Automation Controller на русском языке.
- Устранена ошибка, из-за которой переменные, определенные в шаблоне, не сохранялись при запуске задания.
- Устранена ошибка, из-за которой при создании обычного расписания задания появлялось сообщение об ошибке вида «Variables resource_types are not allowed on launch,,,».
- Исправлено поведение меню настроек **Settings > Jobs**, когда пункт **Extra Environment Variables** не был доступен в режиме редактирования.
- Устранены ошибки, из-за которых в исполнительном узле, установленном из пакета offline bundle, не запускались задания.
- Устранена ошибка, из-за которой не получалось зайти на исполняющий узел по SSH для создания резервной копии данных.
- Поправлены ссылки в пользовательском интерфейсе Automation Controller на раздел документации *Быстрый старт*.
- Внесены правки для оптимизации работы сервиса `receptor`.
- Устранена ошибка, из-за которой в Automation Controller не получалось загружать коллекции из Automation Hub, используя FQDN.

- Устранена ошибка, которая отображалась, если изменить количество ответвляемых процессов (forks) в настройках Automation Controller.

Коллекции Ansible

Исправлены следующие ошибки в коллекциях, опубликованных в Automation Hub:

- Исправлено поведение при обновлении сертификатов и названия переменных в коллекции `astra.rupost`.
- Исправлены модули `required_if` и `required_one_of` в коллекции `astra.termidesk`.

Обновление платформы до версии 1.2

Переход от одной из предыдущих версий к версии `1.2` необходимо производить, используя возможности утилиты `aa-setup` предыдущей версии:

Для перехода с версии `1.1-upd1` при установленном Private Automation Hub необходимо выполнить следующие шаги:

1. Подготовьте файл сценария, например, `write_version.yml`, для создания специального файла отслеживания версий на узлах Private Automation Hub:

```
---
- name: Write product version to /etc
  hosts: automationhub
  become: true
  tasks:
    - name: Write product version to /etc
      copy:
        content: "{{ product_version | default('1.1-upd1') }}"
        dest: "{{ product_version_filename | default('/etc/astra_automation.version
↪') }}"
        force: true
```

2. Выполните сценарий:

```
ansible-playbook -i /opt/rbta/aa/astra-automation-setup/inventory write_version.yml
```

3. Если необходимо установить контроллер EDA, добавьте описание *группы automationedacontroller* в инвентарный список (по умолчанию в файле `/opt/rbta/aa/astra-automation-setup/inventory`).
4. Выполните обычную процедуру обновления:

```
sudo ./aa-setup --upgrade
```

Если Private Automation Hub не установлен, то переход с версий `1.1` и `1.1-upd1` упрощается:

1. Для установки Private Automation Hub и контроллера EDA добавьте описание групп *automationhub* и *automationedacontroller* в инвентарный список (по умолчанию в `/opt/rbta/aa/astra-automation-setup/inventory`).
2. Выполните процедуру обновления:

```
sudo ./aa-setup --upgrade
```

Примечание

Если вы используете внешнюю СУБД под управлением PostgreSQL версии 13, то после обновления необходимо мигрировать базы данных Astra Automation на PostgreSQL версии 15.

Развертывание платформы на Astra Linux Special Edition 1.8

Для развертывания платформы на узлах с операционной системой Astra Linux Special Edition 1.8 необходимо явно указать URL пакета:

```
sudo ./aa-setup --repo-url https://dl.astralinux.ru/aa/aa-debs-for-alse-1.8
```

В остальном следуйте *обычной последовательности развертывания*.

Изменено в версии 1.2-upd1: Начиная с версии 1.2-upd1 использование аргумента *--repo-url* не является обязательным.

Известные проблемы

В версии 1.2 обнаружены следующие недостатки:

- Проблема безопасности.

  При развертывании Astra Automation с созданием сервера PostgreSQL средствами платформы вывод команды содержит пароль пользователя базы данных в открытом виде. Чтобы избежать этого, запустите команду развертывания с передачей дополнительной переменной:

  ```
  sudo ./aa-setup -- --extra-vars postgres_users_no_log=true
  ```

  Проблема проявляется при использовании минимальной или базовой топологий и не распространяется на топологию уровня предприятия.

  Изменено в версии 1.2-upd4: Проблема исправлена.

- В EDA не предусмотрена настройка на использование единой системы идентификации пользователей (SSO).
- В настройках контроллера EDA в инвентаре для утилиты `aa-setup` при использовании переменной `ansible_host` возникает ошибка вида `django.core.exceptions.DisallowedHost: Invalid HTTP_HOST header`. Необходимо использовать доменное имя (FQDN) или название, конвертируемое в IP-адрес системой наименований (naming service) узла, без применения переменной `ansible_host`.
- В Automation Controller из настроек переменных задания **Jobs settings > Extra Environment Variables** удалены переменные, настраиваемые по умолчанию. Ранее там была определена переменная `HOME: /var/lib/awx`, что приводило к прямому определению пути к домашнему каталогу учетной записи, используемой для выполнения заданий в среде исполнения. В текущей версии, если не восстановить эту переменную, домашним каталогом будет `/root/`, к которому будет подключен (монтирован) каталог исполняемого проекта. Это может привести к ошибкам, если в сценариях проекта используется каталог `/var/lib/awx/` напрямую. Для восстановление прежнего поведения необходимо определить переменную `HOME`, как это было в предыдущей версии контроллера.
- В Automation Controller может происходить зависание задания, запускаемого по расписанию, если в задании предусмотрена посылка уведомления в какую-либо систему оповещения, например, в Mattermost. Это происходит из-за совпадения двух факторов:
  - В настройках уведомления настроен параметр `Запустивший`, что для задания, запускаемого автоматически, не имеет смысла.
  - При обработке возникшего исключения возникает сбой, вызванный ошибкой в программном коде, которая будет исправлена в следующем обновлении.

  Исправить можно, убрав параметр `Запустивший`.

История обновлений стабильной версии

Следующая таблица содержит перечень обновлений в обратном хронологическом порядке и основные данные по ним.

Версия	Дата выпуска	Обновление характеристик (главное)	Исправления (главные)
1.2-upd4	12.09.2025	–	Исправлено поведение, когда пароль к базе данных сервера PostgreSQL, созданного через `aa-setup`, был доступен в открытом виде.
1.2-upd3	02.09.2025	Улучшены механизмы резервного копирования и восстановления.	Запуск одного из ключевых сервисов с привилегиями awx вместо root.
1.2-upd2	11.07.2025	Добавлена поддержка работы компонентов Astra Automation в режиме замкнутой программной среды (ЗПС). Разработано решение по оптимизации резервного копирования и восстановления данных.	Ошибки, связанные с обработкой сложных сценариев, требующих длительного времени.
1.2_upd1	28.03.2025	В графическом интерфейсе контроллера добавили ссылку на документ по активации подписки, необходимой после развертывания платформы. Упрощен перенос коллекций Ansible в приватный реестр для платформы, установленной а закрытом контуре без доступа к интернету.	Ошибки, связанные с некорректным восстановлением из резервной копии.
1.2	16.02.2025	Первый выпуск компонента Event-Driven Automation. Переход на новое семейство образов служебных контейнеров.	Устранены ошибки, из-за которых в исполнительном узле, установленном из пакета offline bundle, не запускались задания. Исправлено поведение, когда утилита `aa-setup` требовала указать переменную `ansible_host` даже при использовании FQDN в описании узла.

1.2-upd4

Дата выпуска: 12.09.2025
Тип выпуска: обновление стабильной версии

Состав платформы и совместимость с версиями операционной системы представлены в *таблице*.

Основное назначение данного обновления – исправление ошибок.

Главное

Исправлено поведение, когда пароль к базе данных сервера PostgreSQL, созданного через `aa-setup`, был доступен в открытом виде.

Исправление ошибок

В новой версии исправлена ошибка в утилите развертывания платформы `aa-setup`. При развертывании Astra Automation с созданием сервера PostgreSQL средствами платформы журнал и вывод команды содержали пароль пользователя базы данных в открытом виде.

В новой версии пароль не выводится в открытом виде.

Обновление платформы до версии 1.2-upd4

Для перехода от одной из предыдущих версий к версии `1.2-upd4` выполните следующую команду:

```
sudo ./aa-setup --upgrade
```

1.2-upd3

Дата выпуска: 02.09.2025
Тип выпуска: обновление стабильной версии

Состав платформы и совместимость с версиями операционной системы представлены в *таблице*.

Примечание

Перед переходом на новую версию ознакомьтесь также с *известными проблемами*.

Главное

Следующие обновления представляются наиболее важными:

- Расширена поддержка операционных систем – платформу можно развертывать на Astra Linux Special Edition 1.7.7.UU2 и 1.8.2.UU1.
- Улучшены механизмы резервного копирования и восстановления – исправлены критические ошибки при работе с поврежденными базами данных и восстановлении рабочих процессов (workers) после операций backup и restore.

Новые возможности и улучшения

Новая версия содержит ряд новых возможностей и улучшает существующие характеристики.

Утилита aa-setup

Утилита `aa-setup` обладает новыми возможностями:

- Улучшена процедура восстановления поврежденных баз данных:
  - Обновление предназначено для специалистов по техническому обслуживанию Astra Automation.

  - Ускорен процесс восстановления работоспособного состояния после операции restore.
  - Это достигнуто путем улучшения логики остановки и запуска сервисов при выполнении операции restore. Теперь учитываются состояния рабочих процессов – как active, так и inactive (но не disabled), чтобы они автоматически возобновляли работу после восстановления базы данных.
  - Доступно сразу после обновления платформы.
- В пакет платформы добавлены три файла инвентаря для ускорения типовой настройки:
  - Улучшение предназначено для специалистов по развертыванию и обслуживанию Astra Automation.
  - Ускоряет описание инвентаря за счет выбора наиболее подходящего исходного файла в зависимости от выбранной топологии.
  - Созданы три файла для рекомендуемых вариантов топологии:

    `inventory-minimal` – минимальная топология;

    `inventory-base` – базовая топология;

    `inventory-enterprise` – топология уровня предприятия.

  - Улучшение доступно сразу после обновления утилиты `aa-setup`.
- Процесс `failure-event-handler` теперь запускается с привилегиями пользователя `awx` вместо `root`, что делает систему менее уязвимой.

Образ среды исполнения aa-full-ee

Образ `aa-full-ee` включает обновленную утилиту *export_collections.py*:

- Улучшение предназначено для администраторов платформы.
- Упрощает операции по пополнению приватного реестра коллекциями из образа EE.
- Для этого обновленная утилита предоставляет две команды:
  - `list` выводит список коллекций, которые утилита готова загрузить в приватный реестр.
  - `publish` загружает указанные коллекции в заданный приватный реестр. Для автоматического создания необходимых пространств имен в приватном реестре при выполнении этой команды следует использовать аргумент `--autocreate`.
- Улучшение доступно сразу после обновления платформы.

Automation Controller

Удален образ `minimal-ee` из списка сред исполнения, доступных глобально для всех организаций:

- Улучшение предназначено для администраторов платформы.
- Исключает риски применения этого образа для избежания возникновения ошибок.
- Образ не пригоден для большинства проектов, так как в нем установлен минимальный набор программного обеспечения и он не содержит коллекций Ansible. Его применение с большой вероятностью приведет к ошибке выполнения заданий проекта.
- Улучшение доступно сразу после обновления платформы.

Коллекции

Произошли следующие изменения в коллекциях Ansible, доступных в Automation Hub[351].

- Коллекция `astra.aa_controller`:
  - Добавлена поддержка Astra Automation 1.2-upd3.
- Коллекция `astra.ald_pro`:
  - **Важное изменение** – добавлено новое поведение и новые состояния для модулей `user_group` и `computer_group`.
  - Добавлен новый атрибут `fqdn` в метод `get_group_items`.
  - Добавлены модульные тесты для модуля `computer`.
  - Добавлена поддержка косвенного членства в группах в inventory-плагине.
  - Исправлена проблема с кириллицей в `dn`.
  - Исправлено деление на страницы (pagination) в модуле `user_group`.
  - Исправлена ошибка работы с отсутствующими записями в модуле `computer`.
  - Исправлено некорректное условие при установке первого контроллера.
  - Исправлено поведение при редактировании несуществующего компьютера или группы. Теперь в таком случае обнаруживается ошибка и не фиксируются изменения в модуле `computer`.
  - Исправлено поведение при изменении групп по IP-адресу в переменной `name`. Теперь это работает согласно документации в модуле `computer`.
  - Исправлена поддержка *FQDN*. Домен больше не добавляется к полным именам, что предотвращает ошибки в модуле `computer`.
  - Исправлена обработка параметра `path`. Теперь в этом параметре можно задавать полные значения DN в модуле `computer`.
  - Исправлена обработка параметра `address_country`. Теперь выполняется проверка заданного значения и возвращается понятное описание ошибок при некорректных данных в модуле `computer`.
  - Исправлена обработка параметра `address_postal_code`. Теперь он принимает до 6 цифр в соответствии с ограничениями веб-интерфейса в модуле `computer`.
  - Исправлена обработка параметра `computer_mac_address`. Теперь значение проверяется на соответствие формату MAC-адреса и возвращается описание соответствующих ошибок в модуле `computer`.
  - Исправлена обработка названия группы пользователей в модуле: оно не должно содержать пробелов, кроме случаев `ald trust admin` и `trust admins`.
  - Исправлено удаление несуществующего пользователя из группы в модуле.
- Коллекция `astra.astralinux`:
  - Добавлены модули:
    * `admin_event_info`,
    * `admin_event`,
    * integration test for `admin_event`,
    * `int_check`,
    * `autologin_control`,
    * `lock`,
    * `noautonet_control`,

[351] https://hub.astra-automation.ru/

    * sudo_control,
    * nobootmenu_control,
    * mode_apps,
    * docker_isolation,
    * ilev1_control,
    * ufw_control,
    * ulimits_control,
    * secdel_control,
    * swapwiper_control,
    * rootloginssh_control.
  - Добавлены переменные окружения в модуль `migrate`.
  - Исправлено некорректное поведение при статусе `info` при вызове модулей не от `root`.
  - Отключено интерактивное подтверждение в `ufw_control`.
- Коллекция `astra.docker`:
  - Добавлена возможность выбора версии при установке пакетов.
- Коллекция `astra.freeipa`:
  - Исправлено поведение при указании репозитория для релизов **ALSE UU**.
- Коллекция `astra.iscsi`:
  - Добавлена поддержка Astra Linux Special Edition 1.8.2.UU1.
- Коллекция `astra.nfs`:
  - Добавлена поддержка Astra Linux Special Edition 1.8.2.UU1.
- Коллекция `astra.postgresql`:
  - Добавлена поддержка создания пользовательских схем в существующих базах данных.
  - Добавлено автоматическое создание схемы `public` для всех баз данных.
- Коллекция `astra.rupost`:
  - Добавлена поддержка Patroni.

Исправление ошибок

В новой версии исправлены следующие ошибки:

- Исправлена ошибка запуска рабочих процессов после восстановления Private Automation Hub из резервной копии.
- Исправлена ошибка миграции в базу данных с удаленными таблицами.

Обновление платформы до версии 1.2-upd3

Для перехода от одной из предыдущих версий к версии `1.2-upd3` выполните следующую команду:

```
sudo ./aa-setup --upgrade
```

Важно

Если вы установили версию `1.2-upd3` в операционной системе Astra Linux Special Edition 1.8.1.UU1 или 1.7.6.UU2, а затем обновили ОС до 1.8.2.UU1 или 1.7.7.UU2 соответственно, то необходимо повторно запустить утилиту `aa-setup` для адаптации к новой версии ОС.

Известные проблемы

В версии 1.2-upd3 обнаружены следующие недостатки:

- Проблема безопасности.

  При развертывании Astra Automation с созданием сервера PostgreSQL средствами платформы вывод команды содержит пароль пользователя базы данных в открытом виде. Чтобы избежать этого, запустите команду развертывания с передачей дополнительной переменной:

  ```
  sudo ./aa-setup -- --extra-vars postgres_users_no_log=true
  ```

  Проблема проявляется при использовании минимальной или базовой топологий и не распространяется на топологию уровня предприятия.

  Изменено в версии 1.2-upd4: Проблема исправлена.

1.2-upd2

Дата выпуска: 11.07.2025
Тип выпуска: обновление стабильной версии

Состав платформы и совместимость с версиями операционной системы представлены в *таблице*.

Примечание

Перед переходом на новую версию ознакомьтесь также с *известными проблемами*.

Главное

Следующие обновления представляются наиболее важными:

- Добавлена поддержка работы компонентов Astra Automation в режиме замкнутой программной среды (ЗПС).
- Добавлена возможность задания пароля для подключения к Automation Hub в процессе развертывания платформы.
- Разработано решение по оптимизации резервного копирования и восстановления данных.

Новые возможности и улучшения

Новая версия содержит ряд новых возможностей и улучшает существующие характеристики.

Утилита aa-setup

Утилита `aa-setup` обладает новыми возможностями:

- Добавлена возможность задать параметры для подключения к Automation Hub в процессе развертывания платформы:

  - Предназначена для специалистов по развертыванию и обслуживанию Astra Automation.
  - Позволяет утилите развертывания установить соединение с требуемым Automation Hub, используя заданную учетную запись, в процессе развертывания платформы.
  - В файле `inventory` добавлены параметры `registry_username`, `registry_password`, `registry_url` и `registry_verify_ssl`, которые обрабатываются утилитой `aa-setup`.
  - Новая возможность доступна сразу после установки утилиты `aa-setup` версии 1.3.13 в составе нового пакета `astra-automation-setup`.
- В файл описания инвентаря добавлены параметры для подключения контроллера EDA к базе данных:
  - Параметры предназначены для специалистов по развертыванию и обслуживанию Astra Automation.
  - Облегчают настройку контроллера EDA для подключения к СУБД, подсказывая названия соответствующих параметров.
  - Файл `inventory` теперь содержит параметры `automationedacontroller_pg_host` (адрес СУБД) и `automationedacontroller_pg_port` (порт TCP) для подключения к СУБД. Соответствующие строки помечены знаком комментария.
  - Изменения доступны сразу после установки утилиты `aa-setup` версии 1.3.13 в составе нового пакета `astra-automation-setup`.
- Улучшен механизм резервного копирования и восстановления:
  - Обновление предназначено для специалистов по техническому обслуживанию Astra Automation.
  - Создание резервных данных и восстановление платформы ускорилось не менее, чем в 2 раза.
  - Для этой цели изменены алгоритмы выполнения команд `backup` и `restore`.
  - Новая возможность доступна сразу после установки утилиты `aa-setup` версии 1.3.13 в составе нового пакета `astra-automation-setup`.
- Изменен способ предоставления образов среды исполнения при развертывании платформы в закрытом контуре. Утилита не устанавливает промежуточный локальный реестр образов. Образы среды исполнения и принятия решений загружаются сразу в приватный реестр платформы. В связи с этим требования к описанию инвентаря изменились:
  - Переменные `registry_ip` и `registry_port` не должны более использоваться.
  - В секции `[all:vars]`, обязательно должны присутствовать реквизиты пользователя с привилегиями администратора Private Automation Hub, например:

```
[all:vars]
automationhub_admin_user='admin'
automationhub_admin_password='notsecurepass'
```

CDK

Внесены изменения в утилитах для разработчиков:

- Автоматический запуск `apt-update` утилитой `ansible-builder` перед установкой системных пакетов:
  - Предназначено для разработчиков среды исполнения (EE, Execution Environment).
  - Обеспечивается доступ к новейшим или явно указанным версиям пакетов, заданных в файле зависимостей `bindep.txt`, перед их установкой.

  - Утилита `ansible-builder` автоматически вызывает утилиту `apt-update` для обновления индекса пакетов перед установкой требуемых системных пакетов.
  - Новая возможность доступна сразу после установки или обновления утилиты `ansible-builder` версии 3.1.0+aa1.1.1.
- Реализована новая цветовая схема интерактивного интерфейса (TUI, Text User Interface) утилиты `ansible-navigator`.
  - Предназначена для пользователей, разрабатывающих и тестирующих инфраструктурный код с помощью утилиты `ansible-navigator`.
  - Новая цветовая гамма снижает нагрузку на зрение и делает работу с приложением комфортнее для наших пользователей.
  - Для этого изменена цветовая схема в настройках приложения.
  - Новая цветовая схема доступна сразу после установки `ansible-navigator` версии 24.9.0+aa1.1.2.

Automation Controller

В графическом интерфейсе Automation Controller добавлена инструкция по активации лицензии при первоначальной загрузке контроллера после его развертывания:

- Предназначена для администраторов платформы.
- Ускоряет ввод контроллера в эксплуатацию.
- Страница с описанием требуемых шагов открывается сразу после аутентификации администратора при входе в графический интерфейс.
- Инструкция доступна сразу после обновления контроллера.

Обновления среды исполнения

Внесены следующие изменения в образы среды исполнения:

- Теперь образы среды исполнения в реестре Automation Hub содержат подписанные версии утилит Astra Automation:
  - Обновление предназначено для разработчиков среды исполнения и администраторов платформы.
  - Направлено на повышение уровня безопасности платформы в целом и позволяет включить режим замкнутой программной среды (ЗПС) на узлах контроллера и исполняющих узлах.
  - Все программные компоненты образов подписаны с помощью доверенных ключей.
  - Данные образы доступны в реестре Automation Hub и используются в новой версии контроллера.
- В образ `aa-minimal-ee` добавлена утилита `less`:
  - Обновление предназначено для разработчиков инфраструктурного кода.
  - Утилита `less` необходима для просмотра документации по расширениям и модулям Ansible с помощью утилиты `ansible-navigator`.
  - Утилита встроена в этот базовый образ и будет присутствовать во всех остальных образах, построенных на нем.
  - Образ доступен в реестре Automation Hub.

Коллекции

Внесены следующие изменения в коллекции Ansible, опубликованные в Automation Hub:

- В описание каждой коллекции (вкладка **Documentation**) добавлен пример спецификации `execution-environment.yml` для сборки образа среды исполнения с помощью `ansible-builder`.
- Коллекция `astra.aa_controller`:

  Новые возможности:

  - Добавлена поддержка Astra Automation 1.2-upd2.
- Коллекция `astra.ald_pro`:

  Новые возможности:

  - добавлена поддержка ALD Pro 2.5.0;
  - добавлен модуль `dc_info`;
  - добавлено расширение Ansible, автоматически формирующее описание инвентаря.

  Исправленные ошибки:

  - исправлено поведение модуля `user` при работе с большим количеством пользователей, превышающим объем одной страницы в интерфейсе ALD Pro;
  - изменение атрибутов пользователя теперь не приводит к сбросу пароля;
  - исправлены неточности документации по модулям;
  - исправлено кодирование URL для корректной обработки параметров запроса.
- Коллекция `astra.astralinux`:

  Новые возможности:

  - добавлены модуль `astra_migrate` и роль `astra_migrate` для миграции Astra Linux с версии 1.7 на 1.8;
  - добавлены модули для управления функциями безопасности Astra Linux:
    * `pdpl_user`,
    * `digsig_control`,
    * `mic`,
    * `mac`,
    * `mode_switch`,
    * `admin_event`,
    * `admin_event_info`.

  Исправленные ошибки:

  - внесены исправления в документацию модуля `update`.
- Коллекция `astra.hardening`:

  Новые возможности:

  - добавлена роль `fstec` (в качестве *технической предварительной версии*), позволяющая создавать профили безопасности и выполнять аудит операционной системы Astra Linux 1.8 в соответствии с приказами ФСТЭК 17, 21, 31, 239;
  - в роли devsec добавлена возможность аудита и проведен рефакторинг.
- Коллекция `astra.rubackup`:

  Новые возможности:

  - добавлена поддержка RuBackup 2.5;
  - добавлена возможность включения и настройки Tucana API.

- Коллекция `astra.termidesk`:

  Новые возможности:

  - добавлена поддержка Termidesk 5.1.1;
  - добавлена роль `aggregator`;
  - добавлен модуль `authenticator_group`;
  - в модулях добавлены опции таймаута и проверки сертификата.

- Коллекция `astra.yandexcloud`:

  Добавлены модули для управления следующими объектами и действиями:

  - кластерами баз данных;
  - кластерами Kubernetes;
  - балансировщиками нагрузки;
  - хранилищами данных;
  - операциями в облаке.

  Исправленные ошибки:

  - уменьшена минимально допустимая длина названия виртуальной машины с трех символов до двух.

Исправление ошибок

В новой версии исправлены следующие ошибки:

- Обработка сразу нескольких наборов сценариев, требующих длительного времени, приводила эпизодически к потере файлов или зависанию задач. Это поведение исправлено путем внесения изменений в алгоритме обработки сценариев.
- Исправлена визуализация количества используемых подписок на графике «Расход ресурсов подписки» в Automation Controller. Ранее в окне с поясняющим текстом происходило наложение строк. Это неудобство устранено.
- Выполнен перевод на русский язык текста в некоторых элементах общих настроек и управления полномочиями в Automation Controller.
- Ранее при обновлении версии платформы утилита `aa-setup` отображала курсор в строке запроса в неправильной позиции. Это поведение исправлено.
- В утилите `aa-setup` устранены уязвимости, вносимые модулями `virtualenv` (CVE-2024-53899[352]) и `certifi` (CVE-2024-39689[353]), путем использования других версий этих модулей.
- Восстановлена работоспособность компонента `syslog-ng`, используемого для ведения журналов контроллера. Она была утеряна при обновлении на версию платформы 1.2 с миграцией на операционную систему Astra Linux Special Edition 1.8.
- Выполнение команды `pulpcore-manager lock-repository-pull` для защиты от несанкционированного доступа к образам контейнеров на Private Automation Hub, установленного на Astra Linux Special Edition 1.8.1, было блокировано. Работа команды восстановлена.
- При использовании системного репозитория Astra Linux Special Edition 1.8 *stable* вместо *frozen* развертывание Automation Controller блокировалось на шаге установки Podman. Процедура развертывания исправлена для рассмотренной ситуации.
- При установке контроллера EDA на Astra Linux Special Edition 1.8.1.UU1 журналы активации не появлялись в графическом интерфейсе. Это поведение исправлено, журналы активации появляются в интерфейсе пользователя.

[352] https://nvd.nist.gov/vuln/detail/CVE-2024-53899
[353] https://nvd.nist.gov/vuln/detail/CVE-2024-39689

- Исправлена ошибка переопределения пути к сертификату CA через переменную `ca_trust_bundle` в описании инвентаря для утилиты `aa-setup`. После исправления утилита `aa-setup` корректно определяет и использует заданный путь к сертификату.
- Исправлена ошибка создания резервной копии платформы (backup) через утилиту `aa-setup` на установочном узле, когда учетная запись, указанная переменной `ansible_user`, отсутствует на этом узле. После внесенных изменений алгоритм работает следующим образом. Если учетная запись, указанная переменной `ansible_user`, отсутствует на установочном узле, место хранения резервной копии зависит от того, запущена ли утилита `aa-setup` с привилегиями root или от обычного пользователя:
  - При запуске с привилегиями root без указания каталога данные сохраняются в `/var/backups/astra-automation/`.
  - При запуске с привилегиями root с указанием каталога данные сохраняются в указанный каталог.
  - При запуске с привилегиями обычного пользователя без указания каталога данные сохраняются в домашнем каталоге пользователя.
  - При запуске с привилегиями обычного пользователя с указанием каталога данные сохраняются в указанном каталоге, если пользователь обладает необходимыми привилегиями. Иначе данные сохраняются в домашнем каталоге пользователя.
- Исправлена ошибка синхронизации EE в Private Automation Hub с реестром, у которого отсутствует сертификат TLS. Ранее, даже при отключенной проверке TLS (**TLS validation**), синхронизация завершалась сообщением об ошибке. Теперь отключение проверки работает корректно и синхронизация проходит успешно.

Обновление платформы до версии 1.2-upd2

Для перехода от одной из предыдущих версий к версии `1.2-upd2` выполните следующую команду:

```
sudo ./aa-setup --upgrade
```

Известные проблемы

В версии 1.2-upd2 обнаружены следующие недостатки:

- Проблема безопасности.

  При развертывании Astra Automation с созданием сервера PostgreSQL средствами платформы вывод команды содержит пароль пользователя базы данных в открытом виде. Чтобы избежать этого, запустите команду развертывания с передачей дополнительной переменной:

  ```
  sudo ./aa-setup -- --extra-vars postgres_users_no_log=true
  ```

  Проблема проявляется при использовании минимальной или базовой топологий и не распространяется на топологию уровня предприятия.

  Изменено в версии 1.2-upd4: Проблема исправлена.

1.2-upd1

Дата выпуска: 28.03.2025
Тип выпуска: обновление стабильной версии

Состав платформы и совместимость с версиями операционной системы представлены в *таблице*.

Основное назначение данного обновления – исправление ошибок, возникших после внесения существенных изменений в версии 1.2. Однако есть и некоторые функциональные обновления.

Примечание

Перед переходом на новую версию ознакомьтесь также с *известными проблемами*.

Главное

Следующие обновления представляются наиболее важными:

- В графическом интерфейсе контроллера добавили ссылку на документ по управлению подпиской, которую необходимо приобрести и активировать после развертывания платформы.
- Упрощен перенос коллекций Ansible в приватный реестр для платформы, установленной а закрытом контуре без доступа к интернету.
- В Private Automation Hub добавили управление защитой реестра образов контейнеров от несанкционированной выгрузки.

Новые возможности и улучшения

Новая версия содержит некоторые новые возможности и улучшение существующих характеристик.

Обновление образов контейнеров

Образ `aa-full-ee` предоставляет возможность переноса основных коллекций Ansible в приватный реестр:

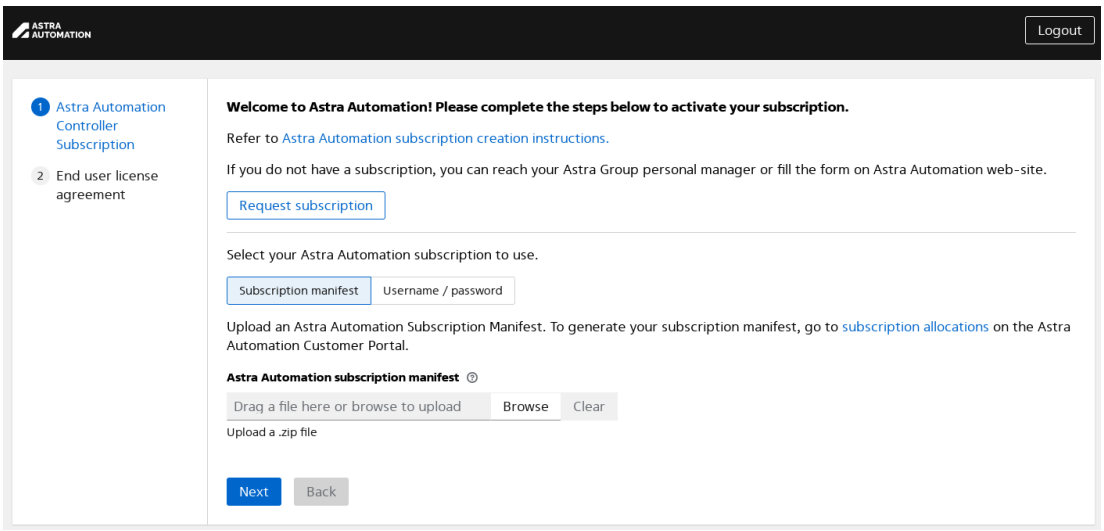
- Предназначено для администраторов платформы.
- Позволяет загрузить в приватный реестр все коллекции Ansible из пространства `astra` и другие коллекции, наиболее часто используемые в Astra Automation. Это особенно необходимо для настройки платформы в закрытом контуре без доступа к интернету.
- Для этого создан скрипт Python `export_collections.py` с *подробным описание его применения*.
- Образ доступен в реестре образов Automation Hub[354] для клиентов платформы.

Automation Controller

В графическом интерфейсе пользователя добавлена ссылка на документ, содержащий подробную информацию об управлении подпиской на Astra Automation, включая приобретение, активацию и эффективное управления лицензиями:

- Необходимо для администраторов контроллера.
- Облегчает процедуру настройки контроллера на нормальную работу после развертывания платформы. Кроме того, ознакомление с документом позволяет администратору эффективно управлять лицензиями, выделенными через подписку.
- При входе в графический интерфейс контроллера администратор видит уведомление о необходимости активации подписки и ссылку на инструкцию по *управлению подпиской*.
- Ссылка доступна сразу после обновления контроллера.

[354] https://hub.astra-automation.ru/ui/containers/aa-1.2/aa-full-ee/_content/images/



Automation Hub

В Private Automation Hub добавлено управление защитой API реестра образов контейнеров от несанкционированного доступа:

- Предназначено для администраторов Private Automation Hub и влияет на пользователей реестра образов.
- Позволяет защитить инфраструктурный код от несанкционированного использования с учетом того, что образы могут содержать коммерческую собственность, например `aa-full-ee` содержит все коллекции Ansible, предназначенные для клиентов Astra Automation. При включенной защите пользователь обязан пройти шаг аутентификации с помощью команды `podman login` или `docker login`. Только после этого ему будут доступны образы контейнеров.
- Для управления добавлена команда `lock-repository-pull` утилиты управления `pulpcore-manager`:
  - включение защиты:

    ```
    sudo pulpcore-manager lock-repository-pull
    ```

  - выключение защиты:

    ```
    sudo pulpcore-manager lock-repository-pull --unlock
    ```

- Функция доступна сразу после обновления платформы.

Исправление ошибок

В новой версии исправлены следующие ошибки:

- После переноса платформы с одной группы узлов на другую с помощью резервного копирования и восстановления при запуске задания из шаблона в Automation Controller оно зависало с выводом сообщения об ошибке в журнал `tower.log`. В новой версии внесены изменения, в результате которых такая миграция выполняется без отмеченной ошибки.
- После восстановления Private Automation Hub, установленного на операционной системе Astra Linux Special Edition 1.8.1.UU1, из резервной копии при попытке получить доступ к пользовательскому графическому интерфейсу возникала ошибка на стороне сервера с выдачей сообщения об ошибке «504 Gateway Time-out». В новой версии внесены изменения, в результате которых восстановление из резервной копии выполняется без отмеченной ошибки.
- При установке контроллера EDA с использованием его доменного имени (FQDN) в описании инвентаря в настройках контроллера не создавались некоторые переменные,

из-за чего пользователь не мог пройти аутентификацию при попытке войти в консоль графического интерфейса.

- Исправлена известная проблема, связанная с тем, что при развертывании платформы на операционной системе Astra Linux Special Edition 1.8 необходимо было явно указывать адрес пакета через аргумент `--repo-url`. При развертывании платформы новой версии можно использовать утилиту `aa-setup` без явного указания URL пакета.

Обновление платформы до версии 1.2-upd1

Переход от одной из предыдущих версий к версии `1.2-upd1` необходимо производить, используя один из следующих вариантов в зависимости от исходной версии:

- Для перехода с версии `1.2` на `1.2-upd1` выполните следующую команду на *установочном узле*:

```
sudo ./aa-setup --upgrade
```

- Для перехода с версии `1.1` на `1.2-upd1` следуйте инструкции по обновлению.

Известные проблемы

В версии 1.2-upd1 обнаружены следующие недостатки:

- Проблема безопасности.

  При развертывании Astra Automation с созданием сервера PostgreSQL средствами платформы вывод команды содержит пароль пользователя базы данных в открытом виде. Чтобы избежать этого, запустите команду развертывания с передачей дополнительной переменной:

```
sudo ./aa-setup -- --extra-vars postgres_users_no_log=true
```

  Проблема проявляется при использовании минимальной или базовой топологий и не распространяется на топологию уровня предприятия.

  Изменено в версии 1.2-upd4: Проблема исправлена.

21.7.3 Версия 1.1

Дата выпуска: 19.08.2024
Тип выпуска: стабильная версия

Примечание

Все обновления представлены по отношению к версии 1.0-upd2, новейшей на момент выпуска данного обновления.

Состав платформы и совместимость с версиями операционной системы представлены в *таблице*.

Главное

Следующие обновления представляются наиболее важными:

- Наиболее важным является добавление утилит *Ansible Navigator* и *Ansible Builder*. Они являются частью компонента CDK и распространяются в виде deb-пакетов в составе репозитория Astra Automation.
- Обеспечена возможность развертывания платформы без доступа к интернету.
- Выпущен новый образ среды исполнения (EE) `aa-creator-ee` для разработки контента.

- Обновлен образ EE, используемый в контроллере по умолчанию.
- Добавлен переключатель языка графического интерфейса контроллера.

Новые возможности и улучшения

Следующие секции содержат описание новых возможностей и улучшений существующих характеристик.

Утилита aa-setup

Выпущена версия `1.1.0` утилиты, доступная в виде пакета `astra-automation-setup`. Она содержит новые возможности:

- Добавлена возможность развертывания платформы без доступа к интернету:
  - Предназначено для специалистов по установке и обслуживанию Astra Automation.
  - Позволяет развернуть платформу без доступа к интернету с сохранением всех ее функций.
  - Для этого произведены следующие изменения:
    * Создан архивированный пакет для автономной установки (offline bundle), включающий все компоненты платформы.
    * Для файла инвентаря созданы дополнительный параметры для режима автономного развертывания.
  - Функция доступна сразу после обновления утилиты.
- Улучшена возможность подключения контроллера к единой системе аутентификации (SSO, single-sign-on):
  - Предназначено для специалистов по установке и обслуживанию Astra Automation.
  - Новая функция позволяет подключать контроллер к системе SSO в процессе развертывания платформы.
  - Настройку функции необходимо выполнять с помощью группы `sso` в файле инвентаря.
  - Функция доступна сразу после обновления утилиты.
- Добавлена возможность запуска `aa-setup` в неинтерактивном режиме:
  - Предназначено для специалистов по установке и обслуживанию Astra Automation.
  - Новый режим позволяет избежать вопросов при выполнении команды `upgrade`.
  - Для включения неинтерактивного режима добавлен флаг `-y`, `--force-yes`, что эквивалентно ответу Да (Yes) на все запросы по разрешению выполнения операций.
  - Данный режим доступен сразу после обновления пакета.
- Изменены некоторые параметры настройки инвентаря:
  - Предназначено для специалистов по установке и обслуживанию Astra Automation.
  - Изменения приводят названия параметров к единой системе и упрощают общую настройку инвентаря.
  - Проведены следующие изменения:
    * Параметр `awx_install_admin_password` переименован в `admin_password`.
    * При использовании внешней базы данных необходимо оставлять секцию `[database]` без параметров (пустая группа). Ранее в таком случае необходимо было ее удалять (комментировать).
  - Изменения вступают в действие сразу после обновления утилиты.

Обновления среды исполнения

Произведены следующие изменения в образах среды исполнения (EE):

- Создан новый образ `aa-creator-ee` версии 0.2.1:
  - Предназначен для разработчиков и тестировщиков коллекций и проектов Ansible.
  - Предоставляет удобную среду для разработки и тестирования, содержащую необходимые для этого компоненты.
  - Образ включает специальные утилиты и драйверы для тестирования:
    * `ansible-lint`,
    * `molecule`,
    * драйверы для `molecule` (azure, containers, default, docker, ec2, gce, molecule_brest, molecule_yandexcloud, openstack, podman, vagrant).
  - Доступен сразу после выпуска новой версии Astra Automation.
- Образ `aa-base-ee`, используемый в контроллере по умолчанию, обновлен до версии `0.6.2`:
  - Предназначен для администраторов и пользователей платформы.
  - Обеспечивает выполнение базовых операций Ansible с помощью встроенной базовой коллекции `ansible.builtin`. Рекомендуется использование этого образа в качестве базового при создании собственных EE.
  - Сделаны следующие изменения:
    * Удалены все прежние коллекции, за исключением `ansible.builtin`, `ansible.posix` и `ansible.utils`.
    * Удалена утилита `task`.
  - Доступен сразу после выпуска новой версии Astra Automation.

Обновления средств разработки

В состав средств разработки контента (CDK, content development kit) добавлены следующие компоненты:

- Первый выпуск пакета CDK (мета-пакет с необходимыми зависимостями), при установке которого происходит установка следующих утилит CDK:
  - `ansible-builder` версии не ниже 3.0.1,
  - `ansible-navigator` версии не ниже 24.2.0,
  - `ansible-lint` версии не ниже 6.22.2,
  - `molecule` версии не ниже 6.0.3.
- Утилита *Ansible Navigator*:
  - Предназначена для пользователей платформы, разработчиков и тестировщиков контента.
  - В командной строке вместо различных утилит `ansible*`, включая `ansible`, `ansible-inventory`, `ansible-playbook` и `ansible-doc`, следует вызывать утилиту `ansible-navigator` с соответствующей командой. Например, вместо `ansible-builder` следует использовать `ansible-navigator builder`, а вместо `ansible-playbook` – `ansible-navigator run`.
  - Разработан и протестирован пакет в формате deb `ansible-navigator`.
  - Утилита доступна сразу после выпуска новой версии Astra Automation. Для использования этого средства необходимо установить пакет с помощью менеджера пакетов apt и использовать согласно инструкции.
- Утилита *Ansible Builder*:

  - Предназначена для разработчиков и тестировщиков контента Ansible.
  - Используется для создания образов EE.
  - Разработан и протестирован пакет в формате deb `ansible-builder`.
  - Утилита доступна сразу после выпуска новой версии Astra Automation. Устанавливается при установке Ansible Navigator. Можно установить отдельно с помощью менеджера пакетов apt и использовать согласно инструкции.
- Утилита Ansible Lint:
  - Предназначена для разработчиков и тестировщиков контента Ansible.
  - Используется для проверки контента (сценарии, роли, коллекции) на соответствие установленным правилам оформления кода (style guide).
  - Утилита встроена в образ EE `aa-creator-ee`. Разработан пакет в формате deb `ansible-lint`.
  - Утилита доступна как *техническая предварительная версия* сразу после выпуска новой версии Astra Automation. Утилита устанавливается при установке Ansible Navigator. Можно также установить отдельно с помощью менеджера пакетов apt.

Внимание

Техническая предварительная версия (Technical preview).

- Утилита Ansible Molecule:
  - Предназначена для разработчиков и тестировщиков контента Ansible.
  - Используется для тестирования контента Ansible в различных операционных системах, виртуальных средах и облачных окружениях. Утилита может быть также использована в составе различных тестовых систем как часть CI-конвейера.
  - Утилита встроена в образ EE `aa-creator-ee`. Разработан пакет в формате deb `molecule`.
  - Утилита доступна как *техническая предварительная версия* сразу после выпуска новой версии Astra Automation. Можно установить с помощью менеджера пакетов `apt`.

Внимание

Техническая предварительная версия (Technical preview).

Обновления контента Ansible

Произведены следующие обновления контента, распределяемого через Automation Hub[355] для пользователей и разработчиков инфраструктурного кода:

- Коллекция `astra.rubackup` поддерживает версию RuBackup 2.0.U3.
- Коллекция `astra.rupost` поддерживает версию Rupost 2.7.1.

Добавлены новые коллекции в Automation Hub:

- `astra.chrony`
- `astra.patroni`
- `astra.pgbouncer`
- `astra.etcd`
- `astra.keepalived`

[355] https://hub.astra-automation.ru/

- `astra.yandexcloud`
- `astra.hardening`
- `astra.astralinux`

Обновления в Automation Controller

Следующие изменения касаются графического web-интерфейса контроллера:

- Добавлен *глобальный фильтр* для выбора ресурсов по принадлежности их к заданному списку организаций:
  - Предназначен для администраторов и пользователей контроллера.
  - Ускоряет поиск требуемых ресурсов.
  - Добавлена кнопка в правом верхнем углу экрана графического интерфейса, при нажатии на которую появляется выпадающая панель, позволяющая выбрать несколько организаций, которые интересуют пользователя. После этого выбора пользователь увидит только те ресурсы (задания, инвентарь, проекты и шаблоны заданий), которые принадлежат выбранным организациям.
  - Доступно сразу после обновления контроллера.
- Добавлена возможность немедленного запуска задания из планировщика:
  - Предназначено для администраторов и пользователей контроллера.
  - Позволяет сразу запустить задание, не дожидаясь его автоматического запуска в запланированное время.
  - В планировщике заданий добавлена кнопка запуска.
  - Доступно сразу после обновления контроллера.
- В настройках *службы периодической очистки заданий* период хранения информации о выполненных заданиях определяется отдельно для каждого типа задания:
  - Предназначен для администраторов контроллера.
  - Позволяет настраивать с помощью графического интерфейса разную продолжительность хранения информации о выполненных заданиях разного типа (ad-hoc commands, project updates, inventory updates, management jobs, notifications, workflow jobs).
  - Преимущество достигнуто путем добавления дополнительных полей в настройках служебного задания на очистку. Путь навигации: **Dashboard > Schedules > Cleanup Job Schedule**.
  - Настройка доступна сразу после обновления контроллера.
- Ссылка **Помощь** ведет адресно на соответствующее описание в документации:
  - Предназначено для администраторов и пользователей контроллера.
  - Ускоряет поиск необходимой информации.
  - Ссылка **Помощь** (Help) в правом верхние углу экрана ведет на конкретный документ или секцию, описывающие текущее окно графического интерфейса. Ранее ссылка была на общий раздел описания графического интерфейса.
  - Доступно сразу после обновления контроллера.
- Добавлен переключатель языка интерфейса:
  - Предназначено для администраторов и пользователей контроллера.
  - Позволяет оперативно переключиться на требуемый язык общения с контроллером. Ранее для переключения на требуемый язык необходимо было настраивать браузер.
  - Переключатель добавлен в правом верхнем углу экрана.

  - Доступно сразу после обновления контроллера.

Изменения в работе служб контроллера:

- Разделение пространства для временного хранения записей, направляемых через разные агрегаторы журналов:
  - Предназначено для администраторов и службы поддержки.
  - Предоставляет больше гибкости при создании системы хранения журналов за счет изолированной настройки параметров для разных агрегаторов журналирования.
  - Для каждого внешнего агрегатора выделяется отдельное пространство для временного хранения сообщений, предназначенных для передачи в этот агрегатор. В настройках *syslog-ng* для каждого подключенного агрегатора предусмотрены отдельные секции для задания параметров буфера памяти и буфера диска.
  - Доступно сразу после обновления контроллера.

Исправление ошибок

В новой версии исправлены следующие ошибки:

- Запуск задания через API с указанием в параметре `unified_job_template` идентификатора в виде целого числа (например, `unified_job_template: '7'`) воспринимался контроллером как ошибка. При использовании текстовой строки в качестве значения этого параметра (например, `unified_job_template: 'Demo Job Template'`) запуск задания происходил успешно.

  Теперь можно задавать любой из этих типов значений.

- Некорректная передача переменных при запуске задания из шаблона касалась переменных (**extra_vars**), определенных в настройках шаблона задания и требующих их уточнения при запуске (**prompt on launch**). Если при запуске задания на шаге prompt убрать переменную, то она все равно передавалась в сценарии задания.

  В новой версии переменная, удаленная на шаге prompt, не передается в набор сценариев.

- В системных настройках **Settings - Miscellaneous Authentication** невозможно было изменить параметр **Allow anonymous users to poll metrics**, который отображался на экране в режиме просмотра, но отсутствовал в режиме редактирования.

  В новой версии этот параметр доступен как для просмотра, так и для редактирования.

- Длинные строки без пробелов искажались (не вмещались) в настройках шаблона задания.

  Для устранения ошибки был изменен дизайн полей **Название** (Name) и **Тип** (Type).

- В образе среды исполнения `aa-base-ee` не была установлена утилита `apt` для Python 3.9, необходимый для управления пакетами в формате Debian.

  В новой версии образа `aa-base-ee` эта утилита существует.

- При обновлении платформы с помощью `aa-setup --upgrade` необходимо было давать ссылку на репозиторий, содержащий дистрибутив платформы (версия не важна), в списке источников утилиты `apt`, то есть в файле `/etc/apt/sources.list.d/astra-automation.list` (source list).

  В новой версии утилита `aa-setup` не требует наличия этой ссылки в списке источников.

- При изменении структуры кластера, в процессе которого необходимо было перенести функции узла типа `hybrid` на другой узел, сеть Mesh становилась неработоспособной из-за несогласованности сертификата рецептора на новом узле типа `hybrid` с сертификатом рецептора на старом узле типа `execution`.

  В новой версии платформы такой сценарий стал работоспособен.

- При использовании параметров `nginx_tls_protocols` (для явного задания версии протокола TLS) и `nginx_user_headers` (для задания специального заголовка `X-Custom-Header`) в файле инвентаря установщика сервис NGINX не мог стартовать в процессе установки контроллера.

  В новой версии утилиты развертывания платформы данный сценарий работоспособен.

- При отключении одного или нескольких узлов, выполняющих роль execution node в кластере, в журнале создавалось большое количество однотипных записей.

  В новой версии платформы эта неисправность устранена.

- В процессе работы утилита `aa-setup` может вносить некоторые изменения в исходный файл инвентаря. Это приводило к удалению всех комментариев.

  В новой версии платформы комментарии сохраняются.

- Ссылка на инструкцию по развертыванию платформы, которая появляется в окне терминала после установки пакета `astra-automation-setup`, была некорректной.

  Ссылка исправлена на URL, ведущий на *инструкцию по быстрому развертыванию платформы*.

- Ответ на запрос API `/api` в окне браузера содержал некорректную ссылку на документацию API (иконка в виде знака вопроса).

  Теперь она содержит правильный URL, указывающий на *описание API контроллера Astra Automation*.

- Исправлены ошибки в коллекциях `astra.ald_pro`, `astra.freeipa` и `astra.postgresql`, негативно влиявшие на установку соответствующих продуктов с использованием этих коллекций.

Сценарии применения Astra Automation

Полностью обновлен демонстрационный проект aac-samples[356] путем замены сценариев на новые, описывающие следующие возможности:

- настройка и обновление ОС Astra Linux;
- настройка сетевого оборудования;
- создание и удаление виртуальных машин в Yandex Cloud;
- развертывание продуктов ПАО Группа Астра.

Обновление платформы до версии 1.1

Переход от одной из предыдущих версий к версии `1.1` необходимо производить, используя возможности предыдущей версии:

- Для перехода с версии `1.0-upd2` и более поздних версий 1.0 на `1.1` выполните следующую команду на *установочном узле*:

  ```
  sudo ./aa-setup --upgrade
  ```

- Для перехода с версии `1.0-upd1` на `1.1` выполните следующую команду на установочном узле:

  ```
  sudo ./aa-setup --upgrade --product-version 1.1
  ```

- Переход с версии `1.0` на `1.1` необходимо выполнять так же, как и при *переходе с 1.0 на 1.0-upd1*.

[356] https://source.astragroup.ru/aa-gca/AA/aac-samples

История обновлений стабильной версии

Следующая таблица содержит перечень обновлений в обратном хронологическом порядке и основные данные по ним.

Версия	Дата выпуска	Обновление характеристик (главное)	Исправления (главные)
1.1-upd1	21.10.2024	Первый выпуск Private Automation Hub.	Устранена ошибка, препятствующая передаче и отображению переменных, заданных в шаблоне, во вкладке **Подробности** (Details) выполненного задания.
1.1	19.08.2024	Выпуск утилит Ansible Navigator и Ansible Builder в составе CDK. Добавлен переключатель языка в графическом интерфейсе пользователя контроллера.	Устранена ошибка передачи данных в сценарии при использовании в задании шага **prompt on launch**. Исправлена обработка параметра `unified_job_template` в API.

1.1-upd1

Дата выпуска: 21.10.2024
Тип выпуска: обновление стабильной версии

Состав платформы и совместимость с версиями операционной системы представлены в *таблице*.

Главное

Следующие обновления представляются наиболее важными:

- К платформе добавлен важный компонент – Private Automation Hub.
- Запуск заданий на основе событий с использованием механизма WebHooks теперь также поддерживает *GitFlic*.

Новые возможности и улучшения

Новая версия содержит несколько новых возможностей и улучшений существующих характеристик.

Private Automation Hub

Выпущен новый компонент платформы – Private Automation Hub – на основе свободно распространяемого кода Pulp Project[357].

Этот компонент существенно расширяет сферу применения платформы, благодаря следующим возможностям:

- Управление через графический web-интерфейс и API.
- Распространение важнейших компонентов инфраструктурного кода, которые могут быть загружены из других реестров, включая Automation Hub, Ansible Galaxy, или созданы самостоятельно:

[357] https://pulpproject.org/

  - Коллекции Ansible – управляющий контент для использования в различных сценариях автоматизации.
  - *Образы среды исполнения*.

Обновления коллекций Ansible

Произошли следующие изменения в составе коллекций, распространяемых через Automation Hub:

- Добавлена новая коллекция `astra.repo_mirror`, предназначенная для следующих целей:
  - развертывание deb-репозиториев из файлов ISO;
  - создание зеркалированных репозиториев.
- Обновлена коллекция `astra.aa-controller`:
  - добавлена возможность развертывания Private Automation Hub;
  - добавлены модули управления Automation Controller.
- Обновлена коллекция `astra.keepalived`:
  - улучшено масштабирование сервиса;
  - оптимизирована установка коллекции.
- Обновлена коллекция `astra.nfs`:
  - добавлены возможности тонкой настройки;
  - добавлена возможность управления правами на каталоги.
- Обновлена коллекция `astra.nginx`:
  - добавлены опции SSL;
  - удалены устаревшие директивы.
- Обновлена коллекция `astra.rupost`:
  - добавлена поддержка новой версии RuPost 3.0.1.
- Обновлена коллекция `astra.termidesk`:
  - добавлена поддержка новой версии Termidesk 5.0.0;
  - добавлена роль шлюза;
  - добавлены модули для управления продуктом.
- Обновлена коллекция `astra.ald_pro`:
  - переработана роль `replica`;
  - коллекция переведена в статус сертифицированной.

Обновления CDK

Произведены следующие изменения:

- Установка утилит Ansible (Ansible Navigator, Ansible Builder, Ansible Molecule, and Ansible Lint) теперь возможна в условиях отсутствия связи с интернетом:
  - Эта возможность предназначена для разработчиков контента и пользователей утилит командной строки.
  - Позволяет устанавливать утилиты в закрытом контуре, когда нет доступа к репозиторию Astra Automation, размещенному в интернете.
  - Создан пакет (offline bundle) для установки утилит локально без доступа в интернет.
  - Пакет доступен в личном кабинете клиента.

- Утилиты Ansible Lint и Ansible Molecule переведены из состояния *technical preview* в состояние полноценной технической поддержки.

Обновления в Automation Controller

В контроллере произведены следующие изменения:

- Добавлена возможность обрабатывать входные запросы от перехватчиков событий webhook системы контроля версий GitFlic:
  - Предназначена для команд devops, интегрирующих различные системы.
  - Позволяет автоматизировать обработку событий в GitFlic средствами Astra Automation.
  - Добавлена функция обработки запросов от GitFlic в приемнике запросов (receiver) контроллера.
  - Новая возможность доступна сразу после обновления контроллера.

Исправление ошибок

В новой версии исправлены следующие ошибки:

- При удалении компонентов платформы с помощью утилиты `aa-setup` кроме компонентов платформы удалялись также некоторые другие компоненты программного обеспечения. Это поведение исправлено, так что команда `aa-setup -u` удаляет только компоненты платформы на узлах, отмеченных в инвентаре.
- Если при обновлении версии платформы в инвентаре был указан узел, который по каким-то причинам оказался недоступен, утилита `aa-setup` ошибочно интерпретировала это как отсутствие требуемого компонента платформы на этом узле и предлагала выполнить установку компонента. В новой версии утилита корректно определяет отсутствие узла и завершает обновление аварийно.
- В контроллере после выполнения задания по шаблону переменные, заданные в шаблоне, не отображались во вкладке **Подробности** (Details) выполненного задания. В новой версии они отображаются корректно.
- В контроллере невозможно было запланировать задания по расписанию. В новой версии расписания (schedules) создаются корректно.
- Исправлено несколько ошибок в коллекциях Ansible, распространяемых через Automation Hub:
  - В коллекциях `astra.aldpro`, `astra.ceph` и `astra.hardening` исправлена ошибка совместимости со средой исполнения `aa-base-ee`.
  - В коллекции `astra.grafana` и `astra.freeipa` исправлены ошибки в документации.
  - В коллекции `astra.yandexcloud` исправлены ошибки в обеспечении безопасности и в документации.
  - В коллекции `astra.iscsi` исправлены ошибки в работе инициатора.
  - В коллекции `astra.postgresql` исправлены ошибки при работе с базой данных PARSEC.
  - В коллекции `astra.rupost` исправлена ошибка установки сертификатов.

Обновление платформы до версии 1.1-upd1

Переход от одной из предыдущих версий к версии `1.1` необходимо производить, используя возможности предыдущей версии:

- Для перехода с версии `1.0-upd2` и более поздних версий на `1.1-upd1` выполните следующую команду на *установочном узле*:

```
sudo ./aa-setup --upgrade
```

- Для перехода с версии `1.0-upd1` на `1.1-upd1` выполните следующую команду на установочном узле:

```
sudo ./aa-setup --upgrade --product-version 1.1-upd1
```

- Переход с версии `1.0` на `1.1-upd1` необходимо выполнять так же, как и при *переходе с 1.0 на 1.0-upd1*.

21.7.4 Версия 1.0 (не поддерживается)

Дата выпуска: 15.02.2024
Тип выпуска: стабильная версия

Примечание

Первый выпуск продукта.

Состав платформы и совместимость с версиями операционной системы представлены в *таблице*.

Основные характеристики

Astra Automation является комплексной платформой для автоматизации задач по развертыванию и управлению инфраструктур различной сложности.

Состав

Платформа Astra Automation состоит из следующих основных компонентов:

- Automation Hub – облачный сервис, предоставляющий доступ к коллекциям Ansible и модулям Terraform, составляющим основу инфраструктурного кода для выполнения задач Astra Automation.
- Automation Controller – контроллер (плоскость управления), предоставляющий удобный графический интерфейс и API для управления выполнением задач Astra Automation. Этот компонент наследует характеристики проекта AWX[358], адаптирует и развивает их для удовлетворения нужд заказчиков ПАО Группа Астра.
- Execution plane – сеть исполнительных узлов (плоскость исполнения), расширяющая возможности Automation Controller по масштабированию и повышению уровня защиты структуры управления. Этот компонент позволяет создавать распределенную систему управления большими и сверхбольшими инфраструктурами, разделенными по географическому или другим принципам.

Функциональные возможности

Платформа позволяет развертывать инфраструктуру заказчика, устанавливать и настраивать программное обеспечение согласно *инструкции*.

Развертывание платформы

Развертыванию на стороне заказчика подлежат следующие компоненты:

- Automation Controller можно установить в различных конфигурациях:
  - на одном узле для ознакомления и тестирования или для управления небольшой инфраструктурой;

[358] https://github.com/ansible/awx

  - в кластере, как основной рекомендуемый вариант.
- (Дополнительно) Компонент Execution plane необходим для создания распределенной структуры управления. Его устанавливают в едином процессе развертывания вместе с контроллером или при последующем обновлении.
- СУБД PostgreSQL необходимо установить на каком либо сервере в процессе развертывания платформы или использовать готовую базу данных.

Особенности установки и ограничения:

1. Установка первой версии платформы протестирована в Astra Linux Special Edition с использованием следующих версий:

Версия операционной системы	Версия ядра
1.7.4	5.15.0-generic
1.7.5	6.1.50-generic

2. При использовании Astra Linux Special Edition 1.7.5 необходимо исключить установку графической системы Fly и средств работы с графикой. Для развертывании кластера на виртуальных машинах рекомендуется использовать универсальные базовые образы Astra Linux от производителя[359], в названии которых отсутствует поле `gui`, например образ alse-vanilla-1.7.5-virtualbox-adv-mg12.1.2.ova[360] для VirtualBox.

Примечание

Исправлено в обновлении *1.0-upd1 (не поддерживается)*.

Преимущества

Astra Automation предоставляет ряд преимуществ по сравнению с решениями, доступными от свободно распространяемых продуктов:

- Безопасный проверенный код инструментария.
- Использование надежной операционной системы, оснащенной средствами защиты информации в соответствии с требованиями официальных регуляторов.
- Доступ к уникальному проверенному инфраструктурному коду, предназначенному для установки на операционной системе Astra Linux.
- Регулярные плановые обновления стабильных версий и внеплановые обновления текущих версий по запросам при обнаружении ошибок или уязвимостей.
- Резервное копирование и восстановление данных и настроек.
- Использование более совершенной системы журналирования, основанной на syslog-ng.
- Использование защищенного протокола SSL/TLS при работе с PostgreSQL.
- Документация на русском языке.
- Техническая поддержка.
- Обучение.

История обновлений стабильной версии

Следующая таблица содержит перечень обновлений в обратном хронологическом порядке и основные данные по ним.

[359] https://registry.astralinux.ru/latest/

[360] https://registry.astralinux.ru/images/alse/virtualbox/alse-vanilla-1.7.5-virtualbox-adv-mg12.1.2.ova

Версия	Дата выпуска	Обновление характеристик (главное)	Исправления (главные)
1.0-upd2	22.04.2024	Обновление кода из проекта AWX. Улучшена функция обновления версии платформы с помощью утилиты `aa-setup`.	Исключен излишний процесс сканирования всего репозитория при развертывании или обновлении платформы.
1.0-upd1	14.03.2024	Добавлена возможность обновления версии платформы с помощью утилиты `aa-setup`.	Исправлен процесс восстановления базы данных кластера, состоящего из узлов типа `hybrid`.
1.0	15.02.2024		

1.0-upd2 (не поддерживается)

Дата выпуска: 22.04.2024
Тип выпуска: обновление стабильной версии

Состав платформы и совместимость с версиями операционной системы представлены в *таблице*.

Главное

Следующие обновления представляются наиболее важными:

- Automation Controller теперь базируется на AWX версии `23.7.0`.
- Улучшена функция обновления версии платформы с помощью утилиты `aa-setup`.
- Утилита `aa-setup` позволяет указать образ среды исполнения при развертывании или обновлении платформы.
- Устранено ограничение на количество узлов в кластере Astra Automation.
- Обновлена версия образа среды исполнения.

Новые возможности и улучшения

Следующие секции содержат описание новых возможностей и улучшений существующих характеристик.

Переход на новую версию проекта AWX

Automation Controller теперь базируется на новой версии upstream-проекта AWX. Произведен переход с версии 22.1.0[361] на версию 23.7.0[362]. Из большого количества обновлений наиболее заметными и значимыми являются следующие:

- В панели навигации пункт просмотра топологии доступен только для системных администраторов и аудиторов[363]. Ранее он был виден также для администраторов организаций, но, поскольку сама топология им недоступна, это приводило к появлению сообщения об ошибке.
- Обновлено расширение (plugin) credential_plugin[364] так, что теперь при работе с хранилищем vault поддерживается аутентификация через LDAP.

[361] https://github.com/ansible/awx/releases/tag/22.1.0
[362] https://github.com/ansible/awx/releases/tag/23.7.0
[363] https://github.com/ansible/awx/pull/13905
[364] https://github.com/ansible/awx/pull/14654

- Поддержка webhooks от Bitbucket Data Center[365] с отправлением обратно состояния соответствующего запроса.

Выполнен перевод на русский язык новых и обновленных строк в графическом интерфейсе контроллера.

Утилита aa-setup

Выпущена версия утилиты `1.0.2` в составе пакета `astra-automation-setup`, содержащая новые возможности:

- Улучшена функция обновления версии платформы:
  - Предназначено для специалистов по установке и обслуживанию Astra Automation.
  - Новая функция автоматически определяет текущую версию платформы на всех компонентах кластера с помощью файла `/etc/astra_automation.version`, присутствующего в каждом узле кластера.
  - Пользователь, у которого есть административные полномочия на узлы кластера с доступом по SSH (используя приватный и публичный ключи SSH), может запускать процесс обновления, указав аргумент `--upgrade`. Утилиту можно запускать с узла, прошедшего процесс подготовки. Ранее процесс обновления необходимо было запускать с того узла, который использовался для первоначального развертывания платформы.
  - Функция доступна сразу после обновления версии утилиты, но применить ее можно только с выпуском следующей версии платформы, то есть при переходе с версии `1.0-upd2` на `1.0-upd3`. Обновление на рассматриваемую версию подробно описано в секции *Обновление платформы до версии 1.0-upd2*.
- Утилита `aa-setup` позволяет указать образ среды исполнения, используемой по умолчанию для выполнения заданий:
  - Предназначено для специалистов по установке и обслуживанию Astra Automation.
  - Добавлен аргумент `--default-job-ee`, указывающий адрес требуемого образа.
  - Данная функция позволяет использовать требуемый образ среды исполнения вместо поставляемого в составе пакета.
  - Опция доступна сразу после установки или обновления утилиты.

  Пример запуска команды:

```
sudo ./aa-setup --default-job-ee registry.astralinux.ru/aa/aa-base-ee:0.5.1
```

- Устранено ограничение на количество узлов в кластере Astra Automation:
  - Предназначено для специалистов по установке и обслуживанию Astra Automation.
  - Устранено ограничение на количество устанавливаемых узлов.
  - Позволяет создавать более сложную топологию платформы.
  - Доступно сразу после установки или обновления утилиты.
- Добавлена проверка корректности установки параметра `pg_host`:
  - Предназначено для специалистов по установке и обслуживанию Astra Automation.
  - Проверяет, что при отсутствии в описании инвентаря группы `database` (установка базы данных на локальном узле), параметру `pg_host` не назначено какое-либо значение. Допускается назначать адрес локального узла, то есть, 127.0.0.1. При нарушении этого условия утилита завершается аварийно с выводом сообщения об ошибке вида:

[365] https://github.com/ansible/awx/pull/14674

```
Параметр pg_host должен содержать пустое значение или '127.0.0.1' при␣
↪выполнении установки с внутренней базой данных (незаполненной секцией␣
↪[database]).
```

  - Эта функция повышает надежность развертывания платформы и сокращает время на выяснение возможных проблем.
  - Доступно сразу после установки или обновления утилиты.
- Утилита `aa-setup` позволяет удалить ранее установленные компоненты платформы:
  - Предназначено для специалистов по установке и обслуживанию Astra Automation.
  - Добавлена функция удаления, вызываемая с помощью аргумента `-u` или `--uninstall`. Она удаляет все данные, установленные в процессе развертывания Astra Automation на узлах, входящих в состав файла инвентаризации.
  - Новая функция упрощает процесс удаления компонентов платформы.
  - Доступна сразу после установки или обновления утилиты.

  Пример полного удаления всех компонентов Astra Automation из узлов, перечисленных в файле `inventory`:

```
aa-setup -i inventory -u --force-postgres-removal
```

Обновление спецификации API

Произведено обновление спецификации API на версию `1.0.2+23.7.0`.

Образ среды исполнения

Образ среды исполнения (EE, Execution Environment) `aa-base-ee`, используемый в платформе по умолчанию, обновлен до `0.5.1`:

- Предназначено для пользователей Astra Automation и специалистов по установке и настройке.
- Выполнены следующие обновления:
  - В основу положен образ Astra Linux Special Edition 1.7.5.UU1 (registry.astralinux.ru/library/alse:1.7.5uu1-mg12.5.0).
  - Python и необходимые пакеты установлены из публичного репозитория Astra Automation.
  - Обновлены пакеты `ansible` и `ansible-core`.
  - Terraform заменен на OpenTofu[366].
- Новая версия расширяет возможности EE по использованию необходимого программного обеспечения и новых возможностей Ansible. Она также позволяет отказаться от Terraform, на использование которого наложены ограничения от производителя.
- Образ доступен с момента выпуска данной версии Astra Automation.

Обновления коллекций Ansible

Произведено обновление следующих коллекций:

- Коллекция `astra.ceph` обновлена до версии `2.1.5` для обеспечения совместимости с EE версии `0.5.x`.
- Коллекция `astra.ald_pro` обновлена до версии `0.8.0` для поддержания ALD Pro версии `2.3.0`.
- Коллекция `astra.postgresql` обновлена до версии `2.2.1` для совместимости с операционной системой Astra Linux Special Edition версии `1.8`.

[366] https://opentofu.org/

Исправление ошибок

В новой версии исправлены следующие ошибки:

- Исключен излишний процесс сканирования версий платформы в репозитории, когда версия платформы явно указана в качестве аргумента при запуске утилиты `aa-setup`.
- Устранена неисправность резервного копирования базы данных (dump), выражавшаяся в том, что прерывание процесса dump по какой-либо причине приводило к невозможности повторного запуска этого процесса.
- В структуре LOG_AGGREGATOR_SYSLOGNG_DEBUG, используемой для записи строк в журнал, исправлено название сервиса с `rsyslogd` на `syslog-ng`.

Обновление платформы до версии 1.0-upd2

Обновление с помощью утилиты `aa-setup`, описанное ранее, доступно уже в этой версии, но применить новые возможности можно будет только при переходе на будущую версию, то есть на `1.0-upd3` и последующие.

Поэтому переход от одной из предыдущих версий необходимо производить, используя возможности той версии:

- Переход с версии `1.0` на `1.0-upd2` необходимо выполнять так же, как и при *переходе с 1.0 на 1.0-upd1*.
- Переход с версии `1.0-upd1` на `1.0-upd2` позволяет использовать утилиту `aa-setup` из установочного пакета для версии `1.0-upd1`:

```
sudo ./aa-setup --upgrade --product-version 1.0-upd2
```

Примечание

Есть временные ограничения:

- Вместе с аргументом `--upgrade` требуется использовать `--product-version` для явного указания версии Astra Automation.
- Обновление необходимо производить с того установочного узла, который использовался при первоначальном развертывании платформы. Вместо такого узла можно воспользоваться другим, на котором установлен пакет для развертывания Astra Automation и восстановлены необходимые настройки установочного узла.

При любом из перечисленных вариантов перехода необходимо использовать файл, описывающий действующий инвентарь платформы. По умолчанию он находится по адресу `/opt/rbta/aa/astra-automation-setup/inventory`.

1.0-upd1 (не поддерживается)

Дата выпуска: 14.03.2024
Тип выпуска: обновление стабильной версии

Состав платформы и совместимость с версиями операционной системы представлены в *таблице*.

Главное

Следующие обновления представляются наиболее важными:

- Добавлена функция автоматического обновления версии платформы с помощью утилиты `aa-setup`.

- Исправлен процесс восстановления базы данных кластера, состоящего из узлов типа `hybrid`.
- Исправлен процесс записи сообщений в журнал в формате JSON.

Новые возможности и улучшения

Следующие секции содержат описание новых возможностей и улучшений существующих характеристик.

Утилита aa-setup

Выпущена версия утилиты 1.0.1 в составе пакета astra-automation-setup, которая содержит новые возможности:

- Утилиту `aa-setup` теперь можно использовать для обновления версии платформы:
  - Предназначено для специалистов по установке и обслуживанию Astra Automation.
  - Добавлена опция `--upgrade` для обновления версии кластера.
  - Новая функция автоматически определяет текущую версию и обновляет ее до новейшей версии, доступной в репозитории Astra Automation.
  - Функция доступна сразу после обновления утилиты, но применить ее можно только с выпуском следующей версии платформы, то есть `1.0-upd2`. Обновление на рассматриваемую версию подробно описано в секции *Обновление платформы до версии 1.0-upd1*.
- С помощью утилиты `aa-setup` можно узнать список версий платформы, доступных в репозиториях APT:
  - Предназначено для специалистов по установке и обслуживанию Astra Automation.
  - Добавлена опция `--check-releases`.
  - Данная опция позволяет инженерам узнать список версий платформы, доступных в главном репозитории (dl.astralinux.ru) или в репозитории, указанном с помощью аргумента `--repo-url`. Таким образом, они могут выбрать конкретную версию продукта для первичного развертывания или обновления.
  - Доступно сразу после установки или обновления утилиты`.

  Пример запуска команды:

```
sudo ./aa-setup --check-releases
```

  Пример вывода:

```
INFO: Astra Automation Setup has been started
INFO:

Following Astra Automation releases available:
mvp, mvp-upd1, mvp-upd2, 1.0, 1.0-upd1

To upgrade to the latest version, use:
./aa-setup --upgrade

Information about Astra Automation releases:
https://docs.astra-automation.ru/latest/releases/

INFO: Available releases checked
```

Исправление ошибок

В новой версии исправлены следующие ошибки:

- Исправлен процесс восстановления базы данных кластера, состоящего из узлов типа `hybrid`. Ранее при восстановлении данные не передавались из архива в базу данных.
- Исправлен процесс записи сообщений в журнал в формате JSON. Ранее формат записи JSON был искажен. Это, в частности, приводило к ошибкам интерпретации записей с помощью внешних обработчиков журналов, например, `logstash`.
- Исправлен счетчик затраченного времени при обработке задания. Ранее, при определенных условиях, он мог начинать отсчет от некоторого отрицательного значения.
- Устранено ограничение на наличие графических средств в узлах, на которых устанавливаются компоненты кластера платформы.

Обновление платформы до версии 1.0-upd1

Обновление с помощью утилиты `aa-setup` доступно уже в этой версии, но применить эту функцию можно будет только при переходе на будущую версию, то есть на `1.0-upd2` и последующие.

Поэтому обновление с версии `1.0` до `1.0-upd1` требует особой подготовки и последовательности.

Подготовка

Необходимо обеспечить выполнение следующих условий:

- Для обновления используйте тот же узел, с помощью которого происходило развертывание платформы версии 1.0, без внесения каких-либо изменений на нем.
- Убедитесь, что на этом узле остался в неизменном виде каталог `/opt/rbta/aa/astra-automation-setup/`, в частности файл инвентаря, описывающий состав платформы:

  `/opt/rbta/aa/astra-automation-setup/inventory` – описание структуры платформы.

Обновление

Обновление с версии `1.0` до `1.0-upd1` выполняйте в следующем порядке:

1. Добавьте ссылку на репозиторий, содержащий файлы Astra Automation, в файл `/etc/apt/sources.list` или же в любой файл в каталоге `/etc/apt/sources.d/`:

   ```
   deb https://dl.astralinux.ru/aa/aa-debs-for-alse-1.7 1.0-upd1 main
   ```

   Примечание

   Убедитесь, что нет конфликтов, то есть, что этот репозиторий не объявлен в каком-либо другом из перечисленных файлов.

2. Обновите индекс пакетов с помощью следующей команды:

   ```
   sudo apt update
   ```

3. Обновите пакет `astra-automation-setup` с помощью следующей команды:

   ```
   sudo apt install --only-upgrade astra-automation-setup
   ```

4. Запустите процесс обновления таким же образом, как вы изначально развертывали платформу, то есть одним из следующих способов:

   - Запуск с использованием привилегий пользователя root:

```
sudo ./aa-setup
```

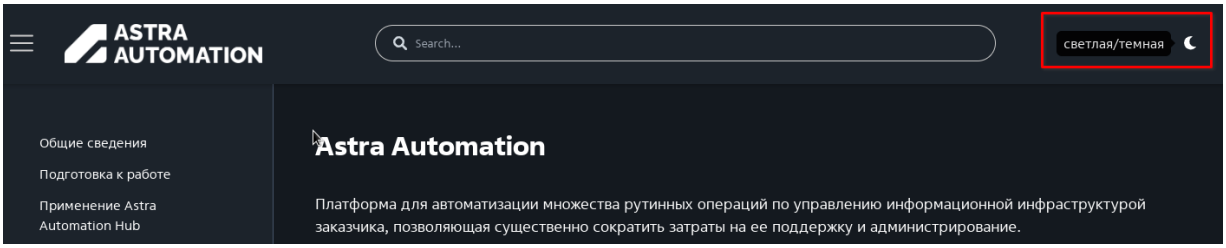
- Запуск с привилегиями обычного пользователя:

```
./aa-setup --log-path=<log_path>
```

Изменения в документации

Добавлены новые возможности в документации:

- Добавлена поддержка темной темы:



- Изменен стиль для более заметного разграничения заголовков второго и третьего уровней.

Не-алфавитный

Г

О

П

С

У

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Y