Red Hat
Training and
Certification

DO467

Travis Michette

Version 1.0

Table of Contents

1. Installing Red Hat Ansible Automation Platform

1.1. Explaining the Red Hat Ansible Automation Platform Architecture

1.1.1. Red Hat Ansible Automation Platform

1.1.2. Red Hat Ansible Automation Platform Components

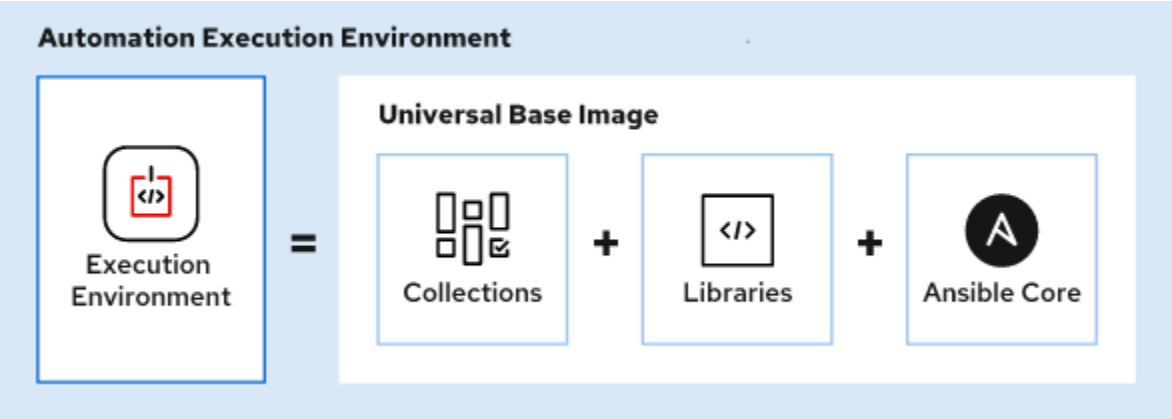

Figure 1. Ansible Execution Environment

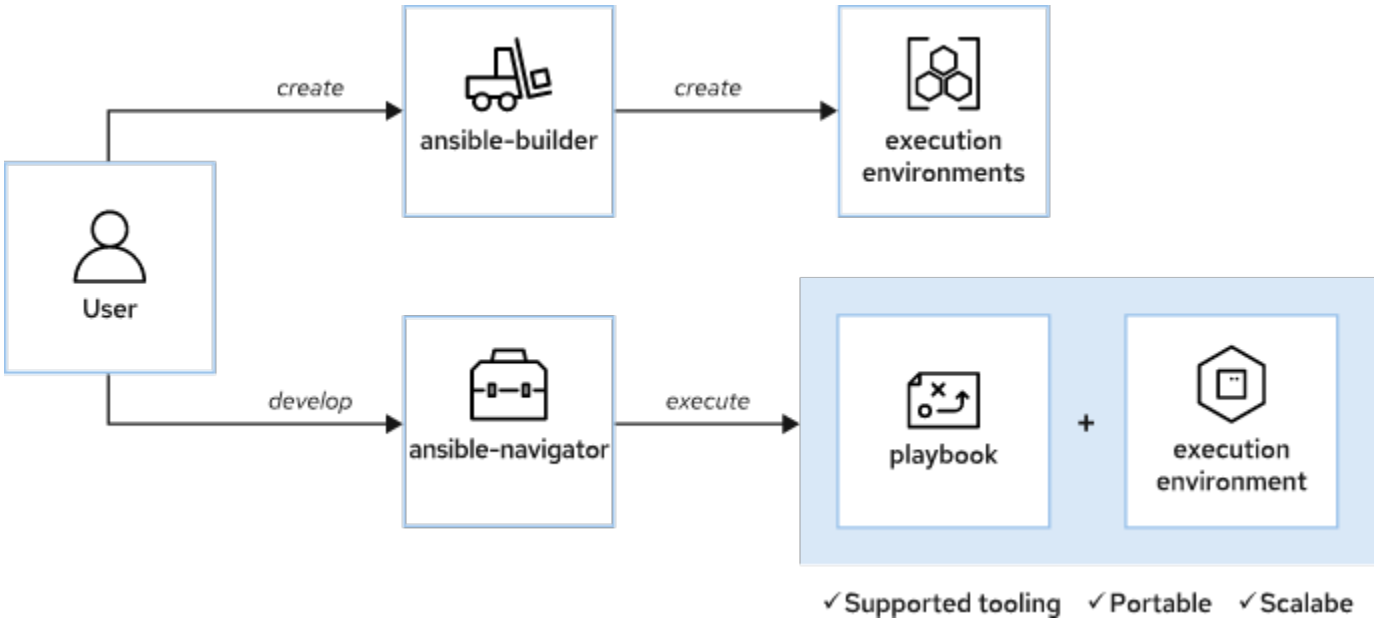

Figure 2. Creating an Ansible Execution Environment

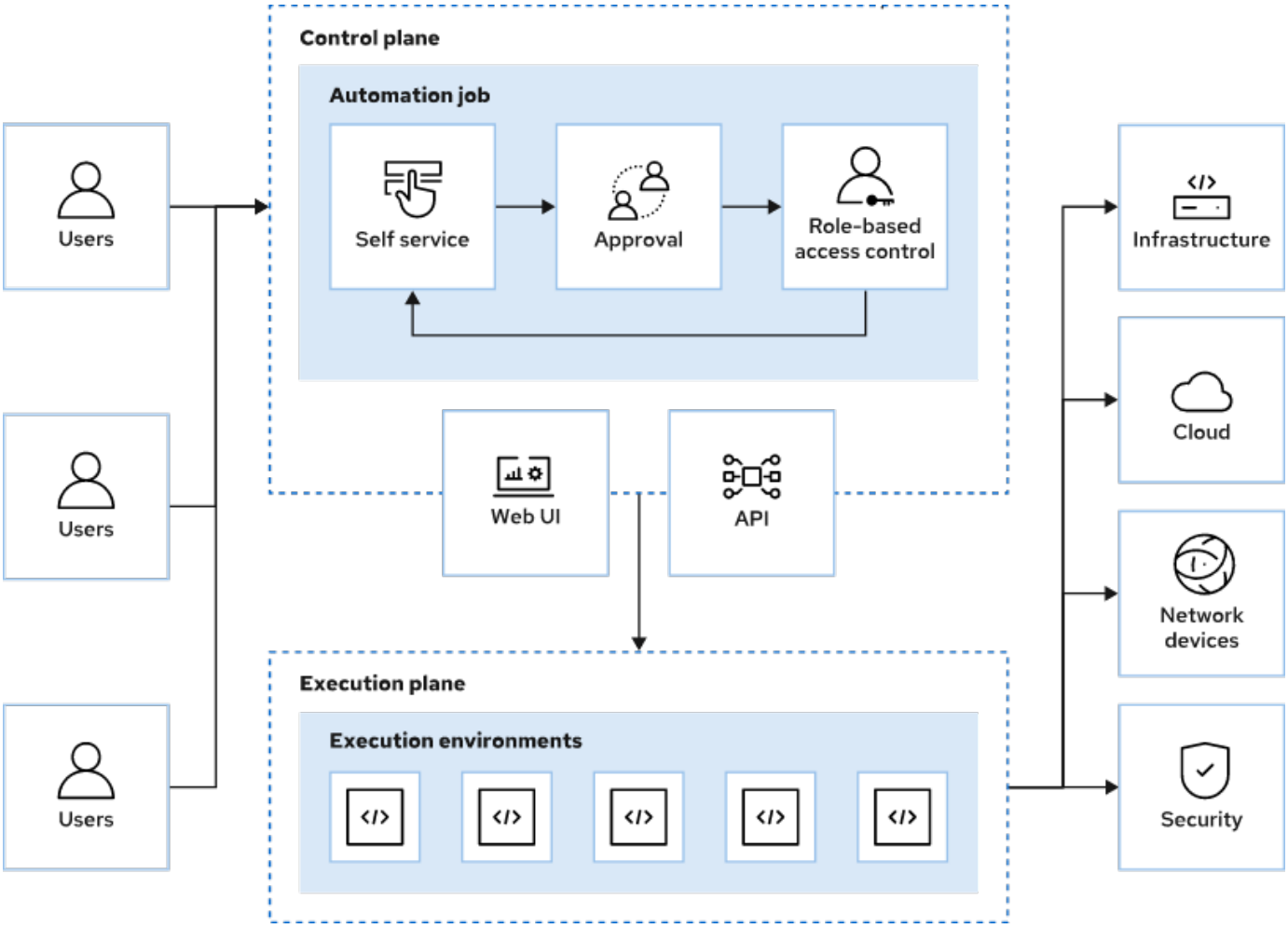

Figure 3. Ansible Automation Controller Components

1.1.2.1. Ansible Core

1.1.2.2. Ansible Content Collections

1.1.2.3. Automation Content Navigator

1.1.2.4. Automation Execution Environments

1.1.2.5. Automation Controller

1.1.2.6. Automation Hub and Private Automation Hub

1.1.2.7. Red Hat Insights for Red Hat Ansible Automation Platform

1.1.3. Why Use Ansible Automation Platform?

1.2. Installing Automation Controller and Private Automation Hub

Installing Exercise Time

It can take up to 15 minutes for the GE **lab start install-installation** so it is recommended to run this script at the start of the lecture.

1.2.1. Planning the Installation

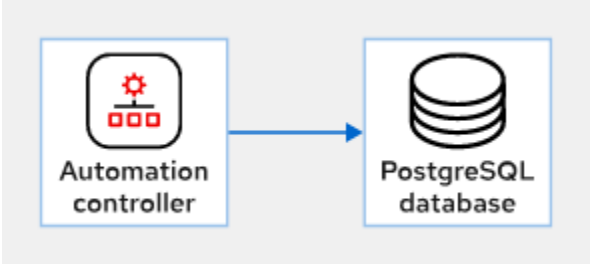

Figure 4. Standalone Automation Controller

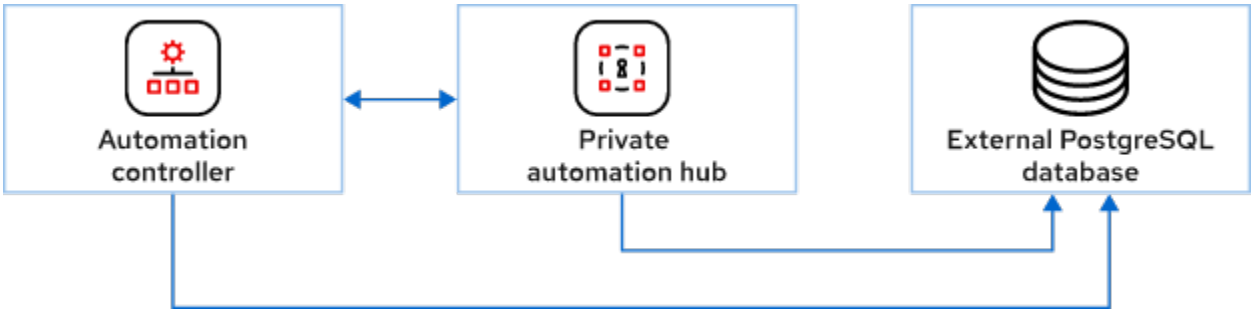

Figure 5. Automation Controller with Private Automation Hub

Automation Mesh Considerations

The Database server must be a separate machine is a requirement for using Ansible Automation Mesh.

Ease of Installation

When installing both Ansible Automation Controller and Private Automation Hub, it is recommended to install both of these items using the same setup command and inventory source. This will allow deployments of both services, but also provides additional benefits with automatic configurations such as creating the "link-style" resources between Controller and Hub like execution environments, and other items such as credentials.

Installation of Automation Controller and Private Automation Hub

In environments having both Automation Controller and Private Automation Hub, it is required that these be installed on separate nodes as they cannot be installed on the same node.

1.2.1.1. Standalone Automation Controller with a Database on the Same Node

1.2.1.2. Standalone Private Automation Hub with a Database on the Same Node

1.2.1.3. Automation Controller and Private Automation Hub with External Database Servers

Classroom and Lab Environment

The deployment for this course will be using both Controller and Hub connected to a shared **External** PostgreSQL database.

1.2.1.4. Advanced Deployment Scenarios

1.2.2. Installation Requirements

Table 1. Hardware Requirements

Machine name	RAM	CPU
Controller	16GB	4 CPUs
Hub	8GB	2 CPUs

Minimum vs. Practical Requirements

The memory and CPU requirements depend really on the size and implementation in the environment. Essentially a good rule of thumb is to have 1GB RAM (memory) for every ten (10) forks and keep at least 2GB for automation controller services. It is also important that with additional forks CPU capacity will also be increased.

Classroom Environment Doesn't Meet Specifications

The classroom environment being utilized doesn't meet the minimum specifications, so during the exercise, the **./setup.sh -e ignore_preflight_errors=true** is run. Specifically, the **-e ignore_preflight_errors=true** instructs the installer to ignore the checks for system requirements. This should not be done in a production environment.

Additionally, for the installation, we are running the setup as **root** since the root user exists on all systems being modified and SSH keys have been pre-distributed.

The classroom environment also uses an internal CA and custom certificates that have been created by the Red Hat Training team. These certificates must be obtained for your environment prior to installation if you want to leverage custom internal CAs.

1.2.2.1. Database Storage

1.2.3. Subscription and Support

1.2.4. Installing Red Hat Ansible Automation Platform

As mentioned above, it is recommended to install both Controller and Hub from the same inventory file using a single **setup.sh** command. In order to successfully install Controller and Hub, the inventory file must be updated and modified providing credentials and FQDMs in the various section headers.

Extra Resources Added when Installed Together

If installed together, the setup script will create additional controller resources such as hub credentials and perform the configuration automatically as part of the installation/setup process. If done separately, it will be necessary to create the resources in controller manually so that controller can communicate with hub. The other benefit of using a combined installation from the bundled installer is that there are three (3) execution environment images (EEIs) included with the bundled installer and these are automatically loaded into hub.

1.2.4.1. Installing Automation Controller

1.2.4.2. Installing Private Automation Hub

1.2.5. Replacing the CA Certificate

1.2.5.1. Gathering Certificates and Private Keys

1.2.5.2. Preparing the Systems

1.2.5.3. Trusting Custom CA Certificates

Installing Exercise Time

It can take up to 15 minutes for the GE **lab start install-installation** so it is recommended to run this script at the start of the lecture.

1.2.6. DEMO: Installing Automation Controller and Private Automation Hub

Automation Controller and Private Automation Hub can both be installed from the **same** machine provided that they are both specified in the inventory file and that the installation user and installation machine has access to all systems specified in the **inventory** file and that the user has the ability to SSH/SUDO without passwords.

Automation Hub and Controller Placement

Ansible Controller and Ansible Private Automation Hub must be installed on separate systems and cannot be installed on the same system.

Example 1. **DEMO: Installing Automation Hub and Controller**

1. Obtain the bundled installer and untar the file

```
[student@workstation ~]$ tar xvf ansible-automation-platform-setup-bundle-2.2.0-
6.1.tar.gz

[student@workstation ~]$ mv ansible-automation-platform-setup-bundle-2.2.0-6.1
AAP2

[student@workstation ~]$ cd AAP2/
```

2. Update the inventory file with the system FQDNs or IP Addresses

Listing 1. Update the Inventory File

```
[student@workstation AAP2]$ vim inventory
```

```
[automationcontroller] ①
controller.lab.example.com


[execution_nodes]

[automationhub] ②
hub.lab.example.com

[automationcatalog]

[database] ③
db.lab.example.com


[all:vars]
admin_password='redhat' ④

pg_host='db.lab.example.com' ⑤
pg_port=5432 ⑥

pg_database='awx'
pg_username='awx'
pg_password='redhat' ⑦


registry_url='hub.lab.example.com' ⑧
```

```
registry_username='admin' ⑨
registry_password='redhat' ⑩



# Automation Hub Configuration ⑪
#

automationhub_admin_password='redhat'

automationhub_pg_host='db.lab.example.com'
automationhub_pg_port=5432

automationhub_pg_database='automationhub'
automationhub_pg_username='automationhub'
automationhub_pg_password='redhat'
automationhub_pg_sslmode='prefer'

# SSL Settings ⑫

custom_ca_cert=/home/student/certs/classroom-ca.pem
web_server_ssl_cert=/home/student/certs/controller.lab.example.com.crt
web_server_ssl_key=/home/student/certs/controller.lab.example.com.key
automationhub_ssl_cert=/home/student/certs/hub.lab.example.com.crt
automationhub_ssl_key=/home/student/certs/hub.lab.example.com.key
postgres_use_ssl=True
postgres_ssl_cert=/home/student/certs/db.lab.example.com.crt
postgres_ssl_key=/home/student/certs/db.lab.example.com.key
```

① Specify the Controller Node

② Specify the Private Automation Hub Node

③ Specify the Database Node

④ Specify the **admin** password for Controller

⑤ Specify the Database FQDN

⑥ Specify the Database Port

⑦ Specify the Database Password

⑧ URL and Registry for Container Images/Execution Environments

⑨ Username for Registry

⑩ Password for Registry

⑪ Ansible Automation Hub Configuration Settings

⑫ SSL Settings

Database

If you are running the database locally and not as a separate installation, you can leave the database section blank and the **pg_host** and **pg_port** blank. This will cause the installer to setup the database locally with the deployed AAP application.

Registry

Setting the registry for **hub.example.com** will allow the installer to link and configure Ansible Automation Hub to Ansible Controller. It will also ensure that the execution environments container in the bundled installer will be loaded properly into Ansible Automation Hub.

SSL

The classroom and lab environment has been configured to run with SSL enabled. In order for the certificates to work properly, the SSL certificates have been supplied in the **/home/student/certs** directory. These certificates must be specified in the **inventory** file. In the default inventory file, the certificates and SSL settings are generally commented out, so it is possible to just place the certificate information at the bottom of the inventory file to prevent searching for each line.

Listing 2. Default SSL Certificate

```
# SSL-related variables

# If set, this will install a custom CA certificate to the system
trust store.
# custom_ca_cert=/home/student/certs/classroom-ca.pem

# Certificate and key to install in nginx for the web UI and API
# web_server_ssl_cert=/path/to/tower.cert
# web_server_ssl_key=/path/to/tower.key
```

3. View final inventory file

```
[student@workstation AAP2]$ grep -Ev "^#|^$" inventory
[automationcontroller]
controller.lab.example.com
[automationcontroller:vars]
peers=execution_nodes
[execution_nodes]
[automationhub]
hub.lab.example.com
[automationcatalog]
[database]
db.lab.example.com
[sso]
[all:vars]
admin_password='redhat'
pg_host='db.lab.example.com'
pg_port=5432
pg_database='awx'
pg_username='awx'
pg_password='redhat'
pg_sslmode='prefer'  # set to 'verify-full' for client-side enforced SSL
registry_url='hub.lab.example.com'
registry_username='admin'
registry_password='redhat'
receptor_listener_port=27199
automationhub_admin_password='redhat'
automationhub_pg_host='db.lab.example.com'
automationhub_pg_port=5432
automationhub_pg_database='automationhub'
automationhub_pg_username='automationhub'
automationhub_pg_password='redhat'
automationhub_pg_sslmode='prefer'
automationcatalog_pg_host=''
automationcatalog_pg_port=5432
automationcatalog_pg_database='automationservicescatalog'
automationcatalog_pg_username='automationservicescatalog'
automationcatalog_pg_password=''
sso_keystore_password=''
sso_console_admin_password=''
custom_ca_cert=/home/student/certs/classroom-ca.pem
web_server_ssl_cert=/home/student/certs/controller.lab.example.com.crt
web_server_ssl_key=/home/student/certs/controller.lab.example.com.key
automationhub_ssl_cert=/home/student/certs/hub.lab.example.com.crt
automationhub_ssl_key=/home/student/certs/hub.lab.example.com.key
postgres_use_ssl=True
postgres_ssl_cert=/home/student/certs/db.lab.example.com.crt
postgres_ssl_key=/home/student/certs/db.lab.example.com.key
```

Using **grep** *to remove comments and blank lines*

Listing 3. Source Description

```
grep -Ev "^#|^$" <FILENAME>
```

4. Run the installation **setup.sh** script as the root user with **ignore_preflight_errors=true** as the systems in this course don't meet the minimum hardware requirements.

```
[student@workstation AAP2]$ sudo -i
[sudo] password for student:

[root@workstation ~]# cd ~student/AAP2/

[root@workstation AAP2]# ./setup.sh -e ignore_preflight_errors=true
```

Bundled Software Installer

It is important to at least save the bundled software installer archive **TGZ** file or to save the entire bundled installation directory. In addition, you will also want to save the **Inventory** file that was created so that adding additional components later, performing system backups/restores, and other administrative and maintenance tasks can be performed easily.

5. Install the licenses for Controller by providing the **manifest.zip** file to controller in the WebUI.

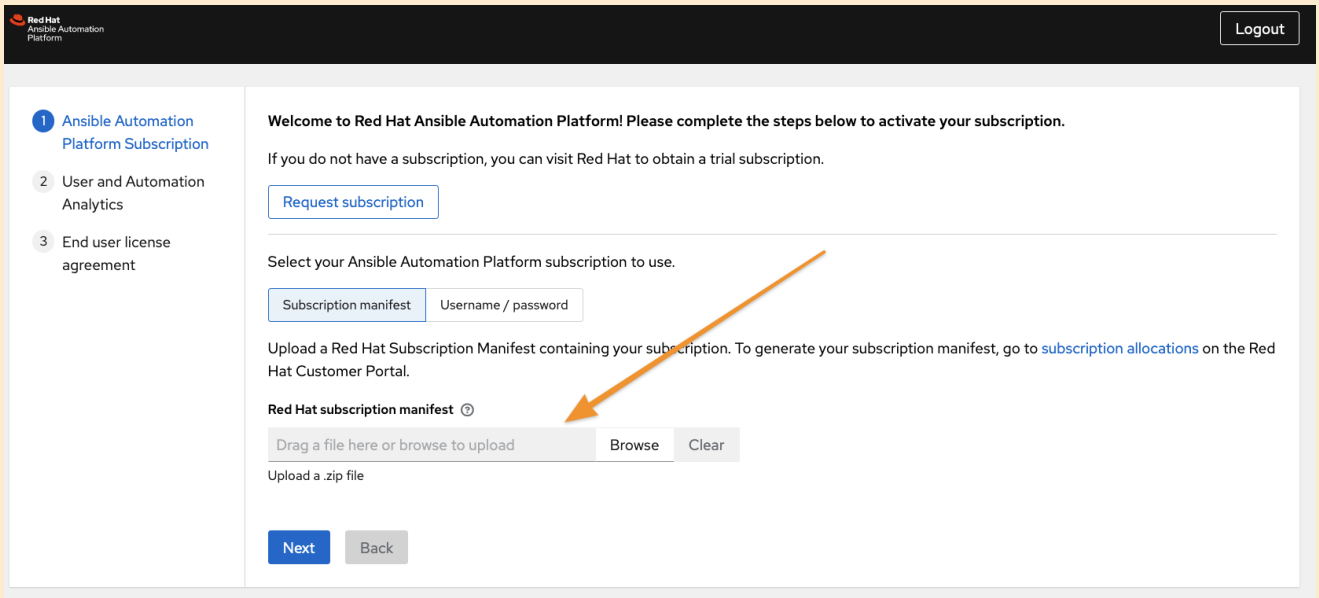

Figure 6. Ansible Controller License

1. Verify **Automation Hub** is installed

1.3. Initial Configuration of Automation Controller and Private Automation Hub

1.3.1. Configuration Overview

Main benefit of AAP2 is the Controller uses execution environments just like developers have tested with **ansible-navigator** so playbooks can run directly on controller without modification. Initial installation will import the base container images for execution environments, but synchronization and some initial configuration is often necessary for custom environments.

1.3.2. Making Automation Execution Environments Available from Private Automation Hub

Private Automation hub provides a container registry where needed execution environment images (EEIs) can be synchronized and stored. The EEIs can also be uploaded manually to private automation hub as well as collections which we will find out about later.

1.3.2.1. Synchronizing Automation Execution Environments

In most instances, you will want to sync the supported EEIs from **https://registry.redhat.io** and get the latest supported versions for the AAP2 platform.

Synchronizing all EEIs

It is possible to synchronize all remote registries getting all versions from a remote catalog by selecting **Index execution environments** from the vertical dots icon.

1.3.2.2. Manually Adding Container Images

There are multiple ways to copy and inspect container images. The **skopeo** command is probably the best, however it can also be done with **Podman**. The benefit of **Skopeo** is that once you are logged into both the remote container registry and private automation hub, it is possible to use **skopeo copy** directly without needing to first download the container image and set the tags.

1.3.2.3. Managing Container Repositories, Images, and Tags

Management of container images within private automation hub can be done through the WebUI or using the **skopeo** command on the CLI. Images can be easily tagged and displayed here.

1.3.3. Synchronizing Ansible Content Collections

Another key piece of AAP2 is the need for content collections. Many modules that were built-in for Ansible have been moved to content collections (firewalld, podman, and networking components and filters). In order to leverage these modules, collections must be installed and available.

Collection Locations

- **Red Hat Certified (Supported) Content Collections** - https://console.redhat.com/ansible/automation-hub
- **Ansible Community (Unsupported) Content Collections provided by Ansible Galaxy** - https://galaxy.ansible.com
- **Homegrown/Manual Collections**: Manually uploaded to private automation hub

1.3.3.1. Synchronizing Red Hat Certified Ansible Content Collections

Login and Credentials required for RH Certified Collections

Red Hat Certified Ansible Collections require a multi-step process.

1. Login to Ansible Automation Platform
   a. Select Collections
   b. Click "Sync"
2. Create an Authentication Token with the **Connect to Hub**
3. Login to Private Automation Hub
   a. Collections ⇒ Repository Management and remote tab
   b. Select **rh-certified** and **Edit** to provide your token.
   c. Click **Save** and then from Repository Management page, select Sync and it will sync all collections marked as Sync.

1.3.3.2. Synchronizing Ansible Content Collections from Ansible Galaxy

Galaxy Doesn't Require Authentication

Since Ansible Galaxy doesn't require authentication, if is possible to configure a **Community** colelction or set of collections by providing a single **requirements.yml** file. This file provides a list of all content collections to synchronize.

1.3.3.3. Manually Adding Ansible Content Collections

In order to manually upload connections, you must first create a **Namespace** in private automation hub.

Collection Security and Signing

The concept of **collection signing** which is signing collection content similar to signing RPMs is currently in Tech Preview for AAP 2.2. This feature is expected to provide an additional level of security with respect to the download content and where it originated from and that it is in its intact and intended format.

1.3.4. Testing Basic Automation Controller Functionality

DEMO Project Benefits

The **Demo** project is essentially there to provide some "smoke" tests allowing a quick way to see if Controller is performing as expected.

1.3.4.1. The Demo Project

Verification of EE and Project Synchronization

The new thing that Controller needs is the ability to use EEIs. The **Demo** project in addition to testing project synchronization components is now able to verify that the EEI can be downloaded and leveraged with Controller.

1.3.4.2. Default Execution Environment Registry Credential

Registry Credential

This is a valid credential and is created based on information at install found in the **inventory** file. This credential cannot be changed from the WebUI and must be modified in the inventory file and have the **setup.sh** script executed again.

1.3.4.3. The Demo Credential

Machine Credential Doesn't Work

This is the only non-working component in the project. The **Machine** credential will need to be updated with a valid username and password or SSH key so that connections can be made to the remote hosts.

1.3.4.4. The Demo Inventory

Modify Inventory to Add Systems

Initially the **Demo** inventory only contains localhost. It should be modified to include one or more hosts from the environment. Ideally, all hosts would be added so that it is possible to verify Controller connectivity to your entire environment.

1.3.4.5. The Demo Job Template

1.3.5. DEMO: Initial Configuration of Automation Controller and Private Automation Hub

Example 2. **DEMO: Initial Configuration of Automation Controller and Private Automation Hub**

Working with Execution Environments

Manually uploading and adding container images (EEs) to Ansible Private Automation Hub.

1. Login to Registries to both Push/Pull and Copy container images

```
[student@workstation Add_EEs]$ skopeo login hub.lab.example.com
```

2. Inspect available containers and tags

```
[student@workstation Add_EEs]$ skopeo inspect docker://hub.lab.example.com/ee-29-
rhel8
```

Grabbing Tags and Release Information from the CLI

Listing 4. **skopeo inspect** *to get release and* **skopeo tags** *to get tags*

```
[student@workstation Add_EEs]$ skopeo inspect
docker://hub.lab.example.com/ee-29-rhel8 --format "{{
.Labels.version }}-{{ .Labels.release }}"
1.0.0-119

[student@workstation Add_EEs]$ skopeo list-tags
docker://hub.lab.example.com/ee-29-rhel8
```

It is also possible to use **podman** to search and list tags, but that is generally considered less reliable. It should also be noted that only **skopeo** has the ability to inspect and act with images remotely. As such, this course will leverage **skopeo** over Podman for many of the exercises.

Listing 5. **podman** *Tag Listing*

```
[student@workstation Add_EEs]$ podman search --list-tags
docker://hub.lab.example.com/ee-29-rhel8
```

The **skopeo** *Command*

Skopeo is another command that can be used with containers and was introduced as part of the **container-tools** suite with RHEL8. The **container-tools** suite installs the RHEL 8 toolchain to work with containers which includes: **podman**, **buildah**, and **skopeo**.

2. Managing User Access

2.1. Creating and Managing Automation Controller Users

Section Info Here

2.1.1. Role-based Access Controls

2.1.2. Automation Controller Organizations

2.1.3. Types of Users

2.1.4. Creating Users

2.1.5. Editing Users

2.1.6. Organization Roles

2.1.7. Managing User Organization Roles

2.2. Managing Automation Controller Access with Teams

Section Info Here

2.2.1. Teams in Automation Controller

2.2.2. Creating Teams

2.2.3. Team Roles

2.2.4. Adding Users to a Team and Assigning Team Roles

2.2.5. Organization Roles

2.2.6. Managing Organization Roles

2.3. Creating and Managing Users and Groups for Private Automation Hub

Section Info Here

2.3.1. User Access

2.3.1.1. Creating Groups

2.3.1.2. Creating Users

2.3.1.3. Creating Groups to Manage Content

3. Managing Inventories and Machine Credentials

3.1. Creating a Static Inventory

Section Info Here

3.1.1. Red Hat Ansible Inventory

3.1.2. Creating an Inventory Using the Automation Controller Web UI

3.1.2.1. Creating a New Inventory

3.1.2.2. Creating a Host Group in an Inventory

3.1.2.3. Creating Hosts in an Inventory

3.1.3. Inventory Roles

3.1.3.1. Assigning Roles

3.1.4. Inventory Variables

3.2. Creating Machine Credentials for Access to Inventory Hosts

Section Info Here

3.2.1. Storing Secrets in Credentials

3.2.2. Credential Types

3.2.3. Creating Machine Credentials

3.2.4. Editing Machine Credentials

3.2.5. Credential Roles

3.2.6. Managing Credential Access

3.2.7. Common Credential Scenarios

3.2.7.1. Credentials Protected by Automation Controller, Not Known to Users

3.2.7.2. Credential Prompts for Sensitive Password, Not Stored in Automation Controller

4. Managing Projects and Launching Ansible Jobs

4.1. Creating a Project for Ansible Playbooks

Section Info Here

4.1.1. Automation Controller Projects

4.1.2. Creating a Project

4.1.3. Project Roles

4.1.4. Managing Project Access

4.1.5. Creating SCM Credentials

4.1.6. SCM Credential Roles

4.1.7. Managing Access to SCM Credentials

4.1.8. Updating Projects

4.1.8.1. Update Revision on Launch

4.1.8.2. Manual Updates

4.1.9. Support for Ansible Content Collections and Roles

4.2. Creating Job Templates and Launching Jobs

Section Info Here

4.2.1. Job Templates

4.2.2. Creating Job Templates

4.2.3. Modifying Job Execution

4.2.4. Prompting for Job Parameters

4.2.5. Job Template Roles

4.2.6. Managing Job Template Access

4.2.7. Launching Jobs

4.2.8. Evaluating the Results of a Job

5. Advanced Job Configuration

5.1. Improving Performance with Fact Caching

Section Info Here

5.1.1. Fact Caching

5.1.1.1. Enabling Fact Caching in Automation Controller

5.2. Creating Job Template Surveys to Set Variables for Jobs

Section Info Here

5.2.1. Managing Variables

5.2.2. Defining Extra Variables

5.2.3. Job Template Surveys

5.2.3.1. Managing Answers to Survey Questions

5.2.3.2. Creating a Job Template Survey

5.3. Scheduling Jobs and Configuring Notifications

Section Info Here

5.3.1. Scheduling Job Execution

5.3.1.1. Temporarily Disabling a Schedule

5.3.1.2. Scheduled Management Jobs

5.3.2. Reporting Job Execution Results

5.3.2.1. Notification Templates

5.3.2.2. Creating Notification Templates

5.3.2.3. Enabling Job Result Notification

6. Constructing Job Workflows

6.1. Creating Workflow Job Templates and Launching Workflow Jobs

Section Info Here

6.1.1. Workflow Job Templates

6.1.2. Creating Workflow Job Templates

6.1.2.1. Using the Workflow Visualizer

6.1.2.2. Adding Multiple Nodes with the Same Relationship

6.1.2.3. Creating Convergent Nodes

6.1.2.4. Workflow Job Template Surveys

6.1.3. Launching Workflow Jobs

6.1.3.1. Evaluating Workflow Job Execution

6.2. Requiring Approvals in Workflow Jobs

Section Info Here

6.2.1. Approval Nodes

6.2.2. Adding Approval Nodes to Workflows

6.2.3. Approving and Denying Workflow Approval Requests

6.2.4. Approval Time-outs

6.2.5. Approval Notifications

7. Managing Advanced Inventories

7.1. Importing External Static Inventories

Section Info Here

7.1.1. Importing Existing Static Inventories

7.1.2. Storing an Inventory in a Project

7.2. Configuring Dynamic Inventory Plug-ins

Section Info Here

7.2.1. Dynamic Inventories

7.2.2. OpenStack Dynamic Inventories

7.2.3. Red Hat Satellite 6 Dynamic Inventories

7.3. Filtering Hosts with Smart Inventories

Section Info Here

7.3.1. Defining Smart Inventories

7.3.2. Using Ansible Facts in Smart Inventory Filters

7.3.2.1. Creating a Smart Inventory Based on Ansible Facts

7.3.3. Other Smart Inventory Filters

8. Automating Configuration of Ansible Automation Platform

8.1. Configuring Red Hat Ansible Automation Platform with Collections

Section Info Here

8.1.1. Automating Red Hat Ansible Automation Platform Configuration

8.1.2. Getting the Supported Ansible Content Collection

8.1.3. Exploring the Supported Ansible Content Collection

8.1.3.1. Reading Documentation with Ansible Content Navigator

8.1.3.2. Reading Documentation on Automation Hub

8.1.4. Examples of Automation with ansible.controller

8.1.4.1. Creating Automation Controller Users

8.1.4.2. Creating Automation Controller Teams

8.1.4.3. Adding Users to Organizations and Teams

8.1.5. Community-supported Ansible Content Collections

8.2. Automating Configuration Updates with Git Webhooks

Section Info Here

8.2.1. Introducing Red Hat Ansible Automation Platform Webhooks

8.2.1.1. What Are the Benefits of Webhooks

8.2.2. Configuring Webhooks

8.2.2.1. Configuring a Webhook for a Job Template

8.2.2.2. Creating the Webhook for the Repository in GitLab

8.2.3. Use Cases for Using Webhooks

8.2.3.1. Triggering Different Job Templates Using Branches

8.2.3.2. Configuration as Code for Automation Controller

8.3. Launching Jobs with the Automation Controller API

Section Info Here

8.3.1. The Automation Controller REST API

8.3.1.1. Using the REST API

8.3.1.2. JSON Pagination

8.3.1.3. Accessing the REST API From a Graphical Web Browser

8.3.2. Launching a Job Template Using the API

8.3.3. Launching a Job Using the API from an Ansible Playbook

8.3.3.1. Vault Credentials

8.3.4. Token-based Authentication

9. Maintaining Red Hat Ansible Automation Platform

9.1. Performing Basic Troubleshooting of Automation Controller

Section Info Here

9.1.1. Automation Controller Components

9.1.1.1. Starting, Stopping, and Restarting Automation Controller

9.1.1.2. Supervisord Components

9.1.2. Automation Controller Configuration and Log Files

9.1.2.1. Configuration Files

9.1.2.2. Log Files

9.1.2.3. Other Automation Controller Files

9.1.3. Common Troubleshooting Scenarios

9.1.3.1. Problems Running Playbooks

9.1.3.2. Problems Connecting to Your Host

9.1.3.3. Playbooks Do Not Appear in the List of Job Templates

9.1.3.4. Playbook Stays in Pending State

9.1.3.5. Error: Provided Hosts List Is Empty

9.1.4. Performing Command-Line Management

9.1.4.1. Changing the Automation Controller Admin Password

9.2. Backing Up and Restoring Red Hat Ansible Automation Platform

Section Info Here

9.2.1. Backing Up Red Hat Ansible Automation Platform

9.2.1.1. Backup Procedure

9.2.2. Restoring Ansible Automation Platform From Backup

9.2.2.1. Restoration Procedure

10. Getting Insights into Automation Performance

10.1. Gathering Data for Cloud-based Analysis

Section Info Here

10.1.1. Introducing Red Hat Hybrid Cloud Console Services

10.1.2. Collecting Data for Cloud Services

10.1.3. Registering Managed Hosts with Insights for Ansible Automation Platform

10.1.4. Accessing the Red Hat Hybrid Cloud Console

10.2. Getting Insights into Automation Performance

Section Info Here

10.2.1. Insights for Ansible Automation Platform

10.2.2. Generating Remediation Playbooks with Advisor

10.2.2.1. Automating Remediation of an Issue for Multiple Systems

10.2.2.2. Automating Remediation of Multiple Issues for One System

10.2.3. Comparing Systems with Drift

10.2.3.1. Finding Differences Between Systems

10.2.3.2. Comparing the State of One System at Different Times

10.2.3.3. Comparing Systems to a Standard Baseline

10.2.4. Sending Alerts Based on Ansible Facts with Policies

10.3. Evaluating Performance with Automation Analytics

Section Info Here

10.3.1. Automation Analytics

10.3.2. Reporting Playbook Execution Status

10.3.3. Examining Job History

10.3.4. Monitoring Notifications

10.4. Producing Reports from Automation Analytics

Section Info Here

10.4.1. Producing Reports from Automation Analytics

10.4.1.1. Choosing an Appropriate Report

10.4.1.2. Using Automation Calculator to Compute Savings

10.4.1.3. Exporting a Report

10.4.2. Predicting the Cost Savings of Automation

10.4.2.1. Creating a Savings Plan

10.4.2.2. Reviewing the Cost Savings Calculations

11. Building a Large Scale Red Hat Ansible Automation Platform Deployment

11.1. Designing a Clustered Ansible Automation Platform Implementation

11.1.1. Running Red Hat Ansible Automation Platform at Scale

11.1.2. Automation Mesh

11.1.2.1. Benefits of Automation Mesh

11.1.2.2. Types of Nodes on Automation Mesh

11.1.2.3. What Are Instance Groups?

11.1.3. Planning Network Communication and Firewalls

11.1.3.1. Requirements for Control Nodes and Hybrid Nodes

11.1.3.2. Requirements for Hop Nodes

11.1.3.3. Requirements for Execution Nodes

11.1.4. Planning for Automation Mesh

11.1.4.1. Providing Resilient Services

11.2. Deploying Distributed Execution with Automation Mesh

11.2.1. Configuring Automation Mesh

11.2.1.1. Creating Instance Groups

11.2.1.2. Adding Nodes to the Automation Mesh

11.2.1.3. Removing Nodes from the Automation Mesh

11.2.2. Visualizing Automation Mesh Topology

11.2.3. Automation Mesh Design Patterns

11.2.4. Validation Checks

11.3. Managing Distributed Execution with Automation Mesh

11.3.1. Managing Instance Groups in Automation Controller

11.3.1.1. Creating Instance Groups

11.3.1.2. Assigning Execution Nodes to an Instance Group

11.3.1.3. Running a Health Check on the Nodes

11.3.1.4. Disassociating a Node from an Instance Group

11.3.2. Assigning Default Instance Groups to Inventories and Job Templates

11.3.2.1. Configuring an Inventory to Use Instance Groups

11.3.2.2. Configure a Job Template to Use Instance Groups

11.3.2.3. Running a Job Template with Instance Groups

11.3.3. Testing the Resilience of Automation Mesh

11.3.3.1. Testing Control Plane Resilience

11.3.3.2. Testing Execution Plane Resilience

11.3.4. Monitoring Automation Mesh from the Web UI

11.3.5. Monitoring Automation Mesh from the Command Line

11.3.5.1. Listing Nodes and Instance Groups

11.3.5.2. Monitoring Automation Mesh Using the receptorctl Command

Appendix A: References and Additional Information

Ansible Docs/Tips and Tricks

- **Installing Software and other Packages**: https://ansible-tips-and-tricks.readthedocs.io/en/latest/os-dependent-tasks/installing_packages/
- **Ansible Tips and Tricks (Examples)**: https://github.com/nfaction/ansible-tips-and-tricks/wiki
- **Ansible Product Demos**: https://github.com/ansible/product-demos
- **Ansible Workshops**: https://github.com/ansible/workshops/tree/devel/provisioner
- **Red Hat CoP - Automation Good Practices**:
  - https://redhat-cop.github.io/automation-good-practices/
  - https://github.com/redhat-cop/automation-good-practices/
- **Ansible Controller Collection**: https://console.redhat.com/ansible/automation-hub/repo/published/ansible/controller/docs?keywords=

Ansible KB Articles and Solutions

- **How Do I Perform Security Patching / OS Package Upgrades On Ansible Tower/Automation Controller Nodes Without Breaking Any Ansible Tower/Automation Controller Functionality ?**: https://access.redhat.com/solutions/4566711

Ansible Filters and Collections

- **Using filters to manipulate data (Jinja2 Templating)**: https://docs.ansible.com/ansible/latest/user_guide/playbooks_filters.html
- **Community General**: https://docs.ansible.com/ansible/latest/collections/community/general/index.html

Ansible Blogs and Articles

- **When localhost isn't what it seems in Red Hat Ansible Automation Platform 2**: https://www.ansible.com/blog/when-localhost-isnt-what-it-seems-in-red-hat-ansible-automation-platform-2

Ansible Execution Environments

- **Execution Environments**: https://docs.ansible.com/automation-controller/4.2.0/html/userguide/execution_environments.html#ee-mount-options