Oracle Linux Automation Manager 2.3

User's Guide

G32925-02
August 2025

Oracle Linux Automation Manager 2.3 User's Guide,

G32925-02

Contents

4 Using Views

Preface

Oracle Linux Automation Manager 2.3: User's Guide describes how to use Oracle Linux Automation Manager, which is a task engine management platform with a Web user interface.

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at https://www.oracle.com/corporate/accessibility/.

Access to Oracle Support for Accessibility

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit https://www.oracle.com/corporate/accessibility/learning-support.html#support-tab.

Diversity and Inclusion

Oracle is fully committed to diversity and inclusion. Oracle respects and values having a diverse workforce that increases thought leadership and innovation. As part of our initiative to build a more inclusive culture that positively impacts our employees, customers, and partners, we are working to remove insensitive terms from our products and documentation. We are also mindful of the necessity to maintain compatibility with our customers' existing technologies and the need to ensure continuity of service as Oracle's offerings and industry standards evolve. Because of these technical constraints, our effort to remove insensitive terms is ongoing and will take time and external cooperation.

1

About Oracle Linux Automation Manager

Oracle Linux Automation Manager provides features that let an organization manage infrastructure configuration through a browser user interface. Using Oracle Linux Automation Manager you can:

- Create and configure permissions for organizations, users, and teams that use Oracle Linux Automation Manager. You must have at least one organization before you can create a project.
- Create a project in Oracle Linux Automation Manager that imports Oracle Linux Automation Engine playbooks from various sources, such as the local machine which runs Oracle Linux Automation Manager or from a remote git repository.
- Create an inventory which specifies the hosts where playbook tasks run. You can do this manually or import an Oracle Linux Automation Engine inventory file.
- Create credentials which contains SSH authentication details for the hosts where the playbook tasks run and any required credentials for Oracle Linux Automation Engine playbooks in remote repositories specified in projects.
- Create job templates that specify the projects and playbooks to run.
- Manually launch or schedule the launch of job templates and review the results.

The following example gives an overview of the steps a system administrator might follow to set up users, a team, and an organization to manage the running of playbooks on an inventory of target machines:

> **Note**
>
> The following outline is an overview only. The individual chapters in this book discuss the creation of each resource in more detail.

1. Log in to Oracle Linux Automation Manager with the administrator user account created during the installation process.

   See Oracle Linux Automation Manager 2.3: Installation Guide for more details on the installation process.
2. Create an **organization** called, for example, `Organization_Europe_Production`.
3. Create **users** of `Normal User` type, for example:
   - `NormalUser_Senior`.
   - `NormalUser_1` and `NormalUser_2`.
4. Create a **team**, called for example, `Team_Europe_Production`.
5. Add all the users created in the previous steps to `Team_Europe_Production`.
6. Navigate back to `Organization_Europe_Production`, the organization created in an earlier step.

7. Add team `Team_Europe_Production` to `Organization_Europe_Production` with an `Execute` role.

   Each team member of `Team_Europe_Production` is added to `Organization_Europe_Production` with an `Execute` role.

8. In addition to the `Execute` role that user `NormalUser_Senior` receives in the previous step, you might want `NormalUser_Senior` to also have administrative privileges within the `Organization_Europe_Production` resource. If so, add user `NormalUser_Senior` to the organization again, this time as an individual user, and specify the `Admin` role.

   Upon completion of this step, the user `NormalUser_Senior` is displayed with both `Admin` and `Execute` roles in their Roles list.

9. Create a **project** with **playbooks**. For example, you might create a project called `Project_Europe_Production` with playbooks `Playbook_1`, `Playbook_2`, and `Playbook_3`.

   A project must belong to an organization, so assign the project to `Organization_Europe_Production`.

10. Create an **Inventory**. For example, `Inventory_Europe_Production_Servers`.

11. Create **hosts**, for example, `Target_Host_1`, `Target_Host_2`, and `Target_Host_3`.

    Each host must belong to an inventory, so add each host to `Inventory_Europe_Production_Servers` created in the previous step.

12. In a clustered installation, you might create an **instance group** of nodes, called for example, `Instance_Group_Production_Nodes`, to run playbooks and jobs on certain inventories.

    See Oracle Linux Automation Manager 2.3: Installation Guide for more details on the installation process and different nodes in the service mesh.

13. Create a **job template** called, for example, `Job_Template_Production_1`. Steps to complete the creation of a job template include the following:

    - In the **Project** field, specify the project created in a previous step, `Project_Europe_Production`.
    - After a Project has been specified, the playbooks associated with the project become available in the **Playbook** drop down list. Pick one of its playbooks, for example `Playbook_1`.
    - In the **Instance Group** field, select `Instance_Group_Production_Nodes`, the instance group created in a precious step created in a previous step.
    - Click **Save** upon completion of all necessary fields.

14. **Launch** the playbook or create a **schedule** for it to run later.

2

Setting Up Permissions for Organizations, Teams, and Users

This chapter describes how Oracle Linux Automation Manager lets administrators create organizations, teams, and users where permissions can be allocated at each level. These permissions are based on role-based access controls.

> **Note**
>
> You can integrate the Oracle Linux Automation Manager access levels discussed in this chapter with external identity management services, such as LDAP. Note that LDAP user account information doesn't appear in Oracle Linux Automation Manager until after the LDAP user account first logs in to Oracle Linux Automation Manager. See Oracle Linux Automation Manager 2.3: Administrator's Guide for more information about LDAP authentication and mappings for users, teams, and organizations.

Each level has the following functions:

- **Organizations**: Administrators can specify which organizations can run a playbook on what inventory by associating an organization to a project and an inventory. An organization can specify several projects and inventories, but each project and inventory can specify only one organization.
- **Teams**: A team belongs to one organization and a team can specify default permissions that apply to any user assigned to the team.
- **Users**: A user can belong to one or more organizations or teams. Administrators can create the following types of users:
  - **System Administrator**

    By default, a system administrator has system-wide administrator privileges.
  - **System Auditor**

    By default, a system auditor has read-only access for auditing purposes.
  - **Normal User**

    By default, a normal user has limited privileges. However, a normal user can be assigned high-privilege roles, such as **Admin**, or **Project Admin**, for a specific resource type.

    For example, you might create a normal user called `Payroll_Engineer` and assign that user with an **Admin** system role to organization resource types called `Payroll_Organization` and `Finance_Organization`. In addition, you might also add the `Payroll_Engineer` user with an **Admin** system role to a team resource type called `Payroll_Team`.

    Thus, a user might have different permissions depending on which resource type they're working with.

> **Note**
>
> - For purposes of security, We recommend following the **principle of least privilege** when setting up user access.
> - For illustration, in this book we request you follow the examples with a system administrator account, as this allows the examples to cover the features available in Oracle Linux Automation Manager.

The following table describes the different roles available within each resource in Oracle Linux Automation Manager.

Table 2-1 Role-Based Access Control Role Descriptions

Resource Type	System Role	Description
Job Templates	Admin	Manages all aspects of the job template.
	Executive	Runs the job template.
	Read	Views settings for the job template.
Workflow Job Templates	Admin	Manages all aspects of the workflow job template.
	Executive	Runs the workflow job template.
	Read	Views settings for the workflow job template.
	Approve	Can approve or deny a workflow approval node.
Credentials	Admin	Manages all aspects of the Credential.
	Use	Use the credential in a job template.
	Read	Views settings for the credential.
Inventories	Admin	Manages all aspects of the inventory.
	Update	Updates the inventory.
	Ad Hoc	Runs ad hoc commands on the inventory.
	Use	Use the inventory in a job template.
	Read	Views settings for the inventory.
Projects	Admin	Manages all aspects of the project.
	Update	Updates the project.
	Use	Use the project in a job template.
	Read	Views settings for the project.

Table 2-1 (Cont.) Role-Based Access Control Role Descriptions

Resource Type	System Role	Description
Organizations	Admin	Manages all aspects of the organization.
	Executive	Runs the executable resources in the organization.
	Project Admin	Manages all projects in the organization.
	Inventory Admin	Manages all inventories in the organization.
	Credential Admin	Manages all credentials in the organization.
	Workflow Admin	Manages all workflows in the organization.
	Notification Admin	Manages all notifications in the organization.
	Job Template Admin	Manages all job templates in the organization.
	Execution Environment Admin	Manages all execution environments in the organization.
	Auditor	Views all aspects of the organization
	Member	Makes user a member of the organization.
	Read	Views all organization settings.
	Approve	Approves or denies a workflow approval node.

Setting Up Organizations

The Oracle Linux Automation Manager setup requires that you set up at least one Organization resource.

To set up an organization, do the following:

1. Log in to Oracle Linux Automation Manager with an administrator user account.
2. Display the left navigation menu if it's not already visible by switching the Global navigation menu button in the top-left corner of the page.
3. From the **Access** section, click **Organizations**.

   The Organizations page appears.
4. Click the **Add** button.

   The Create New Organization page appears.
5. In the **Name** field, enter a name for your organization. For example, `Organization 1`.
6. In the **Instance Groups** field, click the search button.
   The Select Instance Groups dialog appears.

7. Select the checkbox next to each instance group you want to add.
8. Click **Select**.
9. In the **Galaxy Credentials** field, click the search button.
   The Select Galaxy Credentials dialog appears.
10. Select the checkbox next to each credential for https://galaxy.ansible.com/ or Private Automation Hub, or both for accessing collections when running playbooks.
    When selecting more than one credential, Oracle Linux Automation Manager checks each server in the order in which the credentials are added to the Organization.
11. Click **Select**.
12. Click **Save**.

    The newly created organization is displayed in the Details tab.
13. Click the **Access** tab.
14. Click the **Add** button.

    The Add Roles dialog appears.
15. Select the **Users** option.
16. Click **Next**.
17. Select the checkbox next to the user accounts you want to add to the organization.
18. Click **Next**.
19. Select the checkbox next to the roles you want to assign to the users you have selected.
20. Click **Save**.

Setting Up Teams

To set up a team, do the following:

1. Log into Oracle Linux Automation Manager with an administrator user account.
2. Display the left navigation menu if it's not already visible by switching the Global navigation menu button in the top-left corner of the page.
3. From the **Access** section, click **Teams**.

   The Teams page appears.
4. Click the **Add** button.

   The Create New Team page appears.
5. In the **Name** field, enter a name for the team. For example, `Team 1`.
6. In the **Organization** field, click the **search** button.

   The Select Organization dialog appears.
7. From the Organization list, select an organization.
8. Click **Select**.
9. Click **Save**.

   The newly created team is displayed in the Details tab.
10. Click the **Access** tab.
11. Click the **Add** button.

The Add Roles dialog appears.

12. Select the **Users** option.
13. Click **Next**.
14. Select the checkbox next to the user accounts you want to add to the team.
15. Click **Next**.
16. Select the checkbox next to the roles you want to assign to the users you have selected.
17. Click **Save**.

Setting Up Users

To set up a user, do the following:

1. Log into Oracle Linux Automation Manager with an administrator user account.
2. From the **Access** section, click **Users**.

   The Users page appears.
3. Click the **Add** button.

   The Create New User page appears.
4. Optionally, complete the **First Name**, **Last Name** and **Email** fields.
5. In the **UserName** field, enter a username for a user. For example, `User1`.
6. In the **Password** field, enter a password.
7. In the **Confirm Password** field, reenter the password.
8. From the User Type, select one of the following user types:
   - **Normal User**: You can limit users read and write access to the resources (such as inventory, projects, and so on) based on roles and privileges.
   - **System Auditor**: You can limit users to read-only permissions for all objects within Oracle Linux Automation Manager.
   - **System Administrator**: You can allow full system administration privileges (full read and write) for all objects within Oracle Linux Automation Manager.
9. In the **Organization** field, click the **search** button.

   The Select Organization dialog appears.
10. From the organization list, select an organization.
11. Click **Select**.
12. Click **Save**.

    The newly created user is displayed in the Details tab.
13. Click the **Organizations** button.

    All organizations that the user is part of appear in the list. This is a read-only page.
14. Click the **Teams** tab.
15. Click the **Associate** button.

    The Select Teams page appears.
16. Select the checkbox next to the user teams you want to associate with the user.

17. Click the **Save**.
18. Click the **Roles** tab.
19. Click the **Add** button.

    The Add User Permissions dialog is displayed.
20. Select one of the following resource types:
    - **Job Templates**
    - **Workflow job templates**
    - **Credentials**
    - **Inventories**
    - **Projects**
    - **Organizations**
21. Click **Next**.

    The available options of the resource type you have selected are displayed. For example, if you have selected Job templates, then all available job templates appear.
22. Select the checkbox next to each resource item you want to add to the user.
23. Click **Next**.

    The available roles you can apply to the resources you have selected are displayed.
24. Select the checkbox next to each role you want to apply to the resources you have selected.

    > **Note**
    >
    > The roles you opt to apply are applied to all the resources you have selected.
    >
    > For more information about available roles by resource type, see Setting Up Permissions for Organizations, Teams, and Users
25. Click **Save**.

3
Managing Resources

This chapter describes managing resources. Resources are the primary configuration touch points between Oracle Linux Automation Manager and playbooks and inventory. These resources are projects, inventories, credentials, job templates, and workflow templates.

Managing Projects

A Project is a resource that represents a repository of Oracle Linux Automation Engine playbooks. The repository can be local or on a remote source code management (SCM) system such as Git.

To create a project, do the following:

1. Log into Oracle Linux Automation Manager with an administrator user account.
2. Display the navigation menu if it's not already visible by switching the Global navigation menu button in the top-left corner of the page.
3. From the **Resources** section, click **Projects**.

   The Projects page appears.
4. Click the **Add** button.

   The Create New Project page appears.
5. In the **Name** field, enter a name. For example, `Project1`.
6. In the **Organization** field, click the **search** button.

   The Select Organization dialog appears.

   If the project has ansible collection requirements, you must ensure you select an organization that lets you use one or more of the following options:

   - **Download Collections from Repositories at Runtime**

     To use this option, you must select an organization that has a credential set up for each repository you need to download collections from. See Setting Up Ansible Galaxy/Automation Hub API Token Credentials for more information.
   - **Utilize a Custom Execution Environment Hosted in Private Automation Hub**

     To use this option, you must select an organization with an execution environment whose **Image** property references a Private Automation Hub custom execution environment containing ansible collections.

   From the organization list, select an organization.
7. Click **Select**.
8. From the **Source Control Credential Type** list, select one of the following:

   - **Manual**:

     The **Manual** option lets you select a local playbook on the machine hosting Oracle Linux Automation Manager. You select the playbook from a drop-down list labeled **Playbook Directory**.

See Accessing a Local Playbook for more information.

- **Git**:

  The **Git** options lets you use a playbook from a remote repository. Selecting one of these options displays source control fields for you to complete so that you can access a chosen remote repository.

  For example, go to Accessing a Remote Playbook on Git for information about setting up source details and credentials.

- **Subversion**:
  The **Subversion** options lets you use a playbook from a remote repository. Selecting one of these options displays source control fields for you to complete so that you can access a chosen remote repository.

- **Remote Archive**:
  The **Remote Archive** options lets you use a playbook from a remote archive. Selecting one of these options displays source control fields for you to complete so that you can access a chosen remote repository.

9. Click **Save**.

Accessing a Local Playbook

To make a local playbook accessible to an instance of Oracle Linux Automation Manager, ensure you have the required administrative privileges and perform the following steps:

1. Log onto the server hosting Oracle Linux Automation Manager.
2. Create a directory to store the playbooks. The default location is `/var/lib/ol-automation-manager/projects`.

> **Note**
>
> The Oracle Linux Automation Manager might list the Project Base Path as `/var/lib/awx/projects` instead of `/var/lib/ol-automation-manager/projects`. Both locations are the same as `/var/lib/awx` is a symbolic link to `/var/lib/ol-automation-manager`.

3. Copy or create playbook files in the directory. For sample playbooks, see `/var/lib/ol-automation-manager/projects/demo_bundled`.
4. Set the playbook directory and files to the same user and group and permissions that Oracle Linux Automation Manager uses.

> **Note**
>
> Check the SELinux permissions and context settings if you have issues adding a project path.

Accessing a Remote Playbook on Git

Git is a distributed version control system containing repositories where you can store playbooks that you can clone to Oracle Linux Automation Manager systems as part of projects.

To access a remote playbook on Git, do the following:

1. Set up a new Git repository or use an existing Git repository on a remote Git server.

   For example, GitHub hosts the public Ansible example projects at https://github.com/ansible/ansible-tower-samples.

   > **Caution**
   >
   > As this is a public repository, Oracle isn't responsible for examples contained therein.

2. If you're using a private repository that requires SSH authentication, you must set up SSH private and public keys on the system running Oracle Linux Automation Manager and copy the public key to your Git user account.
3. Return to the Oracle Linux Automation Manager UI and from the **Resources** section, click **Projects**.
   The Projects page appears.
4. Click the name of the project you want to change, then click **Edit**.
5. Confirm that the **Source Control Type** is set to Git, then enter a URL to the SCM repository in the **Source Control URL** field.
6. If you need to specify an SCM branch, tag, or commit in the **Source Control Branch/Tag/Commit** field, enter a branch, tags, or commit hash.
7. If you need to retrieve a refspec from a branch, in the **Source Control Refspec** field, enter a refspec.
8. If the Git repository you specified is private and requires credentials to access, from the **Source Control Credential** list, select a credential. For more information about setting up credentials, see Setting Up Credentials.
9. Click **Save**.
10. If Oracle Linux Automation Manager requires access to the Git repository through a proxy, expand the left side menu and click **Settings**.

    The Settings page appears.
11. Click **Jobs settings**.

    The Details page appears.
12. Click the **Edit** button.
13. In the **Extra Environment Variables** field, enter the following:

    ```
    {
     "https_proxy": "proxy_address:proxy_port"
    }
    ```

    In the previous example, *proxy_address* is the proxy address and *proxy_port* is the port for the proxy.
14. Click **Save**.

    > **Note**
    >
    > If a playbook uses the OCI Ansible collection, see https://docs.oracle.com/iaas/Content/API/SDKDocs/ansible.htm and find the setup instructions relating to AWX.

Managing Inventories, Groups, and Hosts

The following sections provide information about managing inventories and hosts.

Managing Inventories and Groups

An inventory is a collection of hosts against which jobs and commands can be run. You can create groups within inventories and further subdivide these groups into subgroups.

Creating an inventory

To set up an inventory, do the following:

1. Log into Oracle Linux Automation Manager with an administrator user account.
2. From the **Resources** section, click **Inventories**.

   The Inventories page appears.
3. Click the **Add** button.

   A menu appears.
4. Click the **Add Inventory** option.

   The Create new inventory page appears.
5. In the **Name** field, enter a name for the inventory. For example, `Inventory1`.
6. In the **Organization** field, click the **search** button.

   The Select Organization dialog appears.
7. From the Organization list, select an organization.
8. Click **Select**.
9. In the **Variables** field (a code box), you can enter YAML or JSON code to specify any inventory variables by doing the following:
   a. Click the **YAML** or the **JSON** button to specify which syntax to use.
   b. Type the variables in the code edit box.
10. Click **Save**.

Adding groups or subgroups to an inventory

To add a group or subgroup, or both, do the following:

1. Click **Groups**.
   A list of Groups appears.
2. Click **Add**.
   The Create new group dialog appears.
3. In the Name field, add a name.
4. Click **Save**.
   The Group details page appears.
5. Click **Related Groups**.
   The Related Groups page appears.
6. Click **Add**.
7. Do one of the following:

- To add an existing group, click **Add existing group** and choose one or more existing groups to relate to the existing group.
- To add a new group, click **Add new group** and create a group that is related to the current group.

8. Click **Save**.

Managing Hosts

To create a host, do the following:

1. Log into Oracle Linux Automation Manager with an administrator user account.
2. From the **Resources** section, click **Hosts**.

   The Hosts page appears.
3. Click the **Add** button.

   The Create New Host page appears.
4. In the **Name** field, enter the IP address or host name.
5. In the **Inventory** field click the **search** button.
   The Select Inventory page appears.
6. From the inventory list, select an inventory.
7. Click **Save**.

   The Details page appears.
8. Set the host to **On** or **Off** to indicate whether the host is available for jobs or not.
9. Click **Facts** to review any facts gathered for the host from jobs run on the host. For more information about enabling fact storage, see Setting Up a Job Template.
10. Click **Groups**.
    A list of groups appears, if any are configured.
11. Click **Add** to add a group.
12. Select one or more groups from the list to add the host to the group. The groups that can be selected are created in the inventory that the host is associated with. For more information about creating groups, see Managing Inventories and Groups.
13. Click **Save**.

Running One-Off Commands Against a Host or Group

To run single commands using Oracle Linux Automation Engine built-in modules against one or more hosts or group of hosts in an inventory (similar to running a module on the command line), do the following:

1. Log into Oracle Linux Automation Manager with an administrator user account.
2. From the **Resources** section, click **Inventories**.

   The Inventories page appears with a list of available inventories.
3. Click the name of the inventory containing the hosts against which you want to run a command.
   The inventory is displayed in the Details Tab.
4. Do one of the following:

- Click the **Hosts** tab to run commands on one or more hosts.
- Click the **Groups** tab to run commands on one or more groups of hosts.

5. Select the checkbox beside one or more hosts or groups on the left side.

> **Note**
>
> If you don't select any of the check boxes the system assumes you want to run the command against all the hosts or groups.

6. Click the **Run Command** button.

   The Run Command page appears.

7. From the **Modules** list, select a module. For example, choose a module shell.
8. In the **Arguments** field, enter any arguments that are required by the selected module. For example, `uname -a`.
9. Click **Next**.

   A list of available execution environments is displayed.

10. Select the **Execution Environment**.

    If you don't select an environment, the command runs in the `OLAM EE` execution environment by default.

11. Click **Next**.

    A list of available machine credentials is listed.

12. Select the **Machine Credential** for the machines. See Setting Up Credentials for information about setting up credentials.
13. Click **Launch**.

    The Output page appears showing the job operations in real-time and eventually displays a status of **Success** of **Failed**.

Managing Smart Inventories

A smart inventory is a type of inventory where hosts are automatically grouped based on filter criteria rather than manually defined groups. Host filter criteria can be based on attributes such as host names, group memberships, or based on host facts collected and stored by a previously run job that collected facts about the host. The inventory is updated every time a job runs that uses a smart inventory.

To set up a smart inventory, do the following:

1. Log into Oracle Linux Automation Manager with an administrator user account.
2. From the **Resources** section, click **Inventories**.

   The Inventories page appears.

3. Click the **Add** button.

   A menu appears.

4. Click the **Add Smart Inventory** option.

   The Create new smart inventory page appears.

5. In the **Name** field, enter a name for the smart inventory.

6. In the **Organization** field, click **search**.

   The Select Organization dialog appears.

7. From the Organization list, select an organization.

8. Click **Select**.

> **Note**
>
> You can only create smart inventories using inventories that are part of the same organization.

9. In the **Smart host filter** field, click the search icon.
   The Perform a search to define a host filter dialog appears.

10. To perform a search based on information configured in or generated by Oracle Linux Automation Manager, select one of the following:

    - **Name**: Enter either the full name of the host or a part of a name that might be shared by more than one host. For example, if you had several hosts where each name started with either `eastern_region` or `western_region`, you could select only those hosts in the `eastern_region` by entering `eastern_region` in this field.
    - **Groups**: Enter either the full group name or part of the group name. For example, if you have one host in `group1` and another in `group2`, entering `group` displays both hosts. Entering `group1` enters only hosts in `group1`. For more information about creating groups, see Managing Inventories and Groups.
    - **Enabled**: Set this to **Yes** to show all hosts available for jobs. Set it to **No** to display all hosts unavailable for jobs.
    - **Inventory ID**: Enter the inventory ID for an inventory. You can find the inventory ID for inventories using the `api/v2/inventories/` REST API endpoint.
    - **Instance ID**: Enter the instance ID for a control or execution node instance. You can find the instance ID for inventories using the `api/v2/instances/` REST API endpoint.
    - **Last job**: Enter the job ID of a job that ran against one or more hosts. Each host that the job ran against appears. To obtain the job ID, see Viewing Jobs.
    - **Advanced**: Perform more fine grained searches based on set types, various keys, and lookup types such as `exact`, `startswith`, and so on.

11. To perform an advanced search based on stored facts generated from running playbooks on existing hosts, do the following:

    a. Ensure a job has run to collect and store facts about the host to which this smart inventory applies. For more information about storing facts, see Setting Up a Job Template. For more information about viewing stored facts, see Managing Hosts.

    b. From the list, select **Advanced**.

    c. From the Key list, select **ansible_facts**.

    d. In the search field, enter the fact you want to search on. Valid parameters are:

       - ( ) for groupings,
       - and for adding two groupings together,
       - __ to reference related fields in a relational field.
       - __ to indicate separate ansible_facts.

- `[]` to indicate a JSON array in the path.

For example, the following filters illustrate some of these parameters:

- The following lists all Oracle Linux based hosts.

  ```
  ansible_distribution="OracleLinux"
  ```

- The following lists all hosts within the IPv4 `100.102.112.0` network. Within the host facts, this value is specified by two parameters: `ansible_default_ipv4` and the nested parameter `network`. The use of the __ parameter in the search indicates that these are distinct but related parameters.

  ```
  ansible_default_ipv4__network="100.102.112.0"
  ```

- The following lists all hosts using the `AuthenticAMD` processor. Within the host facts, this value is part of an array and appears within brackets `[]` after the `ansible_processor` parameter, and so brackets `[]` are used in the search for this kind of value.

  ```
  ansible_processor[]="AuthenticAMD"
  ```

12. After defining the search, click **Select**.
    The search appears within the Smart host filter field. Click the search icon again to add or remove searches.
13. In the **Variables** field (a code box), you can enter YAML or JSON code to specify any inventory variables by doing the following:
    a. Click the **YAML** or the **JSON** button to specify which syntax to use.
    b. Type the variables in the code edit box.
14. Click **Save**.
15. You can optionally configure Oracle Linux Automation Manager to update the memberships of smart inventories more frequently, by adding the following setting to the /etc/tower/conf.d/<*configuration file*>.py file (where <*configuration file*> is the name of the configuration file) on the control or hybrid nodes.

    ```
    AWX_REBUILD_SMART_MEMBERSHIP=True
    ```

    This updates memberships in the following events:

    - When a new host is added
    - When an existing host is modified (updated or deleted)
    - When a new Smart Inventory is added
    - When an existing Smart Inventory is modified (updated or deleted)

    > **Note**
    >
    > Enabling this parameter increases the load on the system.

Managing External Inventory Sources

You can import inventory hosts and group information from an external source such as a git repository associated with a project, from Oracle Cloud Infrastructure (OCI), or from Oracle Linux Virtualization Manager. You can optionally synchronize an inventory with an external source in the following ways:

- Manually by using the UI,
- Automatically according to a schedule,
- Automatically every time a job runs that uses the inventory.

Creating a external inventory source

To create an external inventory, do the following:

1. Log into Oracle Linux Automation Manager with an administrator user account.
2. Create an inventory as described in Managing Inventories and Groups.
3. If you are sourcing the inventory from a project, create a project configured with access to an inventory (for example, a git inventory) as described in Managing Projects.
4. From the **Resources** section, click **Inventories**.

   The Inventories page appears.
5. Select the inventory you created.
6. Click **Sources**.
7. Click **Add**.
   The Create new source page appears.
8. In the Name field, enter a name.
9. From the Execution Environment list, click the search button and select an execution environment to use when running the job that imports the inventory from the remote repository.

   > **Note**
   >
   > If you leave this field blank, the default execution environment is used. For more information about default execution environments, see Oracle Linux Automation Manager 2.3: Release Notes.

10. From the Source list, select one of the following:
    - To source the external inventory from a project, select **Sourced from a Project**.
    - To source the external inventory from OCI, select **Oracle Cloud Infrastructure**.

      > **Note**
      >
      > You can also access an OCI inventory from a project, but you must access the inventory using a python script, as described in Inventory File Formats

    - To source the external inventory from Oracle Linux Virtualization Manager, select **Oracle Linux Virtualization Manager**.

A new section appears called Source details.

11. From the Credential list, click the search button and select a credential to use, if required. For example, you could use an OCI or Oracle Linux Virtualization Manager credential. For more information, see Setting Up an Oracle Cloud Infrastructure Credentials and Setting Up Oracle Linux Virtualization Manager Credentials.
12. If you selected to source from a project, do the following:
    a. Click the search button and select a project to use that includes a git repository which includes an inventory.
    b. In the Inventory File field, enter the path and name of the inventory file. For more information about valid inventory formats, see Inventory File Formats.

    > **Note**
    >
    > The inventory path begins at the root directory of the git repository. If the inventory file is at the root level of the repository, enter the name of the repository file. If the inventory file is within one or more folders, enter the name of the first folder without a slash, then include slashes before each sub folder. For example, `inventory/north_america/inventory.yaml`.

    c. As you type the path and name of the inventory file, the **Set source to path to** field appears and duplicates the inventory path and file name. When you have finished typing in the inventory path and file name, click this field to select the inventory.
13. From the Verbosity list, enter a verbosity level. This decides the level of output the inventory job update produces. Valid values are:
    - **0 (Warning)**
    - **1 (Info)**
    - **2 (Debug)**
14. From the Update options area, select the following if required:
    - **Overwrite**: Enable this option to remove hosts and groups from the inventory that are no longer present in the external source. Unmanaged hosts and groups move to the next manual group or stay in the "all" group if none exist. If disabled, unmanaged hosts and groups remain unchanged during updates.
    - **Overwrite variables**: Enable this option to replace all variables for child groups and hosts with those from the external source. Disable it to merge local variables with external ones instead.
    - **Update on launch**: Enable this option so that whenever a job runs using this inventory, the inventory is refreshed first.
15. Click **Save**.
16. Click **Sync** to synchronize the source information. This triggers a synchronization job that you can view from the Jobs area. See Viewing Jobs for more information.
17. After the synchronization, return to the main inventory page and review the **Hosts** and **Groups** tab for imported host and group information.

Inventory File Formats

When running inventories from a project, Inventory sources can point to a file in various formats or a script plugin. The following sections provide information about the various file formats you can use.

yaml Inventory

Valid extensions available for a YAML inventory file include `.yaml`, `.yml`, and `.json`. The YAML inventory structure includes the following elements:

- `all`: The file must begin with the `all` group.
- `hosts`: Under the `all` group, `hosts` specifies individual host definitions that can have nested entries treated as variables.
- `vars`: You can optionally use this element with any defined group to include one or more variables.
- `children`: You can optionally use this to specify child groups that can recursively contain their own hosts, vars, and children.

The following is an example YAML inventory file:

```
---
all:
  children:
    web:
      vars:
        environment: web
        timezone: UTC
      children:
        prod:
          vars:
            env_type: production
            debug_mode: False
          hosts:
            web1:
              ansible_host: 192.0.2.254
              http_port: 80
              server_name: web1.example.com
            web2:
              ansible_host: 192.0.2.253
              http_port: 8080
              server_name: web2.example.com
        dev:
          vars:
            env_type: development
            debug_mode: True
          hosts:
            dev-web1:
              ansible_host: 192.0.2.252
              http_port: 8081
              server_name: dev-web1.example.com
            dev-web2:
              ansible_host: 192.0.2.251
              http_port: 8082
```

```
            server_name: dev-web2.example.com
    db:
      vars:
        database_version: 5.7
        backup_schedule: daily
      hosts:
        db1:
          ansible_host: 192.0.2.250
          db_username: admin
          db_password: password
```

INI Inventory

The valid extension for an INI inventory file is `.ini`. The INI inventory structure includes the following elements:

Section Headers

- Sections start with a header in square brackets `[]`, for example, `[webservers]`.
- Each section represents a group or category of hosts.
- Section headers can't be empty.
- Modifiers can be added to section headers, such as:
  - `:children` to indicate that the group contains subgroups.
  - `:vars` to indicate that the group contains variables assigned to all members.

Host Definitions

- Hosts are defined by listing their names under a section header, one per line.
- Hosts can have variables defined inline as key-value pairs separated by =.
- Ungrouped hosts must appear under the `[ungrouped]` section.

Value Interpretation

- Values passed in the INI format using the `key=value` syntax are interpreted differently depending on where they're declared within an inventory.
- When declared under a host, values are treated as host-specific variables.
- When declared under a group with the `:vars` modifier, values are treated as group-wide variables.
- When declared under the `[ungrouped]` section, values are treated as global variables.

The following is an example INI inventory file:

```
[all]
      ; List of all hosts
      host1 ansible_host=192.0.2.10
      host2 ansible_host=192.0.2.20
      host3 ansible_host=192.0.2.30

      [web]
      host1
      host2

      [web:vars]
```

```
http_port=80
server_name=example.com

[web:children]
dev
staging

[dev]
dev-host1 ansible_host=192.0.2.50
dev-host2 ansible_host=192.0.2.51

[dev:vars]
debug_mode=True
log_level=DEBUG

[staging]
stg-host1 ansible_host=192.0.2.100
stg-host2 ansible_host=192.0.2.101

[staging:vars]
debug_mode=False
log_level=INFO

[db]
host3

[db:vars]
db_username=admin
db_password=password
database_version=5.7

[ungrouped]
ungrp-host1 ansible_host=192.0.2.200
ungrp-host2 ansible_host=192.0.2.201
```

TOML Inventory

The valid extension for a TOML inventory file is `.TOML`. The TOML Inventory structure follow the specification found at https://toml.io/en/v1.0.0:

The following is an example TOML inventory file:

```
[all.vars]
environment = "all"

[web.vars]
environment = "web"
timezone = "UTC"

[web.hosts.web1]
ansible_host = "192.0.2.254"
http_port = 80
server_name = "web1.example.com"
env_type = "production"
debug_mode = false
```

```
[web.hosts.web2]
ansible_host = "192.0.2.253"
http_port = 8080
server_name = "web2.example.com"
env_type = "production"
debug_mode = false

[web.hosts.dev-web1]
ansible_host = "192.0.2.252"
http_port = 8081
server_name = "dev-web1.example.com"
env_type = "development"
debug_mode = true

[web.hosts.dev-web2]
ansible_host = "192.0.2.251"
http_port = 8082
server_name = "dev-web2.example.com"
env_type = "development"
debug_mode = true

[db.vars]
database_version = "5.7"
backup_schedule = "daily"

[db.hosts.db1]
ansible_host = "192.0.2.250"
db_username = "admin"
db_password = "password"
```

Script Plugin

The valid extension for an python script file `.py`. The following is an example an inventory script that generates inventory data based on an inventory file find in the same directory as the script:

```
#!/usr/bin/env python3
"""
Generates an Ansible inventory from a JSON payload file.
"""

import json

def load_json_payload(file_path: str) -> dict:
    """
    Loads JSON payload from a file.

    Args:
    - file_path (str): Path to the JSON payload file.

    Returns:
    - dict: Parsed JSON payload.
    """
    try:
        with open(file_path, 'r') as json_file:
```

```
            return json.load(json_file)
    except FileNotFoundError:
        print(f"File '{file_path}' not found.")
        return {}
    except json.JSONDecodeError as e:
        print(f"Failed to parse JSON: {e}")
        return {}

def create_group(inventory: dict, group_name: str, hosts: list) -> None:
    """
    Creates a new group in the inventory.

    Args:
    - inventory (dict): Ansible inventory dictionary.
    - group_name (str): Name of the group to create.
    - hosts (list): List of hosts to add to the group.
    """
    if group_name not in inventory:
        inventory[group_name] = {'hosts': []}
    inventory[group_name]['hosts'].extend(hosts)

def generate_ansible_inventory(payload: dict) -> dict:
    """
    Generates an Ansible inventory from the given JSON payload.

    Args:
    - payload (dict): Parsed JSON payload.

    Returns:
    - dict: Generated Ansible inventory.
    """
    inventory = {
        '_meta': {'hostvars': {}},
        'all': {'hosts': [], 'vars': {}}
    }

    os_groups = {}
    location_groups = {}

    for host in payload.get('hosts', []):
        host_id = host.get('id')
        hostname = host.get('hostname')
        ip_address = host.get('ip_address')
        status = host.get('status')
        os = host.get('os')
        location = host.get('location')
        owner = host.get('owner')

        if not all([host_id, hostname, ip_address, status, os, location,
owner]):
            continue  # Skip hosts with missing information

        inventory['all']['hosts'].append(ip_address)

        host_vars = {
            'ansible_host': ip_address,
```

```
                'status': status,
                'os': os,
                'location': location,
                'owner': owner
            }
            inventory['_meta']['hostvars'][ip_address] = host_vars

            create_group(os_groups, os, [ip_address])
            create_group(location_groups, location, [ip_address])

    inventory.update(os_groups)
    inventory.update(location_groups)

    return inventory

if __name__ == '__main__':
    json_payload_file = 'labs/json-payload'
    payload = load_json_payload(json_payload_file)
    inventory = generate_ansible_inventory(payload)
    print(json.dumps(inventory))
```

The following is a sample inventory file called `json_payload`:

```
{
    "hosts": [
        {
            "id": 1,
            "hostname": "host1.example.com",
            "ip_address": "192.168.1.101",
            "status": "active",
            "os": "Oracle_Linux_8",
            "location": "Ottawa",
            "owner": "User_4"
        },
        {
            "id": 2,
            "hostname": "host2.example.com",
            "ip_address": "192.168.1.102",
            "status": "inactive",
            "os": "Oracle_Linux_8",
            "location": "Texas",
            "owner": "User_1"
        },
        {
            "id": "3",
            "hostname": "host3.example.com",
            "ip_address": "192.168.1.10",
            "status": "active",
            "os": "Oracle_Linux_9",
            "location": "West_Coast",
            "owner": "admin_1"
        },
        {
            "id": "4",
            "hostname": "host4.example.com",
```

```
            "ip_address": "192.168.1.11",
            "status": "active",
            "os": "Oracle_Linux_9",
            "location": "West_Coast",
            "owner": "admin_2"
        },
        {
            "id": "5",
            "hostname": "host5.example.com",
            "ip_address": "192.168.1.15",
            "status": "inactive",
            "os": "Oracle_Linux_7",
            "location": "East_Coast",
            "owner": "user_1"
        },
        {
            "id": "6",
            "hostname": "host6.example.com",
            "ip_address": "192.168.1.16",
            "status": "inactive",
            "os": "Oracle_Linux_7",
            "location": "East_Coast",
            "owner": "user_2"
        },
        {
            "id": "7",
            "hostname": "host7.example.com",
            "ip_address": "192.168.1.16",
            "status": "inactive",
            "os": "Oracle_Linux_7",
            "location": "Ottawa",
            "owner": "user 3"
        }
    ]
}
```

OCI Inventory

Valid extensions for OCI inventory files are `.oci.yml` or `.oci.yaml`. The following is an example an OCI inventory file that generates inventory data based on an OCI tenancy and compartment. The tenancy information is provided by the associated OCI credential. The compartment OCID is specified in the inventory file:

```
---
plugin: oracle.oci.oci

regions:
 - us-ashburn-1
compartments:
 - compartment_ocid: "<sample-ocid>"
  fetch_compute_hosts: true

hostname_format_preferences:
 - "public_ip"
 - "private_ip"
```

> **Note**
>
> To run an OCI inventory, you must use a version of an Oracle Linux Automation Engine Execution Environment that includes the OCI SDK and the OCI Ansible Collection. For more information, see Oracle Linux Automation Manager 2.3: Release Notes. For more information about configuring an OCI Ansible Inventory, see https://docs.oracle.com/en-us/iaas/Content/API/SDKDocs/ansibleinventoryintro.htm.

Setting Up Credentials

Oracle Linux Automation Manager provides functionality to create and store various types of credentials needed for working with private repositories and target inventory hosts. Examples of such credentials include:

- **Git Credentials:**

  Git credentials are used to access private registries you might be using as a source of projects and playbooks, and container images for any execution environments. They can include usernames and passwords, and SCM Private keys and passphrases.

- **Machine credentials:**

  Machine credentials are used to access target hosts through ssh. For example, a machine credential could be a username and password pair, or a username and ssh private key.

- **Orace Cloud Infrastructure:**
  Oracle Cloud Infrastructure (OCI) credentials are used to access OCI machines or inventory. You must use OCI credentials using jobs running on the version of Oracle Linux Automation Engine Execution Environment that includes the OCI SDK. For more information about Oracle Linux Automation Engines Execution Environments that include the OCI SDK, see Oracle Linux Automation Manager 2.3: Release Notes.

In Oracle Linux Automation Manager the credentials you enter are encrypted before being stored to the database. The encryption/decryption algorithm uses a variation of Fernet: a symmetric encryption cipher using AES-256 in CBC mode alongside a SHA-256 HMAC. The key is derived from the SECRET_KEY, which is set in the Oracle Linux Automation Manager installation (see Oracle Linux Automation Manager 2.3: Installation Guide for more details).

The credentials are decrypted when they're extracted and used.

The following sections provide information about setting up various kinds of credentials on Oracle Linux Automation Manager.

Setting Up Git Credentials

To set up Git credentials, do the following:

1. Log into Oracle Linux Automation Manager with an administrator user account.
2. Display the left navigation menu if it's not already visible by switching the Global navigation menu button in the top-left corner of the page.
3. From the **Resources** section, click **Credentials**.

   The Credentials page appears.
4. Click the **Add** button.

   The Create New Credential page appears.

5. In the **Name** field, enter the name of the credential. For example, `GitCredential1`.
6. In the **Organization** field, click the **search** button.

   The Select Organization dialog appears.
7. From the organization list, select an organization.
8. Click **Select**.
9. From the **Credential Type** list, select **Source Control**.
10. In the **UserName** field, enter the username, if required.
11. In the **Password** field, enter a password, if required.
12. In the **SCM Private Key** field, enter the SSH private key (if you're using SSH authentication) that you set up on the host running Oracle Linux Automation Manager.
13. In the **Private Key Passphrase** field, enter a passphrase (if you have set up a passphrase for SSH authentication).
14. Click **Save**.

Setting Up an Oracle Cloud Infrastructure Credentials

To set up Oracle Cloud Infrastructure (OCI) credentials, do the following:

1. Log into Oracle Linux Automation Manager with an administrator user account.
2. Display the left navigation menu if it's not already visible by switching the Global navigation menu button in the top-left corner of the page.
3. From the **Resources** section, click **Credentials**.

   The Credentials page appears.
4. Click the **Add** button.

   The Create New Credential page appears.
5. In the **Name** field, enter the name of the credential. For example, `OCICredential`.
6. In the **Organization** field, click the **search** button.

   The Select Organization dialog appears.
7. From the organization list, select an organization.
8. Click **Select**.
9. From the **Credential Type** list, select **Oracle Cloud Infrastructure**.
10. In the **User OCID** field, enter the OCID.
11. In the **Fingerprint** field, enter a fingerprint.
12. In the **Tenant OCID** field, enter the tenant OCID.
13. In the **Region** field, enter the region.
14. In the **Private Key** field, enter the SSH private key.
15. In the **Private User Key Passphrase** field, enter a passphrase (if you have set up a passphrase for SSH authentication).
16. Click **Save**.

Setting Up Oracle Linux Virtualization Manager Credentials

To set up Oracle Linux Virtualization Manager credentials, do the following:

1. Log into Oracle Linux Automation Manager with an administrator user account.
2. Display the left navigation menu if it's not already visible by switching the Global navigation menu button in the top-left corner of the page.
3. From the **Resources** section, click **Credentials**.

   The Credentials page appears.
4. Click the **Add** button.

   The Create New Credential page appears.
5. In the **Name** field, enter the name of the credential. For example, `OLVMCredential1`.
6. In the **Organization** field, click the **search** button.

   The Select Organization dialog appears.
7. From the organization list, select an organization.
8. Click **Select**.
9. From the **Credential Type** list, select **Oracle Linux Virtualization Manager**.
10. In the **Host (Authentication URL)** field, enter the URL of the Oracle Linux Virtualization Manager host to authenticate. Use the following format:

    `https://<hostname or IP address>/ovirt-engine/api`
11. In the **UserName** field, enter the username.
12. In the **Password** field, enter a password.
13. In the **CA Path** field, enter the absolute path to the CA file, if required.
14. Click **Save**.

Setting Up Machine Credentials

Machine credentials are used to access target hosts through ssh.

To set up a machine credential, do the following:

1. Log into Oracle Linux Automation Manager with an administrator user account.
2. Display the left navigation menu if it's not already visible by switching the Global navigation menu button in the top-left corner of the page.
3. From the **Resources** section, click **Credentials**.

   The Credentials page appears.
4. Click the **Add** button.

   The Create New Credential page appears.
5. In the **Name** field, enter the name of the credential. For example, `InventoryMachineCredential1`.
6. In the **Organization** field, click the **search** button.

The Select Organization dialog appears.

7. From the organization list, select an organization.
8. Click **Select**.
9. From the **Credential Type** list, select **Machine**.
10. In the **Username** field, enter the username, if required.
11. In the **Password** field, enter a password, if required.
12. Select **Prompt on launch** checkbox above the Password field if you want a password prompt to appear when launching job templates or job workflows against an inventory.
13. In the **SSH Private Key** field, enter the SSH private key that Oracle Linux Automation Engine uses to access target hosts when using SSH authentication if required.
14. In the **Signed SSH Certificate** field, enter a signed SSH certificate, if required.
15. In the **Private Key Passphrase** field, enter a passphrase (if you have set up a passphrase for SSH authentication).
16. Select the **Prompt on launch** checkbox above the Private Key Passphrase field if you want a private key passphrase prompt to appear when launching job templates or job workflows against an inventory.
17. From the **Privilege Escalation Method** list, select an escalation method if the playbooks you run in the machine require an escalation method, for example, `sudo`.
18. In the **Privilege Escalation Username** field, enter the username, if required.
19. In the **Privilege Escalation Password** field, enter a password, if required.
20. Select **Prompt on launch** above the Privilege Escalation Password field if you want a password prompt to appear when launching job templates or job workflows against an inventory.
21. Click **Save**.

Setting Up Ansible Galaxy/Automation Hub API Token Credentials

Oracle Linux Automation Manager provides the Ansible Galaxy/Automation Hub Api Token credential type to enable you to access and download collections from a repository location. For example, you might create the following credentials:

- A credential called `My Ansible Galaxy Credential`, to accesses the publicly open location https://galaxy.ansible.com/. Such a location, being publicly open, doesn't require you to enter any authentication information, such API Token, into the credential.
- A credential called `My Private Auto Hub API Token Credential`, to access one of the Private Automation Hub repositories, for example at `https://myprivateautomationhub/published`. Such a location would require authentication, so you would enter an API Token obtained from the Private Automation Hub console associated with that registry.

You can add more than one Ansible Galaxy/Automation Hub API Token credential to an organization. The order in which you add the credentials to an organization is important as it determines the order in which repositories are searched when collections are to be downloaded for a project assigned to that organization.

Important

Ensure You Have Enabled the Download of Collections and Roles

To allow collections and roles listed in an SCM project `requirements.yml` to be dynamically downloaded, you must enable the following **Job Settings**:

- **Enable Collection(s) Download**
- **Enable Role Download**

> **Note**
>
> These settings are enabled by default when you first install Oracle Linux Automation Manager.

To edit Job settings:

1. Click **Settings** in the left-hand navigation menu.
2. Click the **Job settings** link.
3. Click **Edit** and proceed to edit the settings.
4. Click **Save**.

The following examples show you the steps to follow to create such credentials.

Setting Up an Ansible Galaxy Credential

To set up an Ansible Galaxy Credential for accessing collections, do the following:

> **Note**
>
> The `Ansible Galaxy` credential is created by default when you install Oracle Linux Automation Manager.

1. Log into Oracle Linux Automation Manager with an administrator user account.
2. Display the left navigation menu if it's not already visible by switching the Global navigation menu button in the top-left corner of the page.
3. From the **Resources** section, click **Credentials**.

   The Credentials page appears.
4. Click the **Add** button.

   The Create New Credential page appears.
5. In the **Name** field, enter the name of the credential. For example, `My Ansible Galaxy Credential`.
6. In the **Organization** field, click the **search** button.

   The Select Organization dialog appears.
7. From the organization list, select an organization.

8. Click **Select**.
9. From the **Credential Type** list, select **Ansible Galaxy/Automation Hub API Token**.
10. In the **Galaxy Server URL** field, enter the galaxy ansible URL https://galaxy.ansible.com/.
11. Click **Save**.

Setting Up a Private Automation Hub Collections Credential

Private Automation Hub credentials are used to access a Private Automation Hub instance using a URL and API token which contains collections in various repositories.

To set up a Private Automation Hub credential for accessing collections, do the following:

1. Log into Oracle Linux Automation Manager with an administrator user account.
2. Display the left navigation menu if it's not already visible by switching the Global navigation menu button in the top-left corner of the page.
3. From the **Resources** section, click **Credentials**.

   The Credentials page appears.
4. Click the **Add** button.

   The Create New Credential page appears.
5. In the **Name** field, enter the name of the credential. For example, `My Private Auto Hub API Token Credential`.
6. In the **Organization** field, click the **search** button.

   The Select Organization dialog appears.
7. From the organization list, select an organization.
8. Click **Select**.
9. From the **Credential Type** list, select **Ansible Galaxy/Automation Hub API Token**.
10. In the **Galaxy Server URL** field, enter a Private Automation Hub repository URL. For more information about different repository types, see Oracle Linux Automation Manager 2.3: Private Automation Hub User's Guide.
11. In the API Token field, enter an Private Automation Hub API token. For more information about creating an API token in Private Automation Hub, see Oracle Linux Automation Manager 2.3: Private Automation Hub User's Guide.
12. Click **Save**.

Setting Up Container Registry Credentials for an Execution Environment

To set up a Private Automation Hub credentials for accessing collections, do the following:

1. Log into Oracle Linux Automation Manager with an administrator user account.
2. Display the left navigation menu if it's not already visible by switching the Global navigation menu button in the top-left corner of the page.
3. From the **Resources** section, click **Credentials**.

   The Credentials page appears.
4. Click the **Add** button.

   The Create New Credential page appears.

5. In the **Name** field, enter the name of the credential. For example, `Registry Credentials`.
6. In the **Organization** field, click the **search** button.

   The Select Organization dialog appears.
7. From the organization list, select an organization.
8. Click **Select**.
9. From the **Credential Type** list, select **Container Registry** .
10. In the Authentication URL field, enter the IP address or host name of the container registry you want to connect with. For example, this might be a Private Automation Hub instance hosting your execution environment container images or it could be https://container-registry.oracle.com/. For more information about Private Automation Hub, see Oracle Linux Automation Manager 2.3: Private Automation Hub User's Guide.
11. In the **Username** field, enter the username of the container registry user that has access to the execution environment images you need.
12. In the **Password or Token** field, enter the password for the user.
13. Enable Verify SSL if the container registry is configured for SSL authentication.
14. Click **Save**.

Setting Up Credentials Permissions

> **Note**
>
> You can only assign credential access to users and teams that belong to the same organization that the credential belongs to.

To assign users and teams permissions to use a credential perform the following steps:

1. Log into Oracle Linux Automation Manager with an administrator user account.
2. Display the left navigation menu if it's not already visible by switching the Global navigation menu button in the top-left corner of the page.
3. From the **Resources** section, click **Credentials**.

   The Credentials page appears.
4. Select an existing credential.

   The selected credential is displayed with the Details tab active.
5. Click the **Access** tab.

   The Access tab lists users that already have access to the credential, for example users with full administrative privileges.
6. Click the **Add** button.

   The Add Roles page appears and shows options **Users** and **Teams**.
7. Click one of the options. For example, Users.

   A list of available users appears.

> **Note**
>
> Users with full administrator privileges don't appear because they have permissions for all entities in Oracle Linux Automation Manager.

8. From the left column, select the checkbox beside each user you want to provide permission to use the credential.
9. Click **Next**.

   The permissions you can assign to selected users is displayed.
10. Select the checkbox beside each permission you want to assign to the selected users.
11. Click **Save**.

The previous example assumes that users, rather than teams, are being assigned new roles to access the credential. If instead of users you want to assign access to teams, then you follow the same steps as the previous procedure except you choose teams rather than users when asked to specify which resource type you're configuring with new permissions.

Setting Up Job Templates

The following sections provide information about setting up job templates.

Setting Up a Job Template

Job Templates let you run playbooks tasks that are associated with a project against the hosts contained in an inventory. From a job template, you can also specify:

- Permissions: Which teams, users, or both can run the job.
- Notification: What notifications the job template triggers.
- Completed Jobs: A list of all completed jobs.
- Schedules: A list of schedules for when a job runs.

To set up a job template, do the following:

1. Log into Oracle Linux Automation Manager with an administrator user account.
2. Display the left navigation menu if it's not already visible by switching the Global navigation menu button in the top-left corner of the page.
3. From the **Resources** section, click **Templates**.

   The Templates page appears.
4. Click the **Add** button.

   A menu appears.
5. Click the **Add job template** option.

   The Create New Job Template page appears.
6. In the **Name** field, enter a name for the job template. For example, `JobTemplate1`.
7. From the **Job Type** list, select one of the following:
   - **Run**: Runs the playbook.
   - **Check**: Checks that the playbook can run without errors.

8. Select the **Prompt on launch** checkbox next to the **Job Type** field if you want a dialog box to confirm the choice of job type when the job is launched.

> **Note**
>
> Some fields have a **Prompt on launch** checkbox displayed next to them. Selecting such a checkbox next to a field means a dialog is displayed when the job is launched to confirm the value to be used for the field.

9. In the **Inventory** field, click the **search** button.

   The Select Inventory dialog appears.

10. From the inventory list, select the inventory containing the hosts you want this job to manage.

11. In the **Project** field, click the **search** button.

    The Select Project dialog appears.

12. Select the project containing the playbook you want this job to run.

    For example, you might select a project that contains the following sample Oracle Linux Automation Engine playbooks:

    - The sample Yum playbook that runs a yum update on the inventory.
    - The sample httpd playbook that creates an httpd server on the inventory's host.

    For more information, see Setting Up and Running Sample Playbooks.

13. In the **Execution Environment** field, click the **search** button.
    The Select Execution Environment dialog appears.

14. Select the execution environment you want this job to use.
    If you don't select an environment, the command runs on the default execution environment depending on the version of Oracle Linux Automation Manager you are running. For more information about execution environments, see Oracle Linux Automation Manager 2.3: Release Notes.

15. From the **Playbook** list, select a playbook from the selected project. For example, if you're using the sample Yum playbook, select `yum_update.yaml`.

16. In the **Credentials** field, click the **search** button.

    The Select Credentials dialog appears.

17. From the **Selected Category** list, select the one of host categories to display available credentials.

18. Select a credential you have set up to access the hosts specified in the inventory you selected, if required.

19. In the **Labels** field, enter a label that describes the job. You can use these labels in various Oracle Linux Automation Manager views to group and filter job templates and completed jobs.

20. In the **Variables** field (a code box), you can enter YAML or JSON code to pass extra command-line variables to the playbook by doing the following:

    a. Click the **YAML** or the **JSON** button to specify which syntax you want to use.

    b. Type the variables in the code edit box.

21. In the **Forks** field, use scroll arrows to select the number of forks allowed. This is the number of parallel or simultaneous processes that can run while executing an Oracle Linux Automation Engine playbook. An empty value or a value of 1 uses the default specified in `/etc/ansible/ansible.cfg`.
22. In the **Limit** field, you can enter one or more hostname, IP address, or group name, separated by comma, that further limit the hosts specified in the inventory where the job can run, if required. For example, to run a job on only hosts included in a group in the inventory that's called group1, enter `group1`. To run the job against two groups called group1 and group2, enter `group1,group2`.
23. From the **Verbosity** list, select a level of output that Oracle Linux Automation Manager displays as the job template runs. Options are as follows:
    - **0 (Normal)**
    - **1 (Verbose)**
    - **2 (More Verbose)**
    - **3 (Debug)**
    - **4 (Connection Debug)**
    - **5 (WinRM debug)**
24. In the **Job Slicing** field, use scroll arrows to select the number of job slices allowed. Job slices enable you to divide the work done by a job template into the specified number of job slices, each running the same tasks against a part of the inventory.
25. In the **Timeout** field, use scroll arrows to select the time in seconds before a job times out.
26. Enable **Show Changes** to show the changes made by an Oracle Linux Automation Engine task in a job output.
27. In the **Instance Groups** field, click the **search** button.

    The Select Instance Groups dialog appears.
28. Select one or more instance groups you created when you installed Oracle Linux Automation Manager.
29. In the **Job Tags** field, enter one or more job tags. Tags enable you to run portions of large playbook that contain tags. You can specify more than one tag using commas to separate each tag.
30. In the **Skip Tags** field, enter one or more job tags to skip. You can specify more than one tag using commas to separate each tag.
31. In the **Options** section, check the following options if required:
    - **Privilege Escalation**: The playbook runs with administrator privileges.
    - **Provisioning Callbacks**: Creates a provisioning callback URL so a host can contact Oracle Linux Automation Manager and request a configuration update using this job template.
    - **Enable Webhook**: The playbook enables webhooks.
    - **Concurrent Jobs**: Oracle Linux Automation Manager can run several instances of the job template at the same time.
    - **Enable Fact Storage**: Uses facts gathered from playbooks run against hosts and stores discovered facts in the cache. To view gathered facts, see Managing Hosts.

> **Note**
>
> Playbooks gather facts by default unless fact gathering is explicitly disabled in the playbook. If the playbook disables fact gathering, this option has no effect.

- **Prevent Instance Group Fallback**
  Enable to prevent adding any inventory or organization instance groups to the list of preferred instances groups to run the job template on. If the list is empty, the global instance groups is applied.

32. Click **Save**.

Setting Up and Running Sample Playbooks

When you install Oracle Linux Automation Manager, sample Oracle Linux Automation Engine playbooks are also included in the `/var/lib/ol-automation-manager/projects/demo_bundled` directory. This directory contains the following playbooks:

- `httpd.yaml`: This playbook installs an Apache HTTP server on the target inventory.
- `yum_update.yaml`: This playbook runs a yum update on the target inventory.

To set up and run these playbooks as the admin user, do the following:

1. Log into Oracle Linux Automation Manager as the admin user.
2. Set up an organization, for example `Sample Yum and Apache Org`. For more information, see Setting Up Organizations.
3. Set up credentials for the hosts where you want to run the playbooks as required, for example `Sample Yum and Apache Host Credential`. For more information, see Setting Up Machine Credentials.
4. Display the left navigation menu if it's not already visible by switching the Global navigation menu button in the top-left corner of the page.
5. From the **Resources** section, click **Projects**.

   The Projects page appears.
6. Click the **Add** tab.

   The Create New Project page appears.
7. In the **Name** field, enter a name, for example, `Sample Yum and Apache project`.
8. In the **Organization** field, click the **search** button.

   The Select Organization dialog appears.
9. From the Organization List, select an organization you have created, for example `Sample Yum and Apache Org`.
10. From the **Source Control Credential Type** list, select **Manual**.
11. From the Playbook Directory list, select **demo_bundled**.
12. Click **Save**.

> **Note**
>
> For more information on creating projects see Managing Projects

13. Display the left navigation menu if it's not already visible by switching the Global navigation menu button in the top-left corner of the page.
14. From the **Resources** section, click **Inventory**.

    The Inventories page appears.
15. Click the **Add** button.

    A menu appears.
16. Click the **Add Inventory** option.

    The Create new inventory page appears.
17. In the **Name** field, enter a name for the inventory. For example, `Sample Yum an Apache Inventory`.
18. In the **Organization** field, click the **search** button.

    The Select Organization dialog appears.
19. From the Organization list, select an organization you have created, for example, `Sample Yum and Apache Org`.
20. Click **Save**.

> **Note**
>
> For more information on creating inventories see Managing Inventories and Groups

21. Display the left navigation menu if it's not already visible by switching the Global navigation menu button in the top-left corner of the page.
22. From the **Resources** section, click **Hosts**.

    The Hosts page appears.
23. Click the **Add** button.

    The Create New Host page appears.
24. In the **Name** field, enter the IP address or host name of the system you're connecting to. The host must be reachable.
25. In the **Inventory** field click the **search** button.
    The Select Inventory page appears.
26. From the inventory list, select an inventory you have created, for example `Sample Yum and Apache Inventory`.
27. Click **Save**.

> **Note**
>
> Also see Managing Hosts for information on setting up hosts for an inventory.

28. Display the left navigation menu if it's not already visible by switching the Global navigation menu button in the top-left corner of the page.
29. From the Resources section, click **Templates**.

    The Templates page appears.

30. Click the **Add** button.

    A menu appears.

31. Click the **Add job template** option.

    The Create New Job Template appears.

32. In the **Name** field, enter a name for the job template. For example, `Sample Yum or Apache Job`.

33. From the Job Type list, select **Run**.

34. From the Inventory list, select an inventory, for example `Sample Yum an Apache Inventory`.

35. From the **Project** list, select a project, for example, `Sample Yum and Apache project`.

36. From the **Playbook** list, select either `yum_update.yaml` to run a yum update on the host or `httpd.yaml` to set up an Apache instance on the host.

37. From the **Credentials** lists, select a credential, for example `Sample Yum and Apache Host Credential`.

38. From the **Verbosity** list, select **0 (Normal)**.

39. If you're running the `httpd.yaml` playbook, enable **Enable Privilege Escalation** so the playbook runs with administrator privileges.

40. Click **Save**.

41. Click **Launch**.

42. The **Output** page appears so you can observe the progress of the job.

> **Note**
>
> For more information on setting up a template see Setting Up a Job Template

Setting Up Workflow Templates

The following sections provide information about setting up workflow templates.

About Workflow Templates

A workflow template lets you use a graphical tool, the **Workflow Visualizer**, to configure the run sequence of disparate components such as job templates and management jobs, as nodes in a linear graph-like design.

The components in a workflow don't have to share the same organizations and inventories.

Node Types in Workflow Templates

The nodes in a workflow can be any of the following components:

- Job Template
- Management Job
- Workflow Job Template
- Approval

- Project Syncs
- Inventory Source Sync

Workflow Design, Features, and Properties

The following list highlights some key aspects of workflow design, features, and properties:

- **Workflow Graphs Read From Left to Right**

  The Workflow Visualizer's graph-like design is read from left to right. For example, if a workflow design has `example_node_1` on the left connected by a straight line to `example_node_2` to the right, this indicates `example_node_1` runs immediately before `example_node_2`.

- **Parent Nodes and Child Nodes**

  Continuing with the previous example, if *`example_node_1`* is configured to run immediately before *`example_node_2`*, then *`example_node_1`* is said to be the parent node of *`example_node_2`*.

  A node can have more than one parent node. For example, you might have a workflow where you configure parent nodes *`RunSampleYumPlaybook`* and *`RunSampleApachePlaybook`* to run immediately before child node *`InstallWebApplication`*. In such a case the Workflow Visualizer shows the lines from each parent node converging directly on child node *`InstallWebApplication`*.

- **Run Type Condition**

  When you add a child node you must select from one of the following options to stipulate the condition under which the child node should run:

  – **`On Success`** (default):

    Select `On Success` to specify the child node you're adding should only run if the present node you're editing completes its run in a successful state. If you select this option, the connection to the added child node is shown as a **green line**.

  – **`On Failure`**:

    Select `On Failure` to specify the child node you're adding should only run if the present node you're editing completes its run in a failure state. If you select this option, the connection to the added child node is shown as a **red line**.

  – **`Always`**:

    Select `Always` to specify the child node you're adding should always run, regardless of the final state of the present node you're editing. If you select this option, the connection to the added child node is shown as a **blue line**.

- **Convergence Property**

  The Convergence property of a node determines whether it runs based on the final status of the parent nodes that converge directly to it. This option is significant if the node has more than one parent node linking directly to it. You can select one of the following values for the Convergence property:

  – **`Any`** (default): The `Any` option stipulates this node can run if any of the nodes that converge on this node complete as specified.

  – **`All`**: The `All` option implies this node can only run if all the nodes that converge on this node complete as specified.

Setting Up a Workflow Template

> **Note**
>
> For the purposes of the example, it's assumed the aim is to create a workflow of job templates that run the following sample playbooks:
>
> - `yum_update.yaml`: This playbook runs a yum update on the target inventory.
> - `httpd.yaml`: This playbook installs an Apache HTTP server on the target inventory.
>
> For more information on setting up jobs to run the sample playbooks see Setting Up and Running Sample Playbooks

To set up a workflow template, do the following:

1. Create a job template named `RunSampleYumPlaybook` that runs the playbook `yum_update.yaml` against an inventory of servers.
2. Create a job template named `RunSampleApachePlaybook` that runs the play book `httpd.yaml` against an inventory of servers.
3. Display the left navigation menu if it's not already visible by switching the Global navigation menu button in the top-left corner of the page.
4. From the **Resources** section, click **Templates**.

   The Templates page appears.
5. Click the **Add** button.

   A menu appears.
6. Click the **Add workflow template** option.

   The Create New Workflow Template page appears.
7. In the **Name** field, enter a name for the workflow template. For example, `SampleWorkflowTemplate`.

   > **Note**
   >
   > The Name field is the only mandatory field on this page. The organization and inventory fields, for example, don't have to be set at the workflow level.
8. Click **Save**.

   The **Workflow Visualizer** page is displayed in the browser with a **Start** button.
9. Click the **Start** button in the Workflow Visualizer page.
10. The **Add Node** dialog appears.
    a. The **Node Type** drop down list at the top of the Add Node dialog lets you select the type of resource to be added as a node. Accept the default choice of `Job Template`.
    b. Select `RunSampleYumPlaybook` from the list of available job templates.
    c. Accept the default option (`Any`) from the **Convergence** drop down list.

> **Note**
>
> The property is only significant of the node you're adding has more than one parent converging on it.

d. Optionally enter a **Node Alias** to the node to be displayed with a particular display name, different to the resource name, in the Workflow Visualizer.

e. Click **Save**.

The **Add Node** dialog is dismissed.

The added node appears in the Workflow Visualizer.

11. Hover the mouse over the newly added node.

A small toolbar appears within the Workflow Visualizer next to the node.

12. Click the **add ("+")** icon on the toolbar to add another node after the first node.

13. The **Add Node** dialog appears.

a. As this is the second node in the sequence the dialog initially displays the following **Run Type** conditions to select from:

- `On Success`.
- `On Failure`.
- `Always`.

Accept the option selected by default (`On Success`).

b. Click **Next**.

The **Add Node** dialog at this stage displays the same fields (**Node Type**, **Convergence**, **Node Alias**) as discussed earlier in the instructions dealing with the first node.

c. Select `RunSampleApachePlaybook` and complete the fields

d. Click **Save**.

The **Add Node** dialog is dismissed.

The added node, `RunSampleApachePlaybook`, appears in the Workflow Visualizer.

> **Note**
>
> The newly added node, `RunSampleApachePlaybook`, appears to the right of the first node (its parent) and the 2 nodes are connected by a **green line** (the green line indicates that the second node runs only after the first one completes successfully).

14. Click **Save** on the top-right of the Workflow Visualizer.

The Workflow Visualizer is dismissed and the newly created workflow template is displayed with the details tab active.

15. Click **Launch** when you're ready to run the workflow.

The **Output** page is displayed and the node currently in the running state is shown with a flashing green icon in its top left corner. The second node (`RunSampleApachePlaybook`) upon successful completion of the parent (`RunSampleYumPlaybook`).

Upon successful completion of the workflow each job displays the time it took for it to complete

Creating Schedules for Resources

Oracle Linux Automation Manager lets you create schedules for the following resources:

- Job Templates
- Workflow Templates
- Inventory Sources
- Projects

The following example shows you how to create a schedule for a resource, in this case a job template:

1. Log into Oracle Linux Automation Manager.
2. Display the left navigation menu if it's not already visible by switching the Global navigation menu button in the top-left corner of the page.
3. From the **Resources** section, click **Templates**.

   The Templates page appears displaying a list of available templates.
4. Click the name of the job template you want to create a schedule for.

   A tabbed view of the job template appears. Initially the Details tab is open.
5. Switch to the **Schedules** tab.
6. Click the **Add** button.

   The Create New Schedule dialog appears. Type a descriptive name in the Name field.

   Take note of the value in the **Local time zone field** when completing the **start date** calendar and **start time** drop down list controls.

   > **Note**
   >
   > **Best Practice: Set Schedules in UTC time.**
   >
   > The application saves schedules in UTC, a world standard that isn't adjusted for daylight saving time. Therefore, to avoid any confusion involving seasonal clock changes, it's best to set and track schedules in UTC.

7. Click **Save**.

   The schedule is saved and displayed on the Schedule Details page.

   It also appears in the Schedules view.

Managing Execution Environments

The following sections provide an overview on execution environments and how to create and view the execution environments for a system.

About Execution Environments

Execution environments are portable container images in which Oracle Linux Automation Engine automation tasks run. Execution environments replace the Python virtual environments used in prior versions of the Oracle Linux Automation Engine product.

You can also use custom execution environments created with the Builder utility. These custom execution environments include the base components listed above with additional components added. For more information about the Builder utility, see Oracle Linux Automation Manager 2.3: Private Automation Hub User's Guide.

> **Note**
>
> While the Builder utility lets you create custom execution environment, in the case of an issue with the custom execution environment, Oracle might ask you to revert to the provided OLAM-EE default image to troubleshoot the problem.

As portable self-contained container images, execution environments let automation tasks run consistently across several platforms.

The Oracle Linux Automation Manager comes with the following default execution environments:

- `OLAM EE`
- `Control Plane Execution Environment`

Viewing the Execution Environment Pages

To view an execution environment, do the following:

1. Log into Oracle Linux Automation Manager.
2. Display the left navigation menu if it's not already visible by switching the Global navigation menu button in the top-left corner of the page.
3. From the **Administration** section, click **Execution Environments**.

   The **Execution Environments** page displays a summary table relating to the execution environments available in the system. The table columns include those described in the following list:

   - **Name**

     The Name column contains the name of the execution environment.

   - **Image**

     The Image column contains the full image location, including the container registry, image name, and version tag, of the image used for the execution environment. For example, *container-registry.oracle.com/oracle_linux_automation_manager/olam-ee:latest* . Alternatively, the image location might be a Private Automation Hub instance hosting the olam-ee image or a custom image. For more information see, Oracle Linux Automation Manager 2.3: Private Automation Hub User's Guide.

   - **Organization**

     The Organization column contains information about which organizations in your system can access the execution environment.

4. Click the name of an execution environment for more details.

   The execution environment is displayed in a tabbed view with the **Details** tab active. In addition to the Name, Image and Organization fields, the Details tab contains some other fields, including the following:

   - **Created**

     The Created field contains the date and time the execution environment was created.

   - **Last Modified**

     The Last Modified column contains the date and time the execution environment was last modified.

   - **Pull**

     The Pull field contains a value that stipulates the conditions under which Oracle Linux Automation Manager should pull the container image from its repository location before running a resource job. The following values are possible:

     – `Always`

       This option stipulates that a container image is always pulled before running the container.

     – `Missing`

       This option stipulates that a container image is only pulled if the image is not present before running the container.

     – `Never`

       This option stipulates that a container image never pulled before running the container.

     > **Important**
     >
     > Keep the container images updated to ensure the images have the latest CVE fixes.

5. Click the **Back to execution environments** button (the first button in the tab header row) to get back to the **Execution Environments** page.

Creating Execution Environments

To create an execution environment, do the following:

1. Log into Oracle Linux Automation Manager.
2. Display the left navigation menu if it's not already visible by switching the Global navigation menu button in the top-left corner of the page.
3. From the **Administration** section, click **Execution Environments**.
4. Click **Add**.
   The Create new execution environment page appears.
5. In the **Name** field, enter the name of the execution environment.
6. In the **Image** field, enter the image location, including the container registry, image name, and version tag, of the image used for the execution environment. For example, *container-registry.oracle.com/oracle_linux_automation_manager/olam-ee:latest* . Alternatively, you can point to a Private Automation Hub instance hosting the olam-ee image or a custom

image. For more information see, Oracle Linux Automation Manager 2.3: Private Automation Hub User's Guide.

7. From the **Pull** list, select one of the following values that stipulates the conditions under which Oracle Linux Automation Manager should pull the container image from its repository location before running a resource job:

   - **`Always`**

     This option stipulates that a container image is always pulled before running the container.

   - **`Missing`**

     This option stipulates that a container image is only pulled if the image isn't present before running the container.

   - **`Never`**

     This option stipulates that a container image never pulled before running the container.

   > **Important**
   >
   > Keep the container images updated to ensure the images have the latest CVE fixes.

8. **Organization**

   The Organization field contains information about which organizations in the system can access the execution environment.

9. **Registry credential**

   The Registry credential field contains information about which credential the system must use to connect to a container registry. For more information, see Setting Up Container Registry Credentials for an Execution Environment.

10. Click **Save**.

Managing Instance Groups

This chapter looks at how you create and manage instance groups with Oracle Linux Automation Manager.

About Instance Groups and Service Mesh Node Types

The service mesh provided by Oracle Linux Automation Manager is an overlay network that provides a mesh of nodes of different types, such as controller nodes and execution nodes, to let you manage jobs across different inventories.

Oracle Linux Automation Manager provides instance groups to enable you to organize the different hosts in its service mesh into separate groups, depending upon their node type and purpose. For example, to create an instance group called `mesh_nodes_for_production` with nodes to run jobs on production machines, and another group called `mesh_nodes_for_development_machines` to run jobs on development servers.

The following list outlines key points about instance groups and nodes:

- **One Job Queue per Instance Group**

Each instance group has its own job queue, and any node in the instance can process a job from that queue.

- **Configuration of Extra Nodes**

  The nodes initially available for association with Instance Groups are those you configured for the mesh topology during the installation process. If you require extra nodes, for example for a new instance group you have created, then you need to follow the same process you used to create the initial nodes (see Oracle Linux Automation Manager 2.3: Installation Guide for more information on configuring nodes for a mesh topology).

- **Nodes Can Belong to More Than One Instance Group**

  For example, you can make some nodes available across the cluster, whilst reserving others for a specific instance group.

- **Associating Instance Groups With Resources**

  By default, jobs are run on nodes in the **`Execution`** instance group you created during the installation process (see Oracle Linux Automation Manager 2.3: Installation Guide).

  However, if required, you can specify instance groups by associating them with one of the following resources:

  - **Job Template**
  - **Inventory**
  - **Organization**

  When a job runs, Oracle Linux Automation Manager first checks the job template associated with the job to see which instance group the job should be assigned to. If no instance group is associated with the job template, Oracle Linux Automation Manager next checks the inventory resource linked to the job. If no instance group is associated with the inventory resource, the Organization resource is checked.

- **Instance Groups Provided by Default**

  Oracle Linux Automation Manager comes with the following instance groups:

  - **`controlplane`**:

    The `controlplane` instance group contains control nodes. Control nodes run persistent Oracle Linux Automation Manager services, such as the task dispatcher, project and inventory updates, and system jobs, but not regular jobs. Control nodes don't have execution capability.

  - **`execution`**:

    The `execution` instance group contains execution nodes. Execution nodes run jobs under `ansible-runner` with `podman` isolation. Execution nodes only run user-space jobs.

The following sections show you how to view and edit instance groups, how to associate jobs and resources with specific instance groups, and how to create groups.

Viewing the Instance Groups Summary

To view the Instance Groups page, do the following:

1. Log into Oracle Linux Automation Manager with an administrator user account.
2. Display the left navigation menu if it's not already visible by switching the Global navigation menu button in the top-left corner of the page.
3. From the **Administration** section, click **Instance Groups**.

The Instance Groups page appears.

The **Instance Groups** page displays a summary table relating to the instance groups in the system. The table columns include those described in the following list:

- **Name**

  The Name column contains the name of the Instance Group. Clicking the name takes you to a tabbed view of that instance's properties. The **Details** tab is the active tab initially.

- **Running Jobs**

  The Running Jobs column contains the number of jobs associated with the instance group that are running.

- **Total Jobs**

  The Total Jobs column contains the sum of the number of jobs associated with the instance group that have completed, and the number of jobs associated with the group that are still running.

- **Capacity**

  The Capacity column displays a **Used capacity** progress bar that indicates the percentage of capacity of the instance group that's being used.

- **Action**

  The Action column displays an **Edit instance group** icon that lets you edit the instance.

Editing an Instance Group

The following sections show you how to edit an instance group and change properties on the nodes associated with the group.

Editing Instance Group Policy Properties

You can set the following Policy properties on an instance group:

- **Policy instance minimum**

  This is the minimum number of instances to be automatically assigned to the group when new provisioned instances come online.

- **Policy instance percentage**

  Minimum percentage of all instances to be automatically assigned to this group when new provisioned instances come online.

To set the policy properties of an instance group, do the following:

1. Log into Oracle Linux Automation Manager with an administrator user account and navigate to the **Instance Groups** page.
2. In the summary table displayed on the Instance Groups page, go to the row with the group whose policy instance properties you want to edit.
3. Click the **Edit instance group** icon in the **Action** column.

   The **Edit details** page appears.
4. Enter values in the properties **Policy instance minimum** and **Policy instance percentage** fields.
5. Click **Save**.

   The tabbed view of the instance appears.

Associating and Disassociating Nodes with a Group Instance

The following steps show you how to associate and disassociate nodes with a group instance:

1. Log into Oracle Linux Automation Manager with an administrator user account and navigate to the **Instance Groups** page.
2. In the summary table displayed on the Instance Groups page, go to the row with the group whose node association you're looking to change.
3. Click the group instance name in the **Name** column.

   A tabbed view of that instance's properties appears. Initially the **Details** tab is open,
4. Click the **Instances** tab.

   The node instances associated with the group are displayed in a table.
5. To **associate** more node instances with the group, do the following:
   a. Click the **Associate** button.

      The Select Instances dialog appears displaying available node instances.
   b. Select the check boxes next to the nodes you want to associate with the group instance.
   c. Click **Save**.

      The Instances page will be displayed and the nodes you selected appear in the list on the page.
6. To **disassociate** nodes from the current group instance, do the following:
   a. Select the checkbox next to each host you want to disassociate from the current group and click the **Disassociate** button.
   A dialog asks you to confirm the actions.
   b. Confirm the actions if you're happy to proceed by clicking **Disassociate** button on the dialog.

      The dialog is dismissed and the nodes you disassociated from the group no longer appear in the list.

Changing Node Instance Properties

The following procedures show you how to change the properties on a specific node.

A node instance's **capacity** determines how many forks it can run simultaneously. Capacity thus impacts the number of systems a node can run jobs for in parallel. To change a node instance's **capacity**, do the following:

1. Log into Oracle Linux Automation Manager with an administrator user account and navigate to the **Instance Groups** page.
2. In the summary table displayed on the Instance Groups page, go to the row with the group associated with the node you want to edit.
3. Click the group instance name in the **Name** column.

   A tabbed view of that instance's properties appears. Initially the **Details** tab is open,
4. Click the **Instances** tab.

   The node instances associated with the group are displayed in a table.
5. Go to the row with the node instance you want to change.

6. To adjust the **Capacity**, drag the slider control in the **Capacity Adjustment** column to the value of the choice.

> **Note**
>
> - The change is saved automatically. No Save button is available.
> - Consider whether the new capacity setting is suitable for other instance groups the node might belong to.

The **enable/disable** setting lets you control whether a node instance is available for jobs to be assigned to it. For example, you might need to temporarily disable one of the node instances for maintenance. To change a node instance's **enable/disable** setting, do the following:

1. Log into Oracle Linux Automation Manager with an administrator user account and navigate to the **Instance Groups** page.
2. In the summary table displayed on the Instance Groups page, go to the row with the group associated with the node you want to edit.
3. Click the group instance name in the **Name** column.

   A tabbed view of that instance's properties appears. Initially the **Details** tab is open,
4. Click the **Instances** tab.

   The node instances associated with the group are displayed in a table.
5. Go to the row with the node instance you want to change.
6. To change a node instance's **enable/disable** setting, toggle the enable-disable toggle switch in the **Action** column.

> **Note**
>
> - The change is saved automatically. No Save button is available.
> - Consider whether the new setting is suitable for other instance groups the node might belong to.

Creating an Instance Group

The following steps show you how to create an instance group:

1. Log into Oracle Linux Automation Manager with an administrator user account and navigate to the **Instance Groups** page.
2. Click the **Add** button.

   A menu appears.
3. Click the **Add instance group** option.

   The Create new instance group appears.
4. In the **Name** field, enter a name for the instance group. For example, `NodesForDataCentre1`.
5. Optionally, enter integers into the **Policy instance minimum** and **Policy instance percentage** field.

> **Note**
>
> The policy instance fields are defined as follows:
>
> - **Policy instance minimum:**
>
> This is the minimum number of instances to be automatically assigned to the group when new provisioned instances come online.
> - **Policy instance percentage:**
>
> Minimum percentage of all instances to be automatically assigned to this group when new provisioned instances come online

6. Click **Save**.

   The newly created instance group is displayed in a tabbed view with the Details tab active.
7. Click the **Instances** tab to switch to the Instances tab.

   The node instances associated with the group are displayed in a table.
8. To associate more node instances with the group, click the **Associate** button.

   The Select Instances dialog appears displaying available node instances.
9. Select the check boxes next to the nodes you want to associate with the group instance.

   Click **Save**.

   The Instances page is displayed and the nodes you selected appear in the list on the page.

4

Using Views

This chapter describes how to use Oracle Linux Automation Manager views.

Using the Dashboard

The Dashboard view provides a summary of the latest state of your jobs, inventories, hosts, and projects.

To view the **Dashboard**, do the following:

1. Log into Oracle Linux Automation Manager with an administrator user account.
2. Display the left navigation menu if it's not already visible by switching the Global navigation menu button in the top-left corner of the page.
3. From the **Views** section, click **Dashboard**.

The Dashboard page consists of the following groups of controls:

Statistic Buttons
At the top of the page a row of buttons provide access to high level summary statistics for hosts, inventories, and projects.

Job activity tabs
The tabs available are

1. Job status
2. Recent Jobs
3. Recent Templates

When the dashboard page loads the Job status tab is selected by default and displays a graph of historical data for various job activities.

The following sections describe the groups of controls in more detail.

Statistic Buttons

The statistic buttons consolidate the following statistics in one display area:

Hosts
The Hosts button shows the total number of hosts included in all inventories to which the user has access.
Clicking the Hosts button takes you to the Hosts page.

Failed Hosts
The Failed Hosts button shows the number of hosts, to which the user has access, where the last job failed.
Clicking the Failed Hosts button takes you to a view that lists these hosts.

Inventories
The Inventories button shows the total number of inventories to which the user has access.

Clicking the Inventories button takes you to the inventories page.

Inventory Sync Failures
The Inventory Sync Failures button shows the number of inventories, to which the user has access, where synchronization with source inventory lists have failed.
Clicking the Inventory Sync Failures button takes you to a view that lists these inventories.

Projects
The Projects button shows the total number of projects to which the user has access.
Clicking the Projects button takes you to the Projects page.

Project Sync Failure
The Project Sync Failure button shows the number of projects, to which the user has access, where synchronization with the source Oracle Linux Automation Engine playbook have failed.
Clicking the Project Sync Failure button takes you to a view that lists these projects.

Jobs Activity Tabs

The following list describes the job activity tabs.

Job status
This tab consists of a configurable graph that displays historical data for various job activities. You can select from the following filter options to configure the graph to view the data you're most interested in:

1. **Time Period**

   The time period filter options are `Past month`, `Past two weeks`, `Past week`, and `Past 24 hours`.

2. **Job Type**

   The time period filter options are `All job types`, `Inventory sync`, `SCM update`, and `Playbook run`.

3. **Status**

   The Status filter options are `All jobs`, `Successful jobs`, and `Failed jobs`.

Recent Jobs
The Recent Jobs tab shows a table of the most recently run jobs. The table includes job status, and start and finish times for each job.
Each job that can be relaunched also has a **Relaunch Job** button in its row.

Recent Templates
The Recent Templates tab shows a table of the most recently used templates.
The templates that can be launched have **Launch Template** button in their rows.

Viewing Jobs

To view the **Jobs** page, do the following:

1. Log into Oracle Linux Automation Manager with an administrator user account.
2. From the **Views** section, click **Jobs**.

The Jobs view table that lists all activities that have been run in Oracle Linux Automation Manager including job template runs, workflow runs, project synchronizations, inventory synchronizations, one-off commands on hosts, and so on.

The table columns are described in the following list:

Column 1
The first column contains a toggle button to expand or collapse the row.
Expanding the row reveals more fields relating to the job, for example **Launched By**, **Project** and **Inventory**.

Column 2 (Unnamed column)
The second column contains a checkbox that lets you select the row.

Name
The Name column contains the name of the job and job ID number.
You can sort by this column by clicking the column header.

Status
The Status column contains the status of the job, for example `Running`, `Successful`, `Failed`.
You can sort by this column by clicking the column header.
If the job is in `Running` status you have the option of canceling it by selecting the checkbox in the leftmost column and clicking the **Cancel jobs** button.

Type
The Type column contains the type of job, for example `Management Job`, `Command`, `Playbook Run`.

Start Time
The Start Time column contains the Start time of job.
You can sort by this column by clicking the column header.

Finish Time
The Finish Time column contains the Finish time of job.
You can sort by this column by clicking the column header.

Actions
This column contains a **Relaunch Job** button if the job can be relaunched.

Viewing Scheduled Activities

To view the **Schedules** page, do the following:

1. Log into Oracle Linux Automation Manager with an administrator user account.
2. Display the left navigation menu if it's not already visible by switching the Global navigation menu button in the top-left corner of the page.
3. From the **Views** section, click **Schedules**.

The Schedules view displays a table that shows scheduled management jobs, job template runs, workflow template runs, and project SCM updates from the Schedules view.

The table columns are described in the following list:

Column 1 (Unnamed column)
The first column contains a checkbox that lets you select the row.

Name
The Name column contains the name the name of the schedule.
You can sort by this column by clicking the column header.

Type
The Type column contains the type of the schedule, for example `Management Job`.

Next Run
The Next Run column contains the date and time of the next scheduled run.
You can sort by this column by clicking the column header.

Actions
The Action column contains the following buttons:

- **Edit Schedule** button that lets you click and edit the schedule.
- **Schedule is active** on/off toggle switch that lets you switch the active setting of the schedule.

Viewing Activity Streams

The Activity Stream page displays a table that lists changes that have taken place on all objects in Oracle Linux Automation Manager. For example, the table displays edits and updates on jobs, inventories, credentials, users, schedules, and so on.

To view the **Activity Stream** page, do the following:

1. Log into Oracle Linux Automation Manager with an administrator user account.
2. Display the left navigation menu if it's not already visible by switching the Global navigation menu button in the top-left corner of the page.
3. From the **Views** section, click **Activity Stream**.

> **Note**
>
> **You can also use the View activity stream Button on Pages**
> In addition to the Activity Stream link in the left navigation menu, you can also click the **View activity stream** button in the top-right corner of most pages.
>
> One advantage of the buttons on the pages is that they append a filter parameter to the Activity Stream URL so the view initially shows only the data that relates to the object you're viewing. For example, the View activity stream button on the Jobs page points to `https://olam.example.com/#/activity_stream?type=job`.

The columns of the table displaying the changes are described in the following list:

Time
The time column contains the date and time of the change.
You can sort by this column by clicking the column header.

Initiated by
The Initiated by column gives the username of the account that initiated the change.
You can sort by this column by clicking the column header.

Event
The Event column describes the change.
The following are example entries from a test system:

- `updated job_template Sample Yum`
- `associated MyTestOrg execute_role to NewUser1`

Actions
The Actions column contains a **View event details** button. Clicking the button displays an event log for the change.

Filters:

You can set a filter on the Activity Stream view to limit the events to a particular object. The following filters are available from the Options menu drop down list in the top-right of the page.

The following filters options are available:

- Views
- Dashboard (all activity)
- Jobs
- Schedules
- Workflow Approvals
- Resources
- Templates
- Credentials
- Projects
- Inventories
- Hosts
- Access
- Organizations
- Users
- Teams
- Administration
- Credential Types
- Notification Templates
- Instances
- Instance Groups
- Applications & Tokens
- Execution Environments
- Settings