Using Automation Dashboard

This guide describes how to install and configure Automation Dashboard to evaluate automation usage across your environments and the savings associated with it.

Providing feedback on Red Hat documentation

If you have a suggestion to improve this documentation, or find an error, you can contact technical support at https://access.redhat.com to open a request.

1. View key usage metrics with Automation Dashboard

The Automation Dashboard utility is a web-based container application that provides key metrics related to job execution, efficiency, and the value derived from automation. Automation Dashboard uses automation metrics to provide automation usage data from Ansible Automation Platform. You can use this data to determine the cost of performing tasks manually versus the cost of performing tasks through automation. This comparison is used to demonstrate the savings achievable through automation.

Automation Dashboard is designed to help you:

- Get a clear overview of the automation occurring in your environment.
- Track metrics, like time saved and errors reduced, to quantify the benefits of automation.
- Analyze job execution times and failure rates to pinpoint areas where automation can be improved.
- Use the data generated by Automation Dashboard to make informed decisions about automation strategy, resource allocation, and prioritization of automation projects.

By effectively utilizing Automation Dashboard, you can gain valuable insights into your Ansible Automation Platform usage and drive continuous improvement in your automation practices.

1.1. Installing Automation Dashboard

Prerequisites

- One of the following tested configurations:
  - RHEL 9 x86 or ARM based physical or virtual host.
  - With an external database: Postgres v15 database.

IMPORTANT	Do not attempt to install Automation Dashboard on the same host(s) as Ansible Automation Platform.

- Automation Dashboard installation has been tested with the following configuration:
  - 80 GB Harddrive (depending on data growth)
  - 4 vCPUs x 16 GB Ram
  - Disk IOPS - 3000
  - Handle up to 10,000 jobs/month and 47M summaries/month
- Access to *baseos* and *Ansible Automation Platformstream* repo packages for the RHEL 9 host.
- A non-root login account to the RHEL 9 host for installation. This requires passwordless sudo access to root as well. By default, we use the $HOMEDIR of the user account.
- URL details for access to your Ansible Automation Platform instances.
- An Ansible Automation Platform 0Auth2 token, which is used for communication between the Ansible Automation Platform instances and Automation Dashboard.
- Access to download the installation bundle providing installation components for the Automation Dashboard.
- Open firewall access to allow for bi-directional communication between AAP instances and the Automation Dashboard.
  - This includes HTTPS/443 (or your Ansible Automation Platform configured port) from the dashboard to the Ansible Automation Platform instance(s).
  - Port 8447 is the default ingress port for the Automation Dashboard. This port can be configured during installation.
  - RHEL firewall ports that may block 5432 to PostgreSQL.
- A supported version of `ansible-core` installed on supported RHEL versions.

Procedure

1. Access the link: **.tar.gz** installation source.
2. Copy the installation source file to your RHEL 9 host.
3. Untar the installation source. This will require ~500Mb. of disk space. Throughout this example we will use the ec2-user home directory, /home/<username>.

```
tar -xzvf ansible-automation-dashboard-containerized-setup-bundle.tar.gz
cd ansible-automation-dashboard-containerized-setup/
```

4. Verify that the necessary software is installed by running the following commands:

```
cd ansible-automation-dashboard-containerized-setup
sudo dnf install ansible-core
ansible-galaxy collection install -r requirements.yml
```

5. Create an application `client_id/client_secret` in your AAP instance:
    a. Create an OAuth2 application using the following steps :
        i. **For Ansible 2.4**:
            - Navigate to https://AAP_GATEWAY_FQDN:/#/applications
        ii. **For Ansible 2.5 and 2.6**:
            - Navigate to https://AAP_Controller_FQDN:/access/applications
        iii. Add the following information:
            - **Name**: automation-dashboard-sso
            - **Authorization grant type**: authorization-code
            - **Organization**: Default
            - **Redirect URIs**: https://AUTOMATION_DASHBOARD_FQDN/auth-callback
            - **Client type**: Confidential

> **NOTE** The values for **Name**, **Organization**, and HTTPS port number for Ansible Automation Platform are configurable. The examples provided in this document assume use of port 443.

    b. Save the `client_id` and `client_secret information` inputs into the inventory file.
    c. Next, create an Ansible Automation Platform access token:
        i. Navigate to https://AAP_GATEWAY_FQDN/#/users/<id>/tokens, and create a token using the following information:
6. OAuth application: automation-dashboard-sso
7. Scope: read
8. Store this access token value. The access token is used in `clusters.yaml`.
9. Copy the example inventory and modify it before running the installer.

```
cp -i inventory.example inventory
vi inventory
```

> **NOTE**
> - This is an example tested inventory containing default values for Ansible Automation Platform 2.4 and 2.5.
> - You must change the following values to use this inventory configuration in your environment:
>   - Change the RHEL 9 host occurrences from `host.example.com` to your FQDN host
>   - Change the phrase `TODO` to match your passwords within all `_admin_password` or `_pg_password` values.

```
# This is our Automation Dashboard front-end application
[automationdashboard]
host.example.com ansible_connection=local

# These are required vars for the installation and should not be removed
[automationdashboard:vars]
# Configure AAP OAuth2 authentication.
# aap_auth_provider_name - name as shown on login page.
aap_auth_provider_name=Ansible Automation Platform
# aap_auth_provider_protocol - http or https
aap_auth_provider_protocol=https
# AAP version - 2.4, 2.5 or 2.6
aap_auth_provider_aap_version=2.5
# aap_auth_provider_host - AAP IP or DNS name, with optional port
aap_auth_provider_host=my-aap.example.com
# aap_auth_provider_check_ssl - enforce TLS check or not.
aap_auth_provider_check_ssl=true
# aap_auth_provider_client_id and aap_auth_provider_client_secret -
# they are obtained from AAP when OAuth2 application is created in AAP.
aap_auth_provider_client_id=TODO
aap_auth_provider_client_secret=TODO


# Specify amount of old data to synchronoize after installation.
# The initial_sync_days=N requests N days of old data, counting from "today".
# The initial_sync_since requests data from the specified data until "today".
# If both are specified, the initial_sync_since will be used.
initial_sync_days=1
# initial_sync_since=2025-08-08

# Hide warnings when insecure https request are made.
# Use this if your AAP uses self-signed TLS certificate.
# show_urllib3_insecure_request_warning=False

# Force clean install-like
# dashboard_update_secret=true

# HTTP/HTTPS settings
# nginx_disable_https=true
# Change nginx_http_port or nginx_https_port if you want to access dashboard on a
different TCP port.
# nginx_http_port=8083
# nginx_https_port=8447
# TLS certificate configuration
# The dashboard_tls_cert needs:
#   - contain server certificate, intermediate CA certificates and root CA
certificate.
#   - the server certificate must be the first one in the file.
# dashboard_tls_cert=/path/to/tls/dashboard.crt
# dashboard_tls_key=/path/to/tls/dashboard.key
```

```
# Enable Django DEBUG.
# django_debug=True

[database]
host.example.com ansible_connection=local

[all:vars]
postgresql_admin_username=postgres
postgresql_admin_password=TODO

# AAP Dashboard - mandatory
# -------------------------
dashboard_pg_containerized=True
dashboard_admin_password=TODO
dashboard_pg_host=host.example.com
dashboard_pg_username=aapdashboard
dashboard_pg_password=TODO
dashboard_pg_database=aapdashboard
#
bundle_install=true
# <full path to the bundle directory>
bundle_dir='{{ lookup("ansible.builtin.env", "PWD") }}/bundle'
```

10. Run the installer.

```
ansible-playbook -i inventory
collections/ansible_collections/ansible/containerized_installer/playbooks/reporter_
install.yml
```

Verification

For reference, see the following example output:

```
PLAY RECAP
*********************************************************************************
*****************************************************
ec2-54-147-26-173.compute-1.amazonaws.com : ok=126  changed=51   unreachable=0
failed=0    skipped=42   rescued=0    ignored=0
localhost                  : ok=12   changed=0    unreachable=0    failed=0
skipped=9    rescued=0    ignored=0
```

Alternative configurations are possible (for example, the database for Automation Dashboard can be set on a different host). This requires additional changes to variables in the inventory file. Consult the Inventory variables section of this document for available variables.

1.2. Integrating Automation Dashboard with your Ansible Automation Platform

Integrate your Ansible Automation Platform instances into the Automation Dashboard configuration in order to collect and visualise data and gain insights into your automation.

Procedure

1. Verify that Automation Dashboard is running on https port 8447 on your RHEL host. This verification will require your login details as defined in the inventory file.

> **NOTE** Port 8447 is enabled by default, but this is configurable.

2. Enter `admin` as the username and use `dashboard_admin_password` as the password.
3. Your Ansible Automation Platform instances are added into `clusters.yml` using the following information:
   - Your Ansible Automation Platform URLs/ports for front-end access
   - A preconfigured Ansible Automation Platform 0Auth token for *read* access

> **NOTE** If you don't have access to Ansible Automation Platform, consult your administrator.

4. Configure a personal access token. For more information, see Configuring access to external applications with token-based authentication.
5. Once you have configured your access token, run the following command:

```
cp clusters.example.yaml clusters.yaml
vi clusters.yaml
```

6. You can add one or more Ansible Automation Platform instances (of the *same AAP version*) into the Automation Dashboard configuration for pulling and combining data by using the following:

> **NOTE** If you only have one Ansible Automation Platform instance, then remove the second entry.

```
---
clusters:
  - protocol: https         <--- Normally https
    address: my-aap.example.com  <--- Can use IP or FQDN without http(s)://
    port: 443               <--- Normally 443
    access_token: sampleToken   <--- Your preconfigured Ansible Automation Platform
read access token
    verify_ssl: false       <--- Can be used when using self signed certs
    sync_schedules:
```

```
      - name: Every 5 minutes sync
        rrule: DTSTART;TZID=Europe/Ljubljana:20250630T070000
FREQ=MINUTELY;INTERVAL=5
        enabled: true

  - protocol: https
    address: aap2.example.com
    port: 443
    access_token: WRn2swiqg5spEwUndDkrJoCeg4Qwuw
    verify_ssl: true
    sync_schedules:
      - name: Every 5 minutes sync
        rrule: DTSTART;TZID=Europe/Ljubljana:20250630T070000
FREQ=MINUTELY;INTERVAL=5
        enabled: true
```

7. Run the following commands to load and activate your Automation Dashboard configuration:

```
podman cp clusters.yaml automation-dashboard-web:/
podman exec automation-dashboard-web /venv/bin/python ./manage.py setclusters
/clusters.yaml
```

For reference, see the following example output:

```
podman exec automation-dashboard-web /venv/bin/python ./manage.py setclusters
/clusters.yaml
Check if table exists.
Reading YML file.
Adding cluster: address=my-aap.example.com

INFO 2025-05-20 09:55:00,926 connector 187 140208297051968 Checking if is AAP 2.4
at https://my-aap.example.com:443
INFO 2025-05-20 09:55:00,926 connector 187 140208297051968 Pinging api https://my-
aap.example.com:443/api/v2/ping/
INFO 2025-05-20 09:55:00,926 connector 187 140208297051968 Executing GET request to
https://my-aap.example.com:443/api/v2/ping/

ERROR 2025-05-20 09:55:00,032 connector 301 140025281152832 GET request failed with
status 404
Successfully set up AAP clusters
```

NOTE	Automation Dashboard checks for Ansible Automation Platform 2.4, 2.5, and 2.6 instances. As shown in the example output, this can result in 404 errors. For more information on errors, see the Verification section of this chapter.

8. Use the following command to test your Automation Dashboard configuration by manually fetching data:

```
podman exec automation-dashboard-web /venv/bin/python ./manage.py syncdata
--since=2025-04-01 --until=2025-06-01
Successfully created Sync task for Cluster https://my-aap.example.com:443.
```

NOTE Consider using a short date interval to reduce test time. The format is YYYY-MM-DD.

You can then use `journalctl` to monitor progress:

```
sudo journalctl -fn10
```

9. Refresh your browser to view retrieved data within your Automation Dashboard.

Verification

If you encounter error messages during installation, consult the following table:

Issue	Possible Cause	Solution
401 errors	This is an unauthorized access message indicating authentication errors such as incorrect credentials or tokens	Verify that your access token is correct in `clusters.yml`
404 errors	This is a “not found” message indicating that something isn’t configured correctly or pointing to the correct endpoint	Verify that your Ansible Automation Platform instance URLs used in `clusters.yml` are correct

A successful installation should be running the following three container services:

```
podman ps --all --format "{{.Names}}"

postgresql
automation-dashboard-task
automation-dashboard-web
```

You can check your container logs by running the following:

```
journalctl CONTAINER_NAME=container (where container equals one of postgresql
automation-dashboard-task or automation-dashboard-web)

For example:
journalctl CONTAINER_NAME=automation-dashboard-task
May 22 13:02:07 automation-dashboard automation-dashboard-task[1607]: [wait-for-
migrations-dashboard.sh] Waiting for database migrations...
May 22 13:02:07 automation-dashboard automation-dashboard-task[1607]: [wait-for-
```

```
migrations-dashboard.sh] Attempt 1
May 22 13:02:10 automation-dashboard automation-dashboard-task[1607]: INFO 2025-05-22
13:02:10,636 periodic 2 140568371550016 Starting sync task.
May 22 13:02:10 automation-dashboard automation-dashboard-task[1607]: INFO 2025-05-22
13:02:10,636 periodic 2 140568371550016 Retrieving clusters inform>
May 22 13:02:10 automation-dashboard automation-dashboard-task[1607]: INFO 2025-05-22
13:02:10,747 periodic 2 140568371550016 Retrieved 1 clusters.
May 22 13:02:10 automation-dashboard automation-dashboard-task[1607]: INFO 2025-05-22
13:02:10,761 periodic 2 140568371550016 Retrieving data from clust>
May 22 13:02:10 automation-dashboard automation-dashboard-task[1607]: INFO 2025-05-22
13:02:10,761 connector 2 140568371550016 Checking Ansible Automation Platform version
at h>
May 22 13:02:10 automation-dashboard automation-dashboard-task[1607]: INFO 2025-05-22
13:02:10,761 connector 2 140568371550016 Checking if is Ansible Automation Platform
2.5 at>
May 22 13:02:10 automation-dashboard automation-dashboard-task[1607]: INFO 2025-05-22
13:02:10,761 connector 2 140568371550016 Pinging api https://ec2-3>
May 22 13:02:10 automation-dashboard automation-dashboard-task[1607]: INFO 2025-05-22
13:02:10,761 connector 2 140568371550016 Executing GET request to >
May 22 13:02:13 automation-dashboard automation-dashboard-task[1607]: ERROR 2025-05-22
13:02:13,820 connector 2 140568371550016 GET request failed with >
May 22 13:02:13 automation-dashboard automation-dashboard-task[1607]: INFO 2025-05-22
13:02:13,821 connector 2 140568371550016 Checking if is Ansible Automation Platform
2.4 at>
May 22 13:02:13 automation-dashboard automation-dashboard-task[1607]: INFO 2025-05-22
13:02:13,821 connector 2 140568371550016 Pinging api https://ec2-3>
May 22 13:02:13 automation-dashboard automation-dashboard-task[1607]: INFO 2025-05-22
13:02:13,821 connector 2 140568371550016 Executing GET request to >
May 22 13:02:16 automation-dashboard automation-dashboard-task[1607]: ERROR 2025-05-22
13:02:16,892 connector 2 140568371550016 GET request failed with ...
```

You can check how the services are running by using `systemd`:

```
systemctl status --user

● automation-dashboard
    State: running
    Units: 76 loaded (incl. loaded aliases)
     Jobs: 0 queued
   Failed: 0 units
    Since: Thu 2025-05-22 13:02:07 UTC; 22min ago
  systemd: 252-51.el9
   CGroup: /user.slice/user-1000.slice/user@1000.service
           ├──app.slice
           │  ├──automation-dashboard-task.service
           │  │  └──1607 /usr/bin/conmon --api-version 1 -c
84e46532e8ca31b0cadb037479289d030103aa01b7a1591e62b83b17f031e47d -u
84e46532e8ca31b0cadb037479>
           │  ├──automation-dashboard-web.service
           │  │  └──1608 /usr/bin/conmon --api-version 1 -c
```

```
d060f3e3fb2b4c4c5c588149253beed83c78ccc9c9a8c1bf4c96157142a210dc -u
d060f3e3fb2b4c4c5c58814925>
            |  ├─dbus-broker.service
            |  |  ├─1621 /usr/bin/dbus-broker-launch --scope user
            |  |  └─1624 dbus-broker --log 4 --controller 9 --machine-id
612db98503014199bfd8c788c8d3da58 --max-bytes 100000000000000 --max-fds 2500000000>
            |  └─postgresql.service
            |    └─1614 /usr/bin/conmon --api-version 1 -c
eec61745cb6fc3a89a4f7475d7ef63b5899699157d943c2f16a3243311927bef -u
eec61745cb6fc3a89a4f7475d7>
            ├─init.scope
            |  ├─1093 /usr/lib/systemd/systemd --user
            |  └─1128 "(sd-pam)"
            └─user.slice
              ├─libpod-
84e46532e8ca31b0cadb037479289d030103aa01b7a1591e62b83b17f031e47d.scope
              |  └─container
              |    ├─1619 /usr/bin/dumb-init -- /usr/bin/launch_dashboard_task.sh
              |    └─1681 /venv/bin/python periodic.py
              ├─libpod-
d060f3e3fb2b4c4c5c588149253beed83c78ccc9c9a8c1bf4c96157142a210dc.scope
              |  └─container
              |    ├─1617 /usr/bin/dumb-init -- /usr/bin/launch_dashboard_web.sh
              |    ├─1646 /usr/bin/python3.9 /usr/local/bin/supervisord -c
/etc/supervisord_dashboard_web.conf
              |    ├─1877 /bin/bash /usr/local/bin/stop-supervisor
              |    ├─1878 "nginx: master process nginx -g daemon off;"
              |    ├─1879 /venv/bin/uwsgi /etc/tower/uwsgi.ini
              |    ├─1880 "nginx: worker process"
              |    ├─1881 "nginx: worker process"
              |    ├─1882 "nginx: worker process"
              |    ├─1883 "nginx: worker process"
              |    ├─1884 /venv/bin/uwsgi /etc/tower/uwsgi.ini
              |    ├─1885 /venv/bin/uwsgi /etc/tower/uwsgi.ini
              |    ├─1886 /venv/bin/uwsgi /etc/tower/uwsgi.ini
              |    ├─1887 /venv/bin/uwsgi /etc/tower/uwsgi.ini
              |    └─1888 /venv/bin/uwsgi /etc/tower/uwsgi.ini
              ├─libpod-
eec61745cb6fc3a89a4f7475d7ef63b5899699157d943c2f16a3243311927bef.scope
              |  └─container
              |    ├─1623 postgres
              |    ├─1869 "postgres: logger "
              |    ├─1871 "postgres: checkpointer "
              |    ├─1872 "postgres: background writer "
              |    ├─1873 "postgres: walwriter "
              |    ├─1874 "postgres: autovacuum launcher "
              |    ├─1875 "postgres: stats collector "
              |    ├─1876 "postgres: logical replication launcher "
              |    └─1889 "postgres: Ansible Automation Platformreporter Ansible
Automation Platformreports 172.31.28.99(39338) idle"
              └─podman-pause-b6c4e853.scope
```

```
               └─1359 catatonit -P
```

1.3. Uninstalling Automation Dashboard

Run the following command to uninstall Automation Dashboard and its dependencies, including the PostgreSQL database container:

+

```
ansible-playbook -i inventory
ansible.containerized_installer.dashboard_uninstall
```

2. Filter and save automation data for reporting

Automation Dashboard provides filtering options to analyze your Ansible Automation Platform automation runs. You can select one or more filtering options to customize your report, select a time period and a currency, and save your report to your Automation Dashboard.

2.1. Filters

Select one or more of the following filtering options to customize your report:

- **Template:** allows you to select one or more Job Templates
- **Organization:** allows you to select one or more Organizations
- **Project:** allows you to select one or more Projects
- **Label:** allows you to select one or more automations by label. Labels must be preconfigured and assigned to Ansible Automation Platform in order to be displayed in Automation Dashboard. For more information on configuring labels, see Creating a job template.

2.2. Time period and currency

After selecting your filters, select a time period for analysis, and select a currency to demonstrate automation savings.

- A shorter time period is useful when considering specific automation use cases.
- A longer time period is useful when considering overall platform usage and automation growth.
- Changing from one currency to another does not convert the value of that currency. You must manually change the manual and automation cost figures to reflect whichever currency you select.

2.3. Saving a report

Use **Save as Report** to save this report to your Automation Dashboard. You can retrieve it at any time by using **Select a Report**.

3. Summary of top and overview usage

Automation Dashboard displays a summary of the top and overview usage for your selected report. This includes the following data:

- **Total number of successful jobs**: This indicates the number of automation jobs that were completed successfully.
- **Total number of failed jobs**: This shows the number of automation jobs that encountered errors. Analyzing these failures can help improve automation throughput, reduce errors, and improve efficiency.
- **Total number of unique hosts automated**: This is the number of Controller inventory records you have automated.
- **Total hours of automation**: This represents the cumulative time that Ansible Automation Platform spent running jobs.
- **Number of times jobs were run**: This is the total number of individual job executions.
- **Number of hosts jobs are running on**: This is the total number of hosts that jobs are executed upon.
- **Top 5 projects**: This section lists the top five automation projects based on the number of running jobs.
- **Top 5 users**: This section lists the top five users of Ansible Automation Platform, with a breakdown of the total number of jobs run by each user.

NOTE	Scheduled jobs can affect these results, because they do not represent a real, logged-in user.

4. Analyzing costs and savings

The costs and savings analysis compares the cost of manual automation versus the cost of automation execution using Ansible Automation Platform in order to determine the total savings derived from automation execution.

To calculate your savings, use the following procedure:

Procedure

1. Use the **Average cost per minute to manually run the job** field to enter the average cost per minute for an engineer to manually run jobs.
2. Use the **Average cost per minute of running on Ansible Automation Platform** field to enter an average cost per minute of running a job using Ansible Automation Platform. You can

include the costs associated with creating the initial or ongoing automation execution.

3. Select **Time taken to create automation into calculation** to include the costs associated with creating the initial or ongoing automation execution.

Based on these inputs, Automation Dashboard provides the following data:

- **Cost of manual automation**: This number represents the estimated cost of manually performing all of the automated tasks. This is an estimated value, not an actual expenditure. It represents the potential expenses that the organization would incur without automation. The calculation is based on the time taken to manually execute each job, multiplied by a labor cost rate.
- **Cost of automated execution**: This is the cost of automation execution using Ansible Automation Platform. This cost includes the resources consumed by Ansible Automation Platform, such as server time, processing power, and any other operational expenses associated with automation execution. It represents the actual cost incurred for automation execution.
- **Total savings/cost avoided**: This is the difference between the **Cost of manual automation** and the **Cost of automated execution**. This is a key metric for demonstrating the return on investment attainable by using Ansible Automation Platform.
- **Total hours saved/avoided**: This figure is calculated by adding host executions and automation creation time, then subtracting the running time in minutes.
- **Time taken to manually execute (min)**: This metric represents the amount of time it would take for a user to perform the task manually on a host. It is an input provided to compare the value of manual execution and automated execution using the time taken by the organization to manually automate.

NOTE	You can export the data from your cost and savings analysis as a CSV.

5. Appendix

The inventory variables required by the Automation Dashboard installer are described in the following table:

5.1. Inventory variables

The following variables control how Automation Dashboard interacts with remote hosts.

Inventory variable	Description
`aap_auth_provider_name`	Natural language name shown on the login page. Default: `Ansible Automation Platform`
`aap_auth_provider_protocol`	Enter http or https
`aap_auth_provider_aap_version`	Enter one value - the version of AAP , valid values are 2.4, 2.5 or 2.6
`aap_auth_provider_host`	Ansible Automation Platform IP or DNS name, with optional port

Inventory variable	Description
aap_auth_provider_check_ssl	Enforce TLS check or not
aap_auth_provider_client_id	Ansible Automation Platform OAuth2 application client_id
aap_auth_provider_client_secret	Ansible Automation Platform OAuth2 application client_secret
initial_sync_days	Requests x number of days of old data, counting from "today"
initial_sync_since	Requests data from the specified data until "today"
dashboard_update_secret	Forces regeneration of autogenerated podman secrets. A podman secret is used to store the password for database access. Set dashboard_update_secret to true if you changed the dashboard_pg_password in inventory.
nginx_disable_https	Allows using http instead of https for Automation Dashboard.
nginx_http_port	Configures the HTTP port for Automation Dashboard.
nginx_https_port	Configures the HTTPS port for Automation Dashboard.
dashboard_tls_cert	TLS server certificate for dashboard.
dashboard_tls_key	TLS server certificate key for dashboard.
postgresql_admin_username	Admin username to access PostgreSQL database.
postgresql_admin_password	Admin password to access PostgreSQL database.
registry_username	Admin password to access PostgreSQL database.
registry_username	Username used to pull container images from registry.redhat.io. Currently we distribute only a bundled installer, so the tarball contains required container images, and the end user does not need to pull images from the remote registry. The variable is needed when the bundled installer is being built. **NOTE** End users can omit this variable.
registry_username	Username used to pull container images from registry.redhat.io. Currently we distribute only a bundled installer, so the tarball contains required container images, and the end user does not need to pull images from the remote registry. The variable is needed when the bundled installer is being built. **NOTE** End users can omit this variable.
registry_password	Password used to pull container images from registry.redhat.io. The variable is needed when the bundled installer is being built. **NOTE** End users can omit this variable.

Inventory variable	Description
dashboard_pg_containerized	This configures the installer to install and configure the database as a container on the same host as Automation Dashboard. True is also the only supported value.
dashboard_admin_password	This is the password for the Automation Dashboard administrator user. The username is always admin.
dashboard_pg_host dashboard_pg_username dashboard_pg_password dashboard_pg_database	The dashboard_pg_* variables configure additional database user and database schema for Automation Dashboard on the database host dashboard_pg_host.
bundle_install	Indicates the required container images are already included in the installation bundle (tarball). It must be true.
bundle_dir	This is the directory where the installation bundle was unpacked + /bundle (for example: /home/<username>/ansible-automation-dashboard-containerized-setup/bundle). The default value is relative to the current directory (PWD) and should work out of the box.