Post installation configurations to ensure a smooth deployment of Ansible Automation Platform installation
Post installation configurations to ensure a smooth deployment of Ansible Automation Platform installation
Abstract

This guide provides instructions and guidance on post installation activities for Red Hat Ansible Automation Platform.
# Table of Contents

**PREFACE** .......................................................... 3

**MAKING OPEN SOURCE MORE INCLUSIVE** .............................................. 4

**PROVIDING FEEDBACK ON RED HAT DOCUMENTATION** ............................. 5

**CHAPTER 1. ACTIVATING RED HAT ANSIBLE AUTOMATION PLATFORM** .......... 6

1.1. ACTIVATE WITH CREDENTIALS ............................................. 6

1.2. ACTIVATE WITH A MANIFEST FILE ...................................... 6

**CHAPTER 2. OBTAINING A MANIFEST FILE** ........................................ 8

2.1. CREATE A SUBSCRIPTION ALLOCATION ................................... 8

2.2. ADDING SUBSCRIPTIONS TO A SUBSCRIPTION ALLOCATION ............. 8

2.3. DOWNLOADING A MANIFEST FILE ......................................... 9

**CHAPTER 3. CONFIGURING PROXY SUPPORT FOR RED HAT ANSIBLE AUTOMATION PLATFORM** ........................................... 10

3.1. ENABLE PROXY SUPPORT .................................................. 10

3.2. KNOWN PROXIES .......................................................... 10

3.2.1. Configuring known proxies .......................................... 10

3.3. CONFIGURING A REVERSE PROXY ....................................... 11

3.4. ENABLE STICKY SESSIONS ................................................. 12

**CHAPTER 4. CONFIGURING AUTOMATION CONTROLLER WEBSOCKET CONNECTIONS** ........................................ 13

4.1. WEBSOCKET CONFIGURATION FOR AUTOMATION CONTROLLER .......... 13

4.1.1. Configuring automatic discovery of other automation controller nodes .......................................................... 13

**CHAPTER 5. MANAGING USABILITY ANALYTICS AND DATA COLLECTION FROM AUTOMATION CONTROLLER** .............................................. 14

5.1. USABILITY ANALYTICS AND DATA COLLECTION .......................... 14

5.1.1. Controlling data collection from automation controller ................ 14

**CHAPTER 6. ENCRYPTING PLAINTEXT PASSWORDS IN AUTOMATION CONTROLLER CONFIGURATION FILES** .............................................. 15

6.1. CREATING POSTGRESQL PASSWORD HASHES ............................ 15

6.2. Encrypting the Postgres password ....................................... 15

6.3. RESTARTING AUTOMATION CONTROLLER SERVICES ...................... 16
After installing Red Hat Ansible Automation Platform, your system might need extra configuration to ensure your deployment runs smoothly. This guide provides procedures for configuration tasks that you can perform after installing Red Hat Ansible Automation Platform.
MAKING OPEN SOURCE MORE INCLUSIVE

Red Hat is committed to replacing problematic language in our code, documentation, and web properties. We are beginning with these four terms: master, slave, blacklist, and whitelist. Because of the enormity of this endeavor, these changes will be implemented gradually over several upcoming releases. For more details, see our CTO Chris Wright’s message.
PROVIDING FEEDBACK ON RED HAT DOCUMENTATION

We appreciate your feedback on our technical content and encourage you to tell us what you think. If you’d like to add comments, provide insights, correct a typo, or even ask a question, you can do so directly in the documentation.

**NOTE**

You must have a Red Hat account and be logged in to the customer portal.

To submit documentation feedback from the customer portal, do the following:

1. Select the **Multi-page HTML** format.
2. Click the **Feedback** button at the top-right of the document.
3. Highlight the section of text where you want to provide feedback.
4. Click the **Add Feedback** dialog next to your highlighted text.
5. Enter your feedback in the text box on the right of the page and then click **Submit**.

We automatically create a tracking issue each time you submit feedback. Open the link that is displayed after you click **Submit** and start watching the issue or add more comments.
CHAPTER 1. ACTIVATING RED HAT ANSIBLE AUTOMATION PLATFORM

Red Hat Ansible Automation Platform uses available subscriptions or a subscription manifest to authorize the use of Ansible Automation Platform. To obtain a subscription, you can do either of the following:

1. Use your Red Hat customer or Satellite credentials when you launch Ansible Automation Platform.

2. Upload a subscriptions manifest file either using the Red Hat Ansible Automation Platform interface or manually in an Ansible playbook.

1.1. ACTIVATE WITH CREDENTIALS

When Ansible Automation Platform launches for the first time, the Ansible Automation Platform Subscription screen automatically displays. You can use your Red Hat credentials to retrieve and import your subscription directly into Ansible Automation Platform.

Procedures

1. Enter your Red Hat username and password.

2. Click Get Subscriptions.

   NOTE

   You can also use your Satellite username and password if your cluster nodes are registered to Satellite through Subscription Manager.

3. Review the End User License Agreement and select I agree to the End User License Agreement.

4. The Tracking and Analytics options are checked by default. These selections help Red Hat improve the product by delivering you a much better user experience. You can opt out by deselecting the options.

5. Click Submit.

6. Once your subscription has been accepted, the license screen displays and navigates you to the Dashboard of the Ansible Automation Platform interface. You can return to the license screen by clicking the Settings icon and selecting the License tab from the Settings screen.

1.2. ACTIVATE WITH A MANIFEST FILE

If you have a subscriptions manifest, you can upload the manifest file either using the Red Hat Ansible Automation Platform interface or manually in an Ansible playbook.

Prerequisites

You must have a Red Hat Subscription Manifest file exported from the Red Hat Customer Portal. For more information, see Obtaining a manifest file.
Uploading with the interface

1. Complete steps to generate and download the manifest file
2. Log in to Red Hat Ansible Automation Platform.
3. If you are not immediately prompted for a manifest file, go to **Settings → License**.
4. Make sure the **Username** and **Password** fields are empty.
5. Click **Browse** and select the manifest file.
6. Click **Next**.

**NOTE**

If the **BROWSE** button is disabled on the License page, clear the **USERNAME** and **PASSWORD** fields.

Uploading manually

If you are unable to apply or update the subscription info using the Red Hat Ansible Automation Platform interface, you can upload the subscriptions manifest manually in an Ansible playbook using the **license** module in the **ansible.controller** collection.

```yaml
- name: Set the license using a file
  license:
    manifest: "/tmp/my_manifest.zip"
```
CHAPTER 2. OBTAINING A MANIFEST FILE

You can obtain a subscription manifest in the Subscription Allocations section of Red Hat Subscription Management. After you obtain a subscription allocation, you can download its manifest file and upload it to activate Ansible Automation Platform.

To begin, login to the Red Hat Customer Portal using your administrator user account and follow the procedures in this section.

2.1. CREATE A SUBSCRIPTION ALLOCATION

Creating a new subscription allocation allows you to set aside subscriptions and entitlements for a system that is currently offline or air-gapped. This is necessary before you can download its manifest and upload it to Ansible Automation Platform.

Procedure

1. From the Subscription Allocations page, click New Subscription Allocation.

2. Enter a name for the allocation so that you can find it later.

3. Select Type: Satellite 6.8 as the management application.

4. Click Create.

2.2. ADDING SUBSCRIPTIONS TO A SUBSCRIPTION ALLOCATION

Once an allocation is created, you can add the subscriptions you need for Ansible Automation Platform to run properly. This step is necessary before you can download the manifest and add it to Ansible Automation Platform.

Procedure

1. From the Subscription Allocations page, click on the name of the Subscription Allocation to which you would like to add a subscription.

2. Click the Subscriptions tab.

3. Click Add Subscriptions.

4. Enter the number of Ansible Automation Platform Entitlement(s) you plan to add.

5. Click Submit.

Verification

After your subscription has been accepted, subscription details are displayed. A status of Compliant indicates your subscription is in compliance with the number of hosts you have automated within your subscription count. Otherwise, your status will show as Out of Compliance, indicating you have exceeded the number of hosts in your subscription.

Other important information displayed include the following:

Hosts automated
Host count automated by the job, which consumes the license count

**Hosts imported**
Host count considering all inventory sources (does not impact hosts remaining)

**Hosts remaining**
Total host count minus hosts automated

### 2.3. DOWNLOADING A MANIFEST FILE

After an allocation is created and has the appropriate subscriptions on it, you can download the manifest from Red Hat Subscription Management.

**Procedure**

1. From the Subscription Allocations page, click on the name of the Subscription Allocation to which you would like to generate a manifest.
2. Click the Subscriptions tab.
3. Click Export Manifest to download the manifest file.

**NOTE**

The file is saved to your default downloads folder and can now be uploaded to activate Red Hat Ansible Automation Platform.
CHAPTER 3. CONFIGURING PROXY SUPPORT FOR RED HAT ANSIBLE AUTOMATION PLATFORM

You can configure Red Hat Ansible Automation Platform to communicate with traffic using a proxy. Proxy servers act as an intermediary for requests from clients seeking resources from other servers. A client connects to the proxy server, requesting some service or available resource from a different server, and the proxy server evaluates the request as a way to simplify and control its complexity. The following sections describe the supported proxy configurations and how to set them up.

3.1. ENABLE PROXY SUPPORT

To provide proxy server support, automation controller handles proxied requests (such as ALB, NLB, HAProxy, Squid, Nginx and tinyproxy in front of automation controller) via the REMOTE_HOST_HEADERS list variable in the automation controller settings. By default, REMOTE_HOST_HEADERS is set to ["REMOTE_ADDR", "REMOTE_HOST"].

To enable proxy server support, edit the REMOTE_HOST_HEADERS field in the settings page for your automation controller:

Procedure

1. On your automation controller, navigate to Settings → Miscellaneous System.

2. In the REMOTE_HOST_HEADERS field, enter the following values:

   ```
   ["HTTP_X_FORWARDED_FOR",
    "REMOTE_ADDR",
    "REMOTE_HOST"
   ]
   ```

Automation controller determines the remote host’s IP address by searching through the list of headers in REMOTE_HOST_HEADERS until the first IP address is located.

3.2. KNOWN PROXIES

When automation controller is configured with REMOTE_HOST_HEADERS = ["HTTP_X_FORWARDED_FOR", "REMOTE_ADDR", "REMOTE_HOST"], it assumes that the value of X-Forwarded-For has originated from the proxy/load balancer sitting in front of automation controller. If automation controller is reachable without use of the proxy/load balancer, or if the proxy does not validate the header, the value of X-Forwarded-For can be falsified to fake the originating IP addresses. Using HTTP_X_FORWARDED_FOR in the REMOTE_HOST_HEADERS setting poses a vulnerability.

To avoid this, you can configure a list of known proxies that are allowed using the PROXY_IP_ALLOWED_LIST field in the settings menu on your automation controller. Load balancers and hosts that are not on the known proxies list will result in a rejected request.

3.2.1. Configuring known proxies

To configure a list of known proxies for your automation controller, add the proxy IP addresses to the PROXY_IP_ALLOWED_LIST field in the settings page for your automation controller.

Procedure
1. On your automation controller, navigate to Settings → Miscellaneous System.

2. In the PROXY_IP_ALLOWED_LIST field, enter IP addresses that are allowed to connect to your automation controller, following the syntax in the example below:

   Example PROXY_IP_ALLOWED_LIST entry

   ```
   [  
     "example1.proxy.com:8080",  
     "example2.proxy.com:8080"  
   ]
   ```

   IMPORTANT

   - PROXY_IP_ALLOWED_LIST requires proxies in the list are properly sanitizing header input and correctly setting an X-Forwarded-For value equal to the real source IP of the client. Automation controller can rely on the IP addresses and hostnames in PROXY_IP_ALLOWED_LIST to provide non-spoofed values for the X-Forwarded-For field.

   - Do not configure HTTP_X_FORWARDED_FOR as an item in `REMOTE_HOST_HEADERS` unless all of the following conditions are satisfied:

     - You are using a proxied environment with ssl termination;
     - The proxy provides sanitization or validation of the X-Forwarded-For header to prevent client spoofing;
     - `/etc/tower/conf.d/remote_host_headers.py` defines PROXY_IP_ALLOWED_LIST that contains only the originating IP addresses of trusted proxies or load balancers.

3.3. CONFIGURING A REVERSE PROXY

You can support a reverse proxy server configuration by adding HTTP_X_FORWARDED_FOR to the REMOTE_HOST_HEADERS field in your automation controller settings. The X-Forwarded-For (XFF) HTTP header field identifies the originating IP address of a client connecting to a web server through an HTTP proxy or load balancer.

Procedure

1. On your automation controller, navigate to Settings → Miscellaneous System.

2. In the REMOTE_HOST_HEADERS field, enter the following values:

   ```
   [  
     "HTTP_X_FORWARDED_FOR",  
     "REMOTE_ADDR",  
     "REMOTE_HOST"  
   ]
   ```

3. Add the lines below to `/etc/tower/conf.d/custom.py` to ensure the application uses the correct headers:
USE_X_FORWARDED_PORT = True
USE_X_FORWARDED_HOST = True

3.4. ENABLE STICKY SESSIONS

By default, an Application Load Balancer routes each request independently to a registered target based on the chosen load-balancing algorithm. To avoid authentication errors when running multiple instances of automation hub behind a load balancer, you must enable sticky sessions. Enabling sticky sessions sets a custom application cookie that matches the cookie configured on the load balancer to enable stickiness. This custom cookie can include any of the cookie attributes required by the application.

Additional resources

- Refer to Sticky sessions for your Application Load Balancer for more information about enabling sticky sessions.

Disclaimer: Links contained in this note to external website(s) are provided for convenience only. Red Hat has not reviewed the links and is not responsible for the content or its availability. The inclusion of any link to an external website does not imply endorsement by Red Hat of the website or their entities, products or services. You agree that Red Hat is not responsible or liable for any loss or expenses that may result due to your use of (or reliance on) the external site or content.
CHAPTER 4. CONFIGURING AUTOMATION CONTROLLER WEBSOCKET CONNECTIONS

You can configure automation controller in order to align thewebsocket configuration with your nginx or load balancer configuration.

4.1. WEBSOCKET CONFIGURATION FOR AUTOMATION CONTROLLER

Automation controller nodes are interconnected through websockets to distribute allwebsocket-emitted messages throughout your system. This configuration setup enables any browser clientwebsocket to subscribe to any job that might be running on any automation controller node. Websocket clients are not routed to specific automation controller nodes. Instead, any automation controller node can handle anywebsocket request and each automation controller node must know about allwebsocket messages destined for all clients.

You can configure websockets at `/etc/tower/conf.d/websocket_config.py` in all of your automation controller nodes and the changes will be effective after the service restarts.

Automation controller automatically handles discovery of other automation controller nodes through the Instance record in the database.

IMPORTANT

Your automation controller nodes are designed to broadcastwebsocket traffic across a private, trusted subnet (and not the open Internet). Therefore, if you turn off HTTPS forwebsocket broadcasting, thewebsocket traffic, composed mostly of Ansible playbook stdout, is sent unencrypted between automation controller nodes.

4.1.1. Configuring automatic discovery of other automation controller nodes

You can configure websocket connections to enable automation controller to automatically handle discovery of other automation controller nodes through the Instance record in the database.

- Edit automation controllerwebsocket information for port and protocol, and confirm whether to verify certificates with `True` or `False` when establishing thewebsocket connections.

```
BROADCAST_WEBSOCKET_PROTOCOL = 'http'
BROADCAST_WEBSOCKET_PORT = 80
BROADCAST_WEBSOCKET_VERIFY_CERT = False
```
CHAPTER 5. MANAGING USABILITY ANALYTICS AND DATA COLLECTION FROM AUTOMATION CONTROLLER

You can change how you participate in usability analytics and data collection from automation controller by opting out or changing your settings in the automation controller user interface.

5.1. USABILITY ANALYTICS AND DATA COLLECTION

Usability data collection is included with automation controller to collect data to better understand how automation controller users specifically interact with automation controller, to help enhance future releases, and to continue streamlining your user experience.

Only users installing a trial of automation controller or a fresh installation of automation controller are opted-in for this data collection.

Additional resources

- For more information, see the Red Hat privacy policy.

5.1.1. Controlling data collection from automation controller

You can control how automation controller collects data by setting your participation level in the User Interface tab in the Settings menu.

Procedure

1. Log in to your automation controller.

2. Navigate to Settings → User Interface.

3. Select the desired level of data collection from the User Analytics Tracking State drop-down list:
   - **Off**: Prevents any data collection.
   - **Anonymous**: Enables data collection without your specific user data.
   - **Detailed**: Enables data collection including your specific user data.

4. Click **Save** to apply the settings or **Cancel** to discard the changes.
Passwords stored in automation controller configuration files are stored in plain text. A user with access to the `/etc/tower/conf.d/` directory can view the passwords used to access the database. Access to the directories is controlled with permissions, so they are protected, but some security findings deem this protection to be inadequate. The solution is to encrypt the passwords individually.

### 6.1. CREATING POSTGRESQL PASSWORD HASHES

**Procedure**

1. On your automation controller node, run the following:

   ```bash
   # awx-manage shell_plus
   ```

2. Then run the following from the python prompt:

   ```python
   >>> from awx.main.utils import encrypt_value, get_encryption_key
   >>> postgres_secret = encrypt_value('$POSTGRES_PASS')
   >>> print(postgres_secret)
   ```

   **NOTE**
   Replace the `$POSTGRES_PASS` variable with the actual plain text password you wish to encrypt.

   The output should resemble the following:

   ```plaintext
   $encrypted$UTF8$AESCBC$Z0FBQUFBQmtLdGRWVFJwGtkV1ZBR3hkNGVVbFFIU3hhY21UT081eXFkR09aUWZLcG9TSmpndmZYQXyRHFQ3ZYGSE15OUFuM1RHZHBoTFU3S0MyNEo2Y2JWUURSYktsdmc9PQ==
   ```

3. Copy the full values of these hashes and save them.

   - The hash value begins with `$encrypted$`, and is not just the string of characters, as shown in the following example:

     ```plaintext
     $encrypted$AESCBC$Z0FBQUFBQmNONU9BbGQ1VjJyNDJRTtRtKHFRFRI09l2U5TgdBYVrlcXFRjimdmpZNd0ZVvE21QRWViMmNDOGJaM0dPeHN2b194NVxQ1M5X3dSc1gxQ29TdBKRLkljWHc9PQ==
     ```

   Note that the `$*_PASS` values are already in plain text in your inventory file.

These steps supply the hash values that replace the plain text passwords within the automation controller configuration files.

### 6.2. ENCRYPTING THE POSTGRES PASSWORD

The following procedure replaces the plain text passwords with encrypted values. Perform the following steps on each node in the cluster:
Procedure

1. Edit `/etc/tower/conf.d/postgres.py` using:

   ```bash
   $ vim /etc/tower/conf.d/postgres.py
   ```

2. Add the following line to the top of the file.

   ```python
   from awx.main.utils import decrypt_value, get_encryption_key
   ```

3. Remove the password value listed after `PASSWORD`: and replace it with the following line, replacing the supplied value of `$encrypted..` with your own hash value:

   ```python
   decrypt_value(get_encryption_key('value'), '$encrypted$AESCBC$Z0FBQUFBQmNONU9BbGQ1VjYoNDJRVTRkaFRIR09ib2U5TGdaYVRfcXFXRjimmdmpZNjoZVpEZ21QRWViMmNDOGJaM0dPeHN2b194NUxvQ1M5X3dSc1gxQ29TdDBKRkljWHc9PQ=='),
   ```

   **NOTE**
   The hash value in this step is the output value of `postgres_secret`.

4. The full `postgres.py` resembles the following:

   ```python
   # Ansible Automation platform controller database settings. from awx.main.utils import decrypt_value, get_encryption_key DATABASES = { 'default': { 'ATOMIC_REQUESTS': True, 'ENGINE': 'django.db.backends.postgresql', 'NAME': 'awx', 'USER': 'awx', 'PASSWORD': decrypt_value(get_encryption_key('value'), '$encrypted$AESCBC$Z0FBQUFBQmNONU9BbGQ1VjYoNDJRVTRkaFRIR09ib2U5TGdaYVRfcXFXRjimmdmpZNjoZVpEZ21QRWViMmNDOGJaM0dPeHN2b194NUxvQ1M5X3dSc1gxQ29TdDBKRkljWHc9PQ=='), 'HOST': '127.0.0.1', 'PORT': 5432, } }
   ```

6.3. RESTARTING AUTOMATION CONTROLLER SERVICES

Procedure

1. When encryption is completed on all nodes, perform a restart of services across the cluster using:

   ```bash
   # automation-controller-service restart
   ```

2. Navigate to the UI, and verify you are able to run jobs across all nodes.