



Red Hat Ansible Automation Platform 2.1

Deploying a high availability automation hub

Overview of the requirements and procedures for a high availability deployment of automation hub.

Red Hat Ansible Automation Platform 2.1 Deploying a high availability automation hub

Overview of the requirements and procedures for a high availability deployment of automation hub.

Legal Notice

Copyright © 2023 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux[®] is the registered trademark of Linus Torvalds in the United States and other countries.

Java[®] is a registered trademark of Oracle and/or its affiliates.

XFS[®] is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL[®] is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js[®] is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack[®] Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

Abstract

Providing Feedback: If you have a suggestion to improve this documentation, or find an error, please contact technical support at [to create an issue on the Ansible Automation Platform Jira project](#) using the Docs component.

Table of Contents

PREFACE	3
MAKING OPEN SOURCE MORE INCLUSIVE	4
CHAPTER 1. REQUIREMENTS FOR A HIGH AVAILABILITY AUTOMATION HUB	5
1.1. REQUIRED SHARED FILESYSTEM	5
1.2. NETWORK STORAGE INSTALLATION REQUIREMENTS	5
CHAPTER 2. INSTALLING A HIGH AVAILABILITY AUTOMATION HUB	6
2.1. HIGHLY AVAILABLE AUTOMATION HUB INSTALLATION	6
2.2. INSTALL A HIGH AVAILABILITY (HA) DEPLOYMENT OF AUTOMATION HUB ON SELINUX	6

PREFACE

This guide provides an overview of the requirements and procedures for a high availability deployment of your automation hub.

A high availability (HA) configuration increases reliability and scalability for automation hub deployments.

HA deployments of automation hub have multiple nodes that concurrently run the same service with a load balancer distributing workload (an "active-active" configuration). This configuration eliminates single points of failure to minimize service downtime and allows you to easily add or remove nodes to meet workload demands.

This guide covers deployment of a HA automation hub application stack only. Other HA components, such as database and file system HA, or setting up DNS load balancing, are out of scope for this guide.

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](#).

CHAPTER 1. REQUIREMENTS FOR A HIGH AVAILABILITY AUTOMATION HUB

Before deploying a high availability (HA) automation hub, ensure that you have a shared filesystem installed in your environment and that you have configured your network storage system, if applicable.

1.1. REQUIRED SHARED FILESYSTEM

A high availability automation hub requires you to have a shared file system, such as NFS, already installed in your environment. Before you run the Red Hat Ansible Automation Platform installer, verify that the **/var/lib/pulp** directory exists across your cluster as a result of your shared file system installation. The Red Hat Ansible Automation Platform installer will return an error if **/var/lib/pulp** is not detected in one of your nodes, causing your HA automation hub setup to fail.

1.2. NETWORK STORAGE INSTALLATION REQUIREMENTS

If you intend to install a HA automation hub using a network storage on the automation hub nodes itself, you must first install and use **firewalld** to open the necessary ports as required by your shared storage system before running the Ansible Automation Platform installer.

Install and configure **firewalld** by executing the following commands:

1. Install the **firewalld** daemon:

```
$ dnf install firewalld
```

2. Add your network storage under <service> using the following command:

```
$ firewall-cmd --permanent --add-service=<service>
```



NOTE

For a list of supported services, use the **\$ firewall-cmd --get-services** command

3. Reload to apply the configuration:

```
$ firewall-cmd --reload
```

CHAPTER 2. INSTALLING A HIGH AVAILABILITY AUTOMATION HUB

Configure the Ansible Automation Platform installer to install automation hub in a highly available (HA) configuration. Install HA automation hub on SELinux by creating mount points and adding the appropriate SELinux contexts to your Ansible Automation Platform environment.

2.1. HIGHLY AVAILABLE AUTOMATION HUB INSTALLATION

Install a highly available automation hub by making the following changes to the **inventory** file in the Ansible Automation Platform installer, then running the **./setup.sh** script:

Specify database host IP

Specify the IP address for your database host, using the **automation_pg_host** and **automation_pg_port** fields. For example:

```
automationhub_pg_host='192.0.2.10'  
automationhub_pg_port='5432'
```

also specify the IP address for your database host in the [database] section, using the value in the **automationhub_pg_port** field:

```
[database]  
192.0.2.10
```

List all instances in a clustered setup

If installing a clustered setup, replace **localhost ansible_connection=local** in the [automationhub] section with the hostname or IP of all instances. For example:

```
[automationhub]  
automationhub1.testing.ansible.com ansible_user=cloud-user ansible_host=192.0.2.18  
automationhub2.testing.ansible.com ansible_user=cloud-user ansible_host=192.0.2.20  
automationhub3.testing.ansible.com ansible_user=cloud-user ansible_host=192.0.2.22
```

Red Hat Single Sign-On requirements

If you are implementing Red Hat Single Sign-On on your automation hub environment, specify the main automation hub URL that clients will connect to, using the **automationhub_main_url** field. For example:

```
automationhub_main_url = 'https://automationhub.ansible.com'
```



NOTE

If **automationhub_main_url** is not specified, the first node in the [automationhub] group will be used as default.

2.2. INSTALL A HIGH AVAILABILITY (HA) DEPLOYMENT OF AUTOMATION HUB ON SELINUX

To set up a high availability (HA) deployment of automation hub on SELinux, create two mount points for **/var/lib/pulp** and **/var/lib/pulp/pulpcore_static**, then assign the appropriate SELinux contexts to each. You must add the context for **/var/lib/pulp/pulpcore_static** and run the Ansible Automation Platform installer before adding the context for **/var/lib/pulp**.

Prerequisites

- You have already configured a NFS export on your server.

Pre-installation procedure

1. Create a mount point at **/var/lib/pulp**:

```
$ mkdir /var/lib/pulp/
```

2. Open **/etc/fstab** using a text editor, then add the following values:

```
srv_rhel8:/data /var/lib/pulp nfs defaults,_netdev,nosharecache 0 0
srv_rhel8:/data/pulpcore_static /var/lib/pulp/pulpcore_static nfs
defaults,_netdev,nosharecache,context="system_u:object_r:httpd_sys_content_rw_t:s0" 0 0
```

3. Run the mount command for **/var/lib/pulp**:

```
$ mount /var/lib/pulp
```

4. Create a mount point at **/var/lib/pulp/pulpcore_static**:

```
$ mkdir /var/lib/pulp/pulpcore_static
```

5. Run the mount command:

```
$ mount -a
```

6. With the mount points set up, run the Ansible Automation Platform installer:

```
$ setup.sh -- -b --become-user root
```

Once the installation is complete, unmount the **/var/lib/pulp/** mount point then apply the appropriate SELinux context:

Post-installation procedure

1. Shut down the Pulp service:

```
$ systemctl stop pulpcore.service
```

2. Unmount **/var/lib/pulp/pulpcore_static**:

```
$ umount /var/lib/pulp/pulpcore_static
```

3. Unmount **/var/lib/pulp/**:

```
$ umount /var/lib/pulp/
```

4. Open **/etc/fstab** using a text editor, then replace the existing value for **/var/lib/pulp** with the following:

```
srv_rhel8:/data /var/lib/pulp nfs
defaults,_netdev,nosharecache,context="system_u:object_r:pulpcore_var_lib_t:s0" 0 0
```

5. Run the mount command:

```
$ mount -a
```

Configure **pulpcore.service**:

1. With the two mount points set up, shut down the Pulp service to configure **pulpcore.service**:

```
$ systemctl stop pulpcore.service
```

2. Edit **pulpcore.service** using **systemctl**:

```
$ systemctl edit pulpcore.service
```

3. Add the following entry to **pulpcore.service** to ensure that automation hub services starts only after starting the network and mounting the remote mount points:

```
[Unit]
After=network.target var-lib-pulp.mount
```

4. Enable **remote-fs.target**:

```
$ systemctl enable remote-fs.target
```

5. Reboot the system:

```
$ systemctl reboot
```

Troubleshooting

A bug in the pulpcore SELinux policies can cause the token authentication public/private keys in **/etc/pulp/certs/** to not have the proper SELinux labels, causing the pulp process to fail. When this occurs, run the following command to temporarily attach the proper labels:

```
$ chcon system_u:object_r:pulpcore_etc_t:s0 /etc/pulp/certs/token_{private,public}_key.pem
```



NOTE

You must repeat this command to reattach the proper SELinux labels whenever you relabel your system.

Additional Resources

- See the [SELinux Requirements on the Pulp Project documentation](#) for a list of SELinux contexts.