Red Hat Ansible Automation Platform 2.4

Troubleshooting Ansible Automation Platform

Troubleshoot issues with Ansible Automation Platform

Last Updated: 2024-05-23

Red Hat Ansible Automation Platform 2.4 Troubleshooting Ansible Automation Platform

Troubleshoot issues with Ansible Automation Platform

Legal Notice

Abstract

This guide provides troubleshooting topics for Red Hat Ansible Automation Platform.

Table of Contents

PREFACE

Use the Troubleshooting Ansible Automation Platform guide to troubleshoot your Ansible Automation Platform installation.

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

If you have a suggestion to improve this documentation, or find an error, please contact technical support at https://access.redhat.com to create an issue on the Ansible Automation Platform Jira project using the **docs-product** component.

CHAPTER 1. DIAGNOSING THE PROBLEM

To start troubleshooting Ansible Automation Platform, use the **must-gather** command on OpenShift Container Platform or the **sos** utility on a VM-based installation to collect configuration and diagnostic information. You can attach the output of these utilities to your support case.

1.1. TROUBLESHOOTING ANSIBLE AUTOMATION PLATFORM ON OPENSHIFT CONTAINER PLATFORM BY USING THE MUST-GATHER COMMAND

The **oc adm must-gather** command line interface (CLI) command collects information from your Ansible Automation Platform installation deployed on OpenShift Container Platform. It gathers information that is often needed for debugging issues, including resource definitions and service logs.

Running the **oc adm must-gather** CLI command creates a new directory containing the collected data that you can use to troubleshoot or attach to your support case.

If your OpenShift environment does not have access to **registry.redhat.io** and you cannot run the **must-gather** command, then run the **oc adm inspect** command instead.

Prerequisites

- The OpenShift CLI (**oc**) is installed.

Procedure

1. Log in to your cluster:

```
oc login <openshift_url>
```

2. Run one of the following commands based on your level of access in the cluster:
   - Run **must-gather** across the entire cluster:

```
oc adm must-gather --image=registry.redhat.io/ansible-automation-platform-24/aap-must-gather-rhel8 --dest-dir <dest_dir>
```

     - **--image** specifies the image that gathers data
     - **--dest-dir** specifies the directory for the output
   - Run **must-gather** for a specific namespace in the cluster:

```
oc adm must-gather --image=registry.redhat.io/ansible-automation-platform-24/aap-must-gather-rhel8 --dest-dir <dest_dir> – /usr/bin/ns-gather <namespace>
```

     - **– /usr/bin/ns-gather** limits the **must-gather** data collection to a specified namespace
3. To attach the **must-gather** archive to your support case, create a compressed file from the **must-gather** directory created before and attach it to your support case.
   - For example, on a computer that uses a Linux operating system, run the following command, replacing ***<must-gather-local.5421342344627712289/>*** with the **must-gather** directory name:

```
$ tar cvaf must-gather.tar.gz <must-gather.local.5421342344627712289/>
```

Additional resources

- For information about installing the OpenShift CLI (**oc**), see Installing the OpenShift CLI in the OpenShift Container Platform Documentation.
- For information about running the **oc adm inspect** command, see the ocm adm inspect section in the OpenShift Container Platform Documentation.

1.2. TROUBLESHOOTING ANSIBLE AUTOMATION PLATFORM ON VM-BASED INSTALLATIONS BY GENERATING AN SOS REPORT

The **sos** utility collects configuration, diagnostic, and troubleshooting data from your Ansible Automation Platform on a VM-based installation.

For more information about installing and using the **sos** utility, see Generating an sos report for technical support.

CHAPTER 2. RESOURCES FOR TROUBLESHOOTING AUTOMATION CONTROLLER

- For information about troubleshooting automation controller, see Troubleshooting automation controller in the Automation Controller Administration Guide.
- For information about troubleshooting the performance of automation controller, see Performance troubleshooting for automation controller in the Automation Controller Administration Guide.

CHAPTER 3. BACKUP AND RECOVERY

- For information about performing a backup and recovery of Ansible Automation Platform, see Backup and restore in the Automation Controller Administration Guide.
- For information about troubleshooting backup and recovery for installations of Ansible Automation Platform Operator on OpenShift Container Platform, see the Troubleshooting section in the Red Hat Ansible Automation Platform Operator Backup and Recovery Guide.

CHAPTER 4. EXECUTION ENVIRONMENTS

Troubleshoot issues with execution environments.

4.1. ISSUE - CANNOT SELECT THE "USE IN CONTROLLER" OPTION FOR EXECUTION ENVIRONMENT IMAGE ON PRIVATE AUTOMATION HUB

You cannot use the **Use in Controller** option for an execution environment image on private automation hub. You also receive the error message: "No Controllers available".

To resolve this issue, connect automation controller to your private automation hub instance.

Procedure

1. Change the **/etc/pulp/settings.py** file on private automation hub and add one of the following parameters depending on your configuration:
   - Single controller

     ```
     CONNECTED_ANSIBLE_CONTROLLERS = ['<https://my.controller.node>']
     ```

   - Many controllers behind a load balancer

     ```
     CONNECTED_ANSIBLE_CONTROLLERS = ['<https://my.controller.loadbalancer>']
     ```

   - Many controllers without a load balancer

     ```
     CONNECTED_ANSIBLE_CONTROLLERS = ['<https://my.controller.node1>',
     '<https://my.controller2.node2>']
     ```

2. Stop all of the private automation hub services:

   ```
   # systemctl stop pulpcore.service pulpcore-api.service pulpcore-content.service pulpcore-worker@1.service pulpcore-worker@2.service nginx.service redis.service
   ```

3. Restart all of the private automation hub services:

   ```
   # systemctl start pulpcore.service pulpcore-api.service pulpcore-content.service pulpcore-worker@1.service pulpcore-worker@2.service nginx.service redis.service
   ```

Verification

- Verify that you can now use the **Use in Controller** option in private automation hub.

CHAPTER 5. INSTALLATION

Troubleshoot issues with your installation.

5.1. ISSUE - CANNOT LOCATE CERTAIN PACKAGES THAT COME BUNDLED WITH THE ANSIBLE AUTOMATION PLATFORM INSTALLER

You cannot locate certain packages that come bundled with the Ansible Automation Platform installer, or you are seeing a "Repositories disabled by configuration" message.

To resolve this issue, enable the repository by using the **subscription-manager** command in the command line. For more information about resolving this issue, see the *Troubleshooting* section of Attaching your Red Hat Ansible Automation Platform subscription in the Red Hat Ansible Automation Platform Planning Guide.

CHAPTER 6. JOBS

Troubleshoot issues with jobs.

6.1. ISSUE - JOBS ARE FAILING WHEN RUN AGAINST LOCALHOST

With Ansible Automation Platform 2 and its containerized execution environments, the usage of localhost has changed. For more information, see Converting playbooks for AAP 2 in the Red Hat Ansible Automation Platform Upgrade and Migration Guide.

6.2. ISSUE - JOBS ARE FAILING WITH "ERROR! COULDN'T RESOLVE MODULE/ACTION" ERROR MESSAGE

Jobs are failing with the error message "ERROR! couldn't resolve module/action 'module name'. This often indicates a misspelling, missing collection, or incorrect module path".

This error can happen when the collection associated with the module is missing from the execution environment.

The recommended resolution is to create a custom execution environment and add the required collections inside of that execution environment. For more information about creating an execution environment, see Using Ansible Builder in Creating and Consuming Execution Environments.

Alternatively, you can complete the following steps:

Procedure

1. Create a **collections** folder inside of the project repository.
2. Add a **requirements.yml** file inside of the **collections** folder and add the collection:

```
collections:
- <collection_name>
```

6.3. ISSUE - JOBS ARE FAILING WITH "TIMEOUT (12S) WAITING FOR PRIVILEGE ESCALATION PROMPT" ERROR MESSAGE

This error can happen when the timeout value is too small, causing the job to stop before completion. The default timeout value for connection plugins is **10**.

To resolve the issue, increase the timeout value by completing one of the following procedures.

NOTE

The following changes will affect all of the jobs in automation controller. To use a timeout value for a specific project, add an **ansible.cfg** file in the root of the project directory and add the **timeout** parameter value to that **ansible.cfg** file.

Add ANSIBLE_TIMEOUT as an environment variable in the automation controller UI

1. Go to automation controller.

2. From the navigation panel, select **Settings → Jobs settings**.
3. Under **Extra Environment Variables** add the following:

```
{
"ANSIBLE_TIMEOUT": 60
}
```

Add a timeout value in the [defaults] section of the ansible.cfg file by using the CLI

- Edit the **/etc/ansible/ansible.cfg** file and add the following:

```
[defaults]
timeout = 60
```

Running ad hoc commands with a timeout

- To run an ad hoc playbook in the command line, add the **--timeout** flag to the **ansible-playbook** command, for example:

```
# ansible-playbook --timeout=60 <your_playbook.yml>
```

Additional resources

- For more information about the **DEFAULT_TIMEOUT** configuration setting, see DEFAULT_TIMEOUT in the Ansible Community Documentation.

6.4. ISSUE - JOBS IN AUTOMATION CONTROLLER ARE STUCK IN A PENDING STATE

After launching jobs in automation controller, the jobs stay in a pending state and do not start.

There are a few reasons jobs can become stuck in a pending state. For more information about troubleshooting this issue, see Playbook stays in pending in the Automation Controller Administration Guide.

Cancel all pending jobs

1. Run the following commands to list all of the pending jobs:

```
# awx-manage shell_plus
```

```
>>> UnifiedJob.objects.filter(status='pending')
```

2. Run the following command to cancel all of the pending jobs:

```
>>> UnifiedJob.objects.filter(status='pending').update(status='canceled')
```

Cancel a single job by using a job id

- To cancel a specific job, run the following commands, replacing ***<job_id>*** with the job id to cancel:

```
# awx-manage shell_plus
```

```
>>> UnifiedJob.objects.filter(id=_<job_id>_).update(status='canceled')
```

6.5. ISSUE - JOBS IN PRIVATE AUTOMATION HUB ARE FAILING WITH "DENIED: REQUESTED ACCESS TO THE RESOURCE IS DENIED, UNAUTHORIZED: INSUFFICIENT PERMISSIONS" ERROR MESSAGE

Jobs are failing with the error message "denied: requested access to the resource is denied, unauthorized: Insufficient permissions" when using an execution environment in private automation hub.

This issue happens when your private automation hub is protected with a password or token and the registry credential is not assigned to the execution environment.

Procedure

1. Go to automation controller.
2. From the navigation panel, select **Administration → Execution Environments**.
3. Click the execution environment assigned to the job template that is failing.
4. Click **Edit**.
5. Assign the appropriate **Registry credential** from your private automation hub to the execution environment.

Additional resources

- For information about creating new credentials in automation controller, see Creating new credentials in the Automation Controller User Guide.

CHAPTER 7. LOGIN

Troubleshoot login issues.

7.1. ISSUE - LOGGING IN TO THE AUTOMATION CONTROLLER UI RESULTS IN “INVALID USERNAME OR PASSWORD. PLEASE TRY AGAIN.”

When you try to log in to the automation controller UI, the login fails and you see the error message: “Invalid username or password. Please try again.”.

One reason this could be happening is if the value for **Maximum number of simultaneous logged in sessions** is **0**. The **Maximum number of simultaneous logged in sessions**value determines the maximum number of sessions allowed per user per device. If this value is **0**, no users can log in to automation controller.

The default value is **-1**, which disables the maximum sessions allowed. This means that you can have as many sessions without an imposed limit.

Procedure

- As root user, run the following command from the command line to set the **SESSIONS_PER_USER** variable to **-1** which disables the maximum sessions allowed:

```
# echo "settings.SESSIONS_PER_USER = -1" | awx-manage shell_plus --quiet
```

Verification

- Verify that you can log in successfully to automation controller.

Additional resources

- For more information about installing and using the controller node CLI, see AWX Command Line Interface and AWX manage utility.
- For more information about session limits, see Session Limits in the Automation Controller Administration Guide.

CHAPTER 8. NETWORKING

Troubleshoot networking issues.

8.1. ISSUE - THE DEFAULT SUBNET USED IN ANSIBLE AUTOMATION PLATFORM CONTAINERS CONFLICTS WITH THE INTERNAL NETWORK

The default subnet used in Ansible Automation Platform containers conflicts with the internal network resulting in "No route to host" errors.

To resolve this issue, update the default classless inter-domain routing (CIDR) value so it does not conflict with the CIDR used by the default Podman networking plugin.

Procedure

1. In all controller and hybrid nodes, run the following commands to create a file called **custom.py**:

```
# touch /etc/tower/conf.d/custom.py
```

```
# chmod 640 /etc/tower/conf.d/custom.py
```

```
# chown root:awx /etc/tower/conf.d/custom.py
```

2. Add the following to the **/etc/tower/conf.d/custom.py** file:

```
DEFAULT_CONTAINER_RUN_OPTIONS = ['--network',
'slirp4netns:enable_ipv6=true,cidr=192.0.2.0/24']
```

   - **192.0.2.0/24** is the value for the new CIDR in this example.

3. Stop and start the automation controller service in all controller and hybrid nodes:

```
# automation-controller-service stop
```

```
# automation-controller-service start
```

   All containers will start on the new CIDR.

CHAPTER 9. PLAYBOOKS

You can use automation content navigator to interactively troubleshoot your playbook. For more information about troubleshooting a playbook with automation content navigator, see Troubleshooting Ansible content with automation content navigator in the Automation Content Navigator Creator Guide.

CHAPTER 10. SUBSCRIPTIONS

For information about keeping your automation controller subscription in compliance, see Troubleshooting: Keep your subscription in compliance in the Automation Controller User Guide.