



Red Hat JBoss Web Server 5.7

Installing JBoss Web Server by using the Red Hat Ansible Certified Content Collection

Automating deployments of JBoss Web Server 5.7 with the Red Hat Ansible Certified Content Collection 1.2

Red Hat JBoss Web Server 5.7 Installing JBoss Web Server by using the Red Hat Ansible Certified Content Collection

Automating deployments of JBoss Web Server 5.7 with the Red Hat Ansible Certified Content Collection 1.2

Legal Notice

Copyright © 2022 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

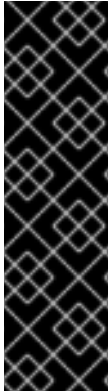
Install and use the Red Hat Ansible Certified Content Collection to automate deployments of JBoss Web Server.

Table of Contents

PREFACE	3
PROVIDING FEEDBACK ON RED HAT DOCUMENTATION	4
MAKING OPEN SOURCE MORE INCLUSIVE	5
CHAPTER 1. AUTOMATION OF JBOSS WEB SERVER DEPLOYMENTS BY USING AN ANSIBLE COLLECTION	6
1.1. ANSIBLE CONCEPTS AND BENEFITS	6
1.2. JBOSS WEB SERVER COLLECTION	6
CHAPTER 2. INSTALLING THE JBOSS WEB SERVER COLLECTION	7
CHAPTER 3. DEFINING VARIABLE SETTINGS	9
3.1. ENABLING THE AUTOMATED INSTALLATION OF JBOSS WEB SERVER ARCHIVE FILES	9
3.2. ENABLING THE AUTOMATED INSTALLATION OF JBOSS WEB SERVER PATCH UPDATES	11
3.3. ENSURING THAT A JDK IS INSTALLED ON THE TARGET HOSTS	13
3.4. ENSURING THAT A PRODUCT USER AND GROUP ARE CREATED ON THE TARGET HOSTS	14
3.5. ENABLING THE AUTOMATED INTEGRATION OF JBOSS WEB SERVER WITH SYSTEMD	15
3.6. ENABLEMENT OF AUTOMATED JBOSS WEB SERVER CONFIGURATION TASKS	16
3.6.1. Enabling the automated configuration of HTTPS support in JBoss Web Server	16
3.6.2. Enabling the automated configuration of mod_cluster support in JBoss Web Server	18
3.6.3. Enabling the automated configuration of the password vault in JBoss Web Server	19
3.6.4. Enabling the automated configuration of SELinux policies in JBoss Web Server	20
3.7. ENABLING THE AUTOMATED DEPLOYMENT OF JBOSS WEB SERVER APPLICATIONS ON YOUR TARGET HOSTS	21
CHAPTER 4. RUNNING THE PLAYBOOK	23
CHAPTER 5. VALIDATING THE DEPLOYMENT	24

PREFACE

The Red Hat Ansible Certified Content Collection for Red Hat JBoss Web Server is a prepackaged Ansible content collection that Red Hat provides. You can use the Red Hat Ansible Certified Content Collection to automate the installation and configuration of the Red Hat JBoss Web Server product. You can also add customized tasks to your playbook to automate the deployment of JBoss Web Server applications either at the same time as the automated product installation or later.



IMPORTANT

The Red Hat Ansible Certified Content Collection is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see [Technology Preview Features Support Scope](#).



NOTE

The rest of this document refers to the Red Hat Ansible Certified Content Collection for Red Hat JBoss Web Server as the *JBoss Web Server collection*.

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.

Thank you for the valuable feedback.

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. AUTOMATION OF JBOSS WEB SERVER DEPLOYMENTS BY USING AN ANSIBLE COLLECTION

Ansible is an information technology (IT) automation engine that you can use to automate and simplify cloud provisioning, configuration management, application deployment, intraservice orchestration, and other tasks across your IT enterprise.

An Ansible *collection* is a standardized distribution format for reusable Ansible content such as playbooks, roles, modules, and plug-ins. Red Hat provides a series of prepackaged Red Hat Ansible Certified Content Collections. You can use these Red Hat Ansible Certified Content Collections to enable the automated deployment of various Red Hat Runtimes products such as Red Hat JBoss Web Server in your system.

1.1. ANSIBLE CONCEPTS AND BENEFITS

Ansible includes various elements such as playbooks, roles, content collections, and automation execution environments. Using Ansible to automate IT processes, workflows, and infrastructure lifecycles provides multiple benefits to your enterprise. If you are unfamiliar with Ansible architecture or you want more information about the benefits of using Ansible, refer to the following **Additional resources** links.

Additional resources

- [How Ansible works](#)
- [Understanding Ansible concepts](#)
- [Learning Ansible basics](#)
- [Red Hat Ansible Automation Platform](#) product page

1.2. JBOSS WEB SERVER COLLECTION

For general information about the JBoss Web Server collection, refer to the [Ansible Collection - redhat.jws](#) page in [Red Hat Automation Hub](#). The [Ansible Collection - redhat.jws](#) page includes information about the roles that the collection contains. You can click the name of a role to view details about the purpose of this role, any requirements or dependencies, and the list of variables and default settings that the role uses to complete automation tasks.

Additional resources

- [What are Ansible content collections?](#)

CHAPTER 2. INSTALLING THE JBOSS WEB SERVER COLLECTION

As a first step toward automating deployments of Red Hat JBoss Web Server by using Ansible, you must download and install the JBoss Web Server collection from the [Red Hat Automation Hub](#). The JBoss Web Server collection is named **redhat.jws** in the [Red Hat Automation Hub](#). Before you install the JBoss Web Server collection, you must ensure that your system complies with certain prerequisites.

Prerequisites

- You have installed the **ansible-core** package version 2.12 or later on a control node in your system by installing Red Hat Ansible Automation Platform 2.x.
For more information about installing Red Hat Ansible Automation Platform, see the [Red Hat Ansible Automation Platform Installation Guide](#).
- You have updated the **ansible.cfg** file to use Red Hat automation hub as your *primary source* of Ansible collections. For more information about updating the **ansible.cfg** file, see [Getting started with automation hub](#).

Procedure

- On your Ansible control node, enter the following command:

```
$ ansible-galaxy collection install redhat.jws
```



NOTE

If the preceding command produces a **Failed to find collection redhat.jws:** error message, ensure that you have updated the **ansible.cfg** file correctly to use the Red Hat Automation Hub, as described in [Getting started with automation hub](#).

The preceding command produces the following output:

```
Starting galaxy collection install process
Process install dependency map
Starting collection install process
[...]
redhat.jws:1.2.x was installed successfully
[...]
```

In the preceding output, **1.2.x** represents the full version number of the installed collection (for example, **1.2.1**).

Verification

- On your Ansible control node, enter the following command:

```
$ ansible-galaxy collection list
```

The preceding command displays the list of installed collections. For example:

Collection	Version

redhat.jws	1.2.x

In the preceding example, **1.2.x** represents the full version number of the installed collection (for example, **1.2.1**).

Additional resources

- [Automation Hub: Connect to Hub](#)

CHAPTER 3. DEFINING VARIABLE SETTINGS

The JBoss Web Server collection provides a comprehensive set of variables and default values that you can manually update to suit your setup requirements. These variable settings provide all the information that the JBoss Web Server collection requires to complete an automated and customized installation of Red Hat JBoss Web Server on your target hosts.

For a full list of variables that the JBoss Web Server collection provides, see the [information page for the **jws** role](#) in the [Red Hat Automation Hub](#). The information page for the **jws** role lists the names, descriptions, and default values for all the variables that you can define.



NOTE

You can define variables in multiple ways. By default, the JBoss Web Server collection includes an example **playbook.yml** file that links to a **vars.yml** file in the same **/playbooks** folder. For illustrative purposes, the instructions in this section describe how to define variables in the **vars.yml** file that the collection provides. You can use a different way to define variables if you prefer.

You can define variables to automate the following tasks:

- [Install JBoss Web Server archive files](#).
- [Install JBoss Web Server patch updates](#).
- [Ensure that a supported JDK version is installed on your target hosts](#).
- [Ensure that a product user account and group are created on your target hosts](#).
- [Integrate JBoss Web Server with **systemd**](#).
- [Configure the JBoss Web Server installation](#).

You can also automate the deployment of web applications by adding customized tasks to the playbook, as described in [Enabling the automated deployment of JBoss Web Server applications on your target hosts](#).

3.1. ENABLING THE AUTOMATED INSTALLATION OF JBOSS WEB SERVER ARCHIVE FILES

By default, the JBoss Web Server collection is configured to install JBoss Web Server from product archive files. Before you use the JBoss Web Server collection to install Red Hat JBoss Web Server, you must obtain and copy the JBoss Web Server archive files to your Ansible control node. After you copy the archive files, you can set the **jws_version** variable to specify the product version that you want to install. If you decide to change the names of the archive files on your local host, you can also define variables to specify the appropriate file names.

After you obtain and set installation details for the product archive files, the JBoss Web Server collection automatically extracts these files and installs the product on your target hosts when you subsequently run the playbook.

**NOTE**

If the JBoss Web Server archive files are not already available on your system, you can download the archive files manually from the Red Hat Customer Portal, as described in [Step 1](#) of this procedure.

Prerequisites

- You have [installed the JBoss Web Server collection](#).

Procedure

1. If you want to download the JBoss Web Server archive files from the Red Hat Customer portal, perform the following steps:
 - a. On your Ansible control node, open a browser and log in to the [Product Downloads page in the Red Hat Customer Portal](#).
 - b. From the list of products, select **Red Hat JBoss Web Server**.
 - c. On the Software Downloads page, from the **Version** drop-down list, select the version of JBoss Web Server that you want to install.
 - d. On the **Releases** tab, click **Download** next to each archive file that you want to download. For more information about any specific product archive files that you must download, refer to the [Red Hat JBoss Web Server Installation Guide](#).

**NOTE**

You do not need to extract the product archive files. The JBoss Web Server collection automatically handles extracting the archive files.

2. To enable the automated installation of the product archive files, perform the following steps:
 - a. On your Ansible control node, open the **vars.yml** file.
 - b. To specify the JBoss Web Server version that you want to install, set the **jws_version** variable to the appropriate baseline version.
For example:

```
[...]
jws_version: 5.7.0
```

**NOTE**

Ensure that the value you specify for the **jws_version** variable matches the version of the product archive files that you want to install. For example, to install the baseline archive files for JBoss Web Server 5.7, specify a value of **5.7.0**.

- c. If you also want to install the native JBoss Web Server archive for the operating system that is on your target hosts, set the **jws_native** variable to **True**.
For example:

```
[...]
jws_native: True
```



NOTE

By default, the **jws_native** variable is set to **False**, which means the JBoss Web Server collection installs the main application server archive only.

If you set the **jws_native** variable to **True**, ensure that you have also obtained and copied the appropriate native archive file to your Ansible control node.

- d. If you changed the names of the copied archive files on your Ansible control node, set the **zipfile_name** and **jws_native_zipfile** variables to specify the files that you want to install. For example:

```
[...]
zipfile_name: <application_server_file>
jws_native_zipfile: <native_file>
```

In the preceding example, replace *<application_server_file>* and *<native_file>* with the appropriate archive file names.



NOTE

If you have not changed the file names, you do not need to define **zipfile_name** and **jws_native_zipfile** variables. The JBoss Web Server collection uses the value of the **jws_version** variable to determine the default file names automatically.

3. Save your changes to the **vars.yml** file.

3.2. ENABLING THE AUTOMATED INSTALLATION OF JBOSS WEB SERVER PATCH UPDATES

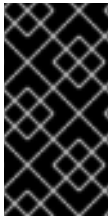
If product patch updates are available for the appropriate JBoss Web Server version, you can also use the JBoss Web Server collection to install the latest cumulative patches on your target hosts. You can use the same steps to enable the automated installation of patch updates regardless of whether you want to install the updates at the same time as the 5.7.0 version or later.

By default, the JBoss Web Server collection is configured to install JBoss Web Server patch updates from product archive files. First, you must obtain and copy the archive files for the latest patch updates to your Ansible control node. You must then set a **jws_apply_patches** variable to **True**. After you obtain and enable automated installation of the latest patch updates, the JBoss Web Server collection automatically extracts these files and installs the updates on your target hosts when you subsequently run the playbook.



NOTE

If copies of the archive files for the latest patch updates are not already available in your system, you can download the archive files manually from the Red Hat Customer Portal, as described in [Step 1](#) of this procedure.



IMPORTANT

You *cannot* use cumulative patch updates to install the baseline (X.X.0) version of a product. For example, installing a 5.7.2 patch installs versions 5.7.1 and 5.7.2 but cannot install the baseline 5.7.0 version. In this situation, you must ensure that the appropriate baseline version of the product is also installed either at the same time or previously.

Prerequisites

- You have [installed the JBoss Web Server collection](#).

Procedure

1. If you want to download the archive files for the latest JBoss Web Server patch updates from the Red Hat Customer Portal, perform the following steps:
 - a. On your Ansible control node, open a browser and log in to the [Product Downloads page in the Red Hat Customer Portal](#).
 - b. From the list of products, select **Red Hat JBoss Web Server**.
 - c. On the Software Downloads page, from the **Version** drop-down list, select the version of JBoss Web Server that you have installed.
 - d. Click the **Patches** tab and then click **Download** next to each archive file that you want to download for the latest patch update.
For more information about any specific product archive files that you must download, refer to the [Red Hat JBoss Web Server Installation Guide](#).



NOTE

Patch updates are cumulative. You do not need to download archive files for any previous patch updates that are not currently installed.

You do not need to extract the archive files. The JBoss Web Server collection automatically handles extracting the archive files.

2. To enable automated installation of the product patch updates, perform the following steps:
 - a. On your Ansible control node, open the **vars.yml** file.
 - b. Set the **jws_apply_patches** variable to **True**.
For example:

```
[...]
jws_version: 5.7.0
jws_native: True
[...]
jws_apply_patches: True
```


**NOTE**

Ensure that the **jws_version** variable is set to the appropriate baseline product version (for example, **5.7.0**).

If the **jws_native** variable is set to **True**, ensure that you have also obtained and copied the appropriate native archive files for the latest patch update to your Ansible control node.

Setting the **jws_apply_patches** variable to **True** enables the JBoss Web Server collection to install the most recent supported cumulative patch for the baseline product version that is specified in the **jws_version** variable. Based on the preceding example, the JBoss Web Server collection automatically installs the latest cumulative patch updates for the 5.7 release when you subsequently run the playbook.

3. Save your changes to the **vars.yml** file.

3.3. ENSURING THAT A JDK IS INSTALLED ON THE TARGET HOSTS

JBoss Web Server requires that a Java Development Kit (JDK) is already installed as a prerequisite on your target hosts to ensure that JBoss Web Server is successfully executed. A JDK includes a Java Runtime Environment (JRE) and Java Virtual Machine (JVM), which must be available on any host where you want to run JBoss Web Server. For a full list of JDK versions that JBoss Web Server supports, see [JBoss Web Server 5 Supported Configurations](#).

By default, the JBoss Web Server collection does not install a JDK automatically, based on the assumption that you have already installed a supported JDK on the target hosts. However, for the sake of convenience, you can configure the JBoss Web Server collection to install a supported version of Red Hat OpenJDK automatically on each target host.

Consider the following guidelines for installing a JDK when you use the JBoss Web Server collection:

- If you want to install a supported version of Red Hat OpenJDK on your target hosts, you can set the **jws_java_version** variable to the appropriate JDK version (for example, **1.8.0**, **11**, or **17**). The JBoss Web Server collection automatically installs the specified OpenJDK version on each target host when you subsequently run the playbook.
- If you want to install a supported version of IBM JDK or Oracle JDK, you must install the JDK manually on each target host or you can automate this process by using your own playbook. For more information about manually installing a version of IBM JDK or Oracle JDK, see the [Red Hat JBoss Web Server Installation Guide](#). In this situation, you do not need to set a variable.
- If you already have a supported JDK installed on your target hosts, you do not need to set a variable.

**NOTE**

Use the following procedure if you want to enable the JBoss Web Server collection to install Red Hat OpenJDK on target hosts where a supported JDK is not already installed.

Prerequisites

- You have [installed the JBoss Web Server collection](#).

Procedure

1. On your Ansible control node, open the **vars.yml** file.
2. Set the **jws_java_version** variable to the appropriate OpenJDK version that you want to install. For example:

```
[...]
jws_java_version: 1.8.0
```

Based on the preceding example, the JBoss Web Server collection automatically installs OpenJDK 8 on each target host when you run the playbook.



NOTE

Alternatively, if you want the JBoss Web Server collection to install Red Hat OpenJDK version 11 or 17, set the **jws_java_version** variable to a value of **11** or **17**, as appropriate.

3. Save your changes to the **vars.yml** file.

3.4. ENSURING THAT A PRODUCT USER AND GROUP ARE CREATED ON THE TARGET HOSTS

JBoss Web Server requires that a product user account and user group are already created as a prerequisite on your target hosts. By default, the JBoss Web Server collection handles this requirement by creating a **tomcat** user account and a **tomcat** group automatically on each target host. However, if you want the JBoss Web Server collection to create a different user account and group, you can modify the behavior of the JBoss Web Server collection to suit your setup requirements.

The product user account is also assigned ownership of the Tomcat directories to run the Tomcat service.



NOTE

Use the following procedure if you want to enable the JBoss Web Server collection to create a different user account and group rather than the **tomcat** default values.

Prerequisites

- You have [installed the JBoss Web Server collection](#).

Procedure

1. On your Ansible control node, open the **vars.yml** file.
2. Set the **jws_user** and **jws_group** variables to the appropriate product user name and group name that you want to create. For example:

```
[...]
jws_user: myuser
jws_group: myuser
```

Based on the preceding example, the JBoss Web Server collection automatically creates a **myuser** user account and group instead of creating the default **tomcat** user account and group.

3. Save your changes to the **vars.yml** file.

3.5. ENABLING THE AUTOMATED INTEGRATION OF JBOSS WEB SERVER WITH SYSTEMD

You can optionally enable the JBoss Web Server collection to set up JBoss Web Server as a service that a system daemon can manage. By default, the JBoss Web Server collection is not configured to integrate JBoss Web Server with a system daemon. If you enable this feature, the JBoss Web Server collection sets up JBoss Web Server as a **jws5-tomcat** service automatically on each target host. However, if you want to use a different service name, you can modify the behavior of the JBoss Web Server collection to suit your setup requirements.

When you integrate JBoss Web Server with a system daemon, the system daemon can automatically start the JBoss Web Server services at system startup. The system daemon also provides functions to start, stop, and check the status of the product. The default system daemon is **systemd**.



NOTE

This configuration task is optional but recommended.

Prerequisites

- You have [installed the JBoss Web Server collection](#).

Procedure

1. On your Ansible control node, open the **vars.yml** file.
2. To enable integration with **systemd**, set the **jws_systemd_enabled** variable to **True**.
For example:

```
[...]
jws_systemd_enabled: True
```

3. If you want JBoss Web Server to use a service name other than **jws5-tomcat**, set the **jws_service_name** variable to the appropriate value.
For example:

```
[...]
jws_service_name: jws
```

Based on the preceding example, the JBoss Web Server collection sets up the product as a **jws** service on each target host when you run the playbook.



NOTE

If you do not set the **jws_service_name** variable, the JBoss Web Server collection sets up the product as a **jws5-tomcat** service automatically.

4. Save your changes to the **vars.yml** file.

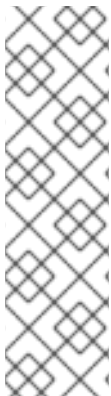
3.6. ENABLEMENT OF AUTOMATED JBOSS WEB SERVER CONFIGURATION TASKS

The JBoss Web Server collection provides a comprehensive set of variables to enable the automated configuration of a JBoss Web Server installation. By default, the JBoss Web Server collection configures JBoss Web Server to listen for nonsecure HTTP connections on port **8080**.

Other product features such as the following are disabled by default:

- Support for secure HTTPS connections
- **Mod_cluster** support for load-balancing HTTP server requests to the JBoss Web Server back end
- The password vault for storing sensitive data in an encrypted Java keystore
- Support for SELinux policies

To enable a wider set of product features, you can define variables to modify the behavior of the JBoss Web Server collection to suit your setup requirements.



NOTE

The following subsections describe only a subset of the automated configuration updates that the JBoss Web Server collection can perform. These example updates focus on enabling support for HTTPS connections, enabling **mod_cluster** support, enabling the password vault, and enabling SELinux policies.

For a full list of variables that the JBoss Web Server collection provides, refer to the [information page for the **jws** role](#) in [Red Hat automation hub](#). For more information about configuring and using JBoss Web Server features, refer to the [Red Hat JBoss Web Server documentation page](#).

3.6.1. Enabling the automated configuration of HTTPS support in JBoss Web Server

You can configure JBoss Web Server to support secure encrypted connections between web clients and the web server over the HTTPS protocol.

Consider the following guidelines for enabling HTTPS support when you use the JBoss Web Server collection:

- If you want to enable HTTPS support, you must ensure that a Java keystore exists on each target host before you subsequently run the playbook. The JBoss Web Server collection does not provide or create a Java keystore automatically. In this situation, you must create a new keystore on your target hosts or copy an existing keystore file to each target host, as described in [Step 1](#) of the following procedure.
- To enable HTTPS support, you can set a **jws_listen_https_enabled** variable to **True**.
- When you enable HTTPS support, the JBoss Web Server collection updates the **server.xml** file on each target host with the appropriate path and password settings for the Java keystore. By default, the JBoss Web Server collection configures these path and password settings in the **server.xml** file with values of **/etc/ssl/keystore.jks** and **changeit**, respectively. However, if you want to use a different keystore path or keystore password, you can modify the behavior of the JBoss Web Server collection to suit your setup requirements.

Prerequisites

- You have [installed the JBoss Web Server collection](#).

Procedure

1. If you want to create a Java keystore, perform the following steps:
 - a. Log in to the *target host* where you want to create the keystore.



NOTE

Ensure that a JDK is already installed and the **JAVA_HOME** variable is already set on the target host.

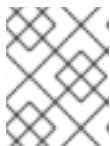
- b. To create the keystore, enter the following command:

```
$JAVA_HOME/bin/keytool -genkeypair -alias tomcat -keyalg RSA -keystore
<path_to_keystore>
```

In the preceding command, replace *<path_to_keystore>* with the full path to the keystore file that you want to create. If you do not specify the **-keystore** option, the command creates the keystore file in some default location that depends on the version of the JDK you have installed. For example, if you are using Red Hat OpenJDK, the default location for the keystore is **/etc/ssl/keystore.jks**.

The preceding command generates a keystore file that contains a pair of public and private keys and a single self-signed certificate for server authentication. The key pair and self-signed certificate are stored in a single keystore entry that is identified by the **-alias** option (for example, **tomcat**).

- c. When the **keytool** command prompts you for the following information, enter the appropriate values for your setup:
 - Keystore password (by default, **changeit**)
 - General information about the certificate
 - Key password for the certificate (by default, the keystore password)



NOTE

Alternatively, rather than create a new keystore, you can use the Linux **scp** command to copy an existing keystore file between different hosts.

2. To enable support for HTTPS connections, perform the following steps:
 - a. On your *Ansible control node*, open the **vars.yml** file.
 - b. Set the **jws_listen_https_enabled** variable to **True**.
For example:

```
[...]
jws_listen_https_enabled: True
```

- c. If the Java keystore on each target host is located in a path other than `/etc/ssl/keystore.jks`, set the `jws_listen_https_keystore_file` variable to the appropriate value.

For example:

```
[...]
jws_listen_https_keystore_file: <keystore_path>
```

In the preceding example, replace `<keystore_path>` with the full path to the keystore file that is on each target host.



NOTE

If you do not set the `jws_listen_https_keystore_file` variable, the JBoss Web Server collection automatically configures the `certificateKeystoreFile` setting in the `server.xml` file with a value of `/etc/ssl/keystore.jks`.

- d. If the Java keystore on each target host uses a password other than `changeit`, set the `jws_listen_https_keystore_password` variable to the appropriate value.

For example:

```
[...]
jws_listen_https_keystore_password: <keystore_password>
```

In the preceding example, replace `<keystore_password>` with the correct password for the Java keystore that is on each target host.



NOTE

If you do not set the `jws_listen_https_keystore_password` variable, the JBoss Web Server collection automatically configures the `certificateKeystorePassword` setting in the `server.xml` with a value of `changeit`.

- e. Save your changes to the `vars.yml` file.

3.6.2. Enabling the automated configuration of `mod_cluster` support in JBoss Web Server

The `mod_cluster` connector is a reduced-configuration and intelligent solution for load-balancing Apache HTTP Server requests to the JBoss Web Server back end. The `mod_cluster` connector also provides features such as real-time load-balancing calculations, application life-cycle control, automatic proxy discovery, and multiple protocol support. To enable `mod_cluster` support, you can define variables to enable the `mod_cluster` listener and specify IP address and port values for the `mod_cluster` instance.

Prerequisites

- You have [installed the JBoss Web Server collection](#).

Procedure

- On your Ansible control node, open the `vars.yml` file.

2. To enable the **mod_cluster** listener, set the **jws_modcluster_enabled** variable to **True**.
For example:

```
[...]
jws_modcluster_enabled: True
```

3. To specify the IP address and port of the **mod_cluster** instance, set the **jws_modcluster_ip** and **jws_modcluster_port** variables to the appropriate values. The default IP address is **127.0.0.1**. The default port is **6666**.

For example:

```
[...]
jws_modcluster_ip: <ip_address>
jws_modcluster_port: <port>
```

In the preceding example, replace *<ip_address>* with the appropriate bind address for the **mod_cluster** instance on the target host, and replace *<port>* with the appropriate port that the **mod_cluster** instance uses to listen for incoming requests.

4. Save your changes to the **vars.yml** file.

For more information about using **mod_cluster**, see the [HTTP Connectors and Load Balancing Guide](#).

3.6.3. Enabling the automated configuration of the password vault in JBoss Web Server

You can use the password vault for JBoss Web Server to mask passwords and other sensitive strings, and to store sensitive information in an encrypted Java keystore. When you use the password vault, you can stop storing clear-text passwords in your JBoss Web Server configuration files. JBoss Web Server can use the password vault to search for passwords and other sensitive strings from a keystore. To enable password vault, you can set a series of variables that enable you to specify various files and configuration settings that the password vault uses.

Prerequisites

- You have [installed the JBoss Web Server collection](#).
- You have created the required **vault.keystore**, **VAULT.dat**, and **vault.properties** files. For more information about creating these files, refer to the [Red Hat JBoss Web Server Installation Guide](#).

Procedure

1. On your Ansible control node, open the **vars.yml** file.
2. To specify the paths to the **vault.keystore**, **VAULT.dat**, and **vault.properties** files that you created as part of the prerequisite step, set the following variables to the appropriate values.
For example:

```
[...]
jws_vault_name: ./vault_files/vault.keystore
jws_vault_data: ./vault_files/VAULT.dat
jws_vault_properties: ./vault_files/vault.properties
```

In the preceding example, ensure that you specify the correct paths that you configured as part of the prerequisite step.

1. To enable the password vault feature, set the **jws_tomcat_vault_enabled** variable to **True**.
For example:

```
[...]
jws_tomcat_vault_enabled: True
```

2. To specify the keystore alias, keystore password, iteration count, and salt values that you configured for the password vault, set the following variables to the appropriate values.
For example:

```
[...]
jws_tomcat_vault_alias: <keystore_alias>
jws_tomcat_vault_storepass: <keystore_password>
jws_tomcat_vault_iteration: <iteration_count>
jws_tomcat_vault_salt: <salt>
```

In the preceding example, ensure that you specify the appropriate values that you configured as part of the prerequisite step.

3. Save your changes to the **vars.yml** file.

For more information about using the password vault, refer to the [Red Hat JBoss Web Server Installation Guide](#).

3.6.4. Enabling the automated configuration of SELinux policies in JBoss Web Server

You can use Security-Enhanced Linux (SELinux) policies to define access controls for JBoss Web Server. These policies are a set of rules that determine access rights to the product. To enable the use of SELinux policies, you can set a **jws_selinux_enabled** variable to **True**.

Prerequisites

- You have [installed the JBoss Web Server collection](#).
- You have set the **jws_native** variable to **True** to enable the automated installation of native JBoss Web Server archive files for the operating system that is on your target hosts. For more information, see [Enabling the automated installation of JBoss Web Server archive files](#).

Procedure

1. On your Ansible control node, open the **vars.yml** file.
2. Set the **jws_selinux_enabled** variable to **True**.
For example:

```
[...]
jws_selinux_enabled: True
```

3. Save your changes to the **vars.yml** file.

For more information about using SELinux policies in JBoss Web Server, refer to the [Red Hat JBoss Web Server Installation Guide](#).



WARNING

The core Ansible engine does not provide support for SELinux policies. If you want to enable SELinux policies, you must also install a community collection from Ansible Galaxy on your Ansible control node.

To install the community collection on your Ansible control node, enter the following command:

```
$ ansible-galaxy collection install community.general
```

3.7. ENABLING THE AUTOMATED DEPLOYMENT OF JBOSS WEB SERVER APPLICATIONS ON YOUR TARGET HOSTS

You can also automate the deployment of web applications on your target JBoss Web Server hosts by adding customized tasks to the playbook. This requires that you place the application **.war** file in the appropriate directory.

If you want to deploy a new or updated application when JBoss Web Server is already running, the JBoss Web Server collection provides a handler to restart the web server when the application is deployed.



NOTE

The following procedure assumes that you have created a custom playbook.

Prerequisites

- You have [installed the JBoss Web Server collection](#).

Procedure

1. On your Ansible control node, open your custom playbook.
2. In the **tasks:** section of the playbook, add a task to deploy the appropriate web application. For example:

```
[...]
tasks:
  [...]
  - name: "Deploy demo webapp"
    ansible.builtin.get_url:
      url: 'https://url_path/app_name.war'
      dest: "{{ jws_home }}/webapps/app_name.war"
    notify:
      - Restart Tomcat service
[...]
```

In the preceding example, replace `<url_path>` and `<app_name>` with the correct path and **.war** file name for the application that you want to deploy.

3. Save your changes to the playbook.

Additional resources

- [Files modules](#)
- [Net Tools modules](#)

CHAPTER 4. RUNNING THE PLAYBOOK

After you define variable settings, you can run the playbook to begin the automated installation process. You can run a playbook by using the [ansible-playbook](#) command on the control node or by using the [Red Hat Ansible automation controller](#). The JBoss Web Server collection then handles all installation and deployment tasks automatically.



NOTE

The following procedure assumes that you have created and updated a custom playbook.

Prerequisites

- You have [defined variable settings](#).
- Your playbook includes an appropriate link to the location where you have defined your variables.

For example:

```
---
- name: "Red Hat JBoss Web Server installation and configuration"
  hosts: all
  become: true
  vars_files:
    - <path_to_vars_file>/vars.yml
  [...]
```

The preceding example assumes that you have defined variables in a **vars.yml** file. Replace `<path_to_vars_file>` with the appropriate path.

- A supported operating system is already installed on your target hosts. For a full list of operating system versions that JBoss Web Server supports, refer to [JBoss Web Server 5 Supported Configurations](#).

Procedure

- Perform either of the following steps:
 - On your Ansible control node, enter the following command:

```
$ ansible-playbook <playbook_name>.yml
```

In the preceding command, replace `<playbook_name>` with the name you have assigned to your playbook.

- Use the Red Hat Ansible automation controller to run your playbook. For more information about getting started with the automation controller, see the [Red Hat Ansible Automation Platform](#) documentation page.

CHAPTER 5. VALIDATING THE DEPLOYMENT

After you successfully run the playbook, the JBoss Web Server collection automatically installs Red Hat JBoss Web Server on your target hosts. If you have added customized tasks to the playbook, Ansible also automatically deploys any JBoss Web Server applications on your targets hosts, as appropriate. You can optionally check the status of JBoss Web Server by using the **systemctl** command on the target host or by using the **curl** command on a remote host.

Prerequisites

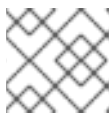
- You have [run the playbook](#).

Procedure

1. Optional: On the JBoss Web Server host, enter the following command as the root user:

```
# systemctl status <service_name>
```

In the preceding command, replace `<service_name>` with the correct service name for your JBoss Web Server installation. The default service name is **tomcat**. For more information about setting up a service name, see [Automating the integration of JBoss Web Server with systemd](#).



NOTE

This step requires that JBoss Web Server is integrated with **systemd**.

2. Optional: On a *remote host*, enter the following command as the root user:

```
# curl http://<target_host>:8080/
```

In the preceding command, replace `<target_host>` with the IP address or host name of the JBoss Web Server host that you want to access. The preceding command assumes that the JBoss Web Server is accessible through the default port **8080** and that the target firewall and network allow remote access to the port.



NOTE

The JBoss Web Server collection also includes a **validate.yml** playbook in the **/playbooks** folder. You can run the **validate.yml** playbook if you want the JBoss Web Server collection to perform automated validation checks. For more information about the **validate.yml** playbook, refer to the [information page for the jws_validation role](#) in the [Red Hat Automation Hub](#).

Additional resources

- [Controlling the JBoss Web Server with systemd](#)