Red Hat JBoss Web Server 3.1

Installation Guide

Install and Configure Red Hat JBoss Web Server 3.1

Last Updated: 2018-08-30

Red Hat JBoss Web Server 3.1 Installation Guide

Install and Configure Red Hat JBoss Web Server 3.1

Legal Notice

Copyright © 2018 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, 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 Software Collections 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

This book contains information related to installation and basic configuration of Red Hat JBoss Web Server.

Table of Contents

CHAPTER 1. INTRODUCTION

1.1. ABOUT RED HAT JBOSS WEB SERVER

Red Hat JBoss Web Server is a fully integrated and certified set of components for hosting Java web applications. It consists of a web server (Apache HTTP Server), an application server (Apache Tomcat Servlet container), load balancers (mod_jk and mod_cluster), and the Tomcat Native Library.

This Installation Guide includes procedures for the installation, minor upgrade, and basic configuration of the Tomcat servers from JBoss Web Server on supported operating systems. Installation and configuration instructions for the Apache HTTP Server are covered in the JBoss Core Services Documentation.

1.2. COMPONENTS

JBoss Web Server consists of the following components:

- **Apache Tomcat** is a servlet container in accordance with Java Servlet Specification. JBoss Web Server contains Apache Tomcat 7 and Apache Tomcat 8.
- **Apache Tomcat Native Library** is a Tomcat library, which improves Tomcat scalability, performance, and integration with native server technologies.
- **Apache Tomcat Connectors** (mod_jk, mod_cluster) are connectors between Apache HTTP Server and Apache Tomcat. Note that the default is mod_cluster, as it is the JBoss native load balancer and is more efficient, reliable, and scalable than mod_jk. As of JBoss Web Server 3.1, the mod_jk and mod_cluster connectors are provided as part of JBoss Core Services. For more information on installation and configuration of these two modules, see the HTTP Connectors and Load Balancing Guide.
- **Apache HTTP Server** is an open source web server developed by the Apache Software Foundation. The implementation follows the current HTTP standards. As of JBoss Web Server 3.1, the Apache HTTP Server is provided as part of JBoss Core Services. For more information on installation and configuration of the http server, see JBoss Core Services Documentation.
- **Hibernate** is an object-relational mapping framework. Hibernate included in JBoss Web Server contains Hibernate Core, Hibernate Annotations, and Hibernate EntityManager with JPA 1.0 APIs.

For a detailed list of component versions included in JBoss Web Server 3.1, see https://access.redhat.com/articles/111723.

NOTE

Tomcat clustering has been removed from JBoss Web Server 3.1. If you need clustering or session replication support for Java applications, Red Hat recommends that you use Red Hat JBoss Enterprise Application Platform (JBoss EAP).

1.3. SUPPORTED OPERATING SYSTEMS AND CONFIGURATIONS

For information on supported operating systems and configurations for JBoss Web Server, see https://access.redhat.com/articles/1377603.

1.4. INSTALLATION METHODS

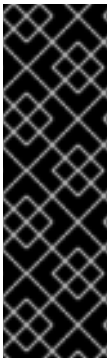
JBoss Web Server can be installed on supported Red Hat Enterprise Linux, Microsoft Windows, and Solaris systems using ZIP installation files available for each platform. JBoss Web Server can also be installed on supported Red Hat Enterprise Linux systems using RPM packages.

For ZIP installations, below is a summary of the components that are included in the ZIP file which form the core part of a JBoss Web Server installation.

- **`jws-application-servers-3.1.0-<platform>-<architecture>.zip`**
  - Tomcat 7
  - Tomcat 8
  - Platform-specific utilities

1.5. UPGRADING JBOSS WEB SERVER

Upgrading JBoss Web Server is not supported. To upgrade to JBoss Web Server 3.1 from an older version, a fresh installation is recommended.



IMPORTANT

If the RPM method was used to install JBoss Web Server 1.x or 2.x, it is not possible to remove JBoss Web Server 1.x or 2.x and install JBoss Web Server 3.1 on the same machine. Many configuration files from the old packages will be left on the system, and will create compatibility issues with the newer RPM packages. For the same reason, manually removing the old RPMs and installing the new ones is also not supported.

On a system where JBoss Web Server 1.x or 2.x was installed using the ZIP method, it is possible to remove the old version and install JBoss Web Server 3.1 on the same system.

1.6. COMPONENT DOCUMENTATION BUNDLE

JBoss Web Server includes an additional documentation bundle that includes the original vendor documentation for each component. This documentation bundle, **`jws-docs-3.1.0.zip`**, is available at the Red Hat Customer Portal, and contains additional documentation for the following:

- mod_cluster
- openssl
- tomcat7
- tomcat8
- tomcat-native

CHAPTER 2. INSTALLING JBOSS WEB SERVER ON RED HAT ENTERPRISE LINUX

You can install JBoss Web Server on Red Hat Enterprise Linux using one of two methods:

- ZIP files
- RPM packages

Regardless of which method you choose, you must first install a supported Java Development Kit (JDK).

2.1. PREREQUISITES

2.1.1. Installing a Java Development Kit (JDK)

Before installing JBoss Web Server, you must first install a supported Java Development Kit (JDK).

1. Subscribe your Red Hat Enterprise Linux system to the appropriate channel:
   - **OpenJDK:**
     - rhel-6-server-rpms
     - rhel-7-server-rpms
   - **Oracle:**
     - rhel-6-server-thirdparty-oracle-java-rpms
     - rhel-7-server-thirdparty-oracle-java-rpms
   - **IBM:**
     - rhel-6-server-supplementary-rpms
     - rhel-7-server-supplementary-rpms
2. As the root user, execute the command to install a 1.7 or 1.8 JDK:
   a. For JDK 1.7:

   ```
   # yum install java-1.7.0-<VENDOR>-devel
   ```

   Replace **<*VENDOR*>** with **`oracle`**, **`ibm`**, or **`openjdk`**.

   b. For JDK 1.8:

   ```
   # yum install java-1.8.0-<VENDOR>-devel
   ```

   Replace **<*VENDOR*>** with **`oracle`**, **`ibm`**, or **`openjdk`**.
3. Run the following commands as the root user to ensure the correct JDK is in use:

```
# alternatives --config java
```

```
# alternatives --config javac
```

These commands return lists of available JDK versions with the selected version marked with a plus (**+**) sign. If the selected JDK is not the desired one, change to the desired JDK as instructed in the shell prompt.

IMPORTANT

All software that use the **`java`** and **`javac`** commands uses the JDK set by **`alternatives`**. Changing Java alternatives may impact on the running of other software.

2.1.2. Red Hat Enterprise Linux Package Prerequisites

Before installing JBoss Web Server on Red Hat Enterprise Linux, ensure the following prerequisites are met.

- A supported JDK is installed.
- You must remove the **`tomcatjss`** package before installing the **`tomcat-native`** package. The **`tomcatjss`** package uses an underlying NSS security model rather than the OpenSSL security model.

Removing the tomcatjss Package

1. As the root user, run the following command to remove **`tomcatjss`**:

```
# yum remove tomcatjss
```

2.2. ZIP INSTALLATION

Ensure that all of the prerequisites are met before installing JBoss Web Server.

2.2.1. Downloading and Extracting JBoss Web Server

To install JBoss Web Server, download and extract the installation ZIP files.

Downloading JBoss Web Server

1. Open a browser and log in to the Red Hat Customer Portal.
2. Click **Downloads**.
3. Click **Red Hat JBoss Web Server** in the **Product Downloads** list.
4. Select the correct JBoss Web Server version from the **Version** drop-down menu.
5. Click **Download** for each of the following files, ensuring that you select the correct platform and architecture for your system:
   - Red Hat JBoss Web Server 3.1 Application Server (**`jws-application-servers-3.1.0-<platform>-<architecture>.zip`**)

Extracting JBoss Web Server

1. Unzip the downloaded ZIP files to your installation directory.
   The directory created by extracting the ZIP archives is the top-level directory for JBoss Web Server. This is referred to as ***JWS_HOME***.

2.2.2. Configuring the JBoss Web Server Installation

Some configuration is required before running JBoss Web Server. This section includes the following configuration procedures:

- Setting the JAVA_HOME Environment Variable
- Creating the tomcat user for simple and secure user management: Creating a Tomcat User.
- Enabling log4j Logging for Tomcat

Setting the JAVA_HOME Environment Variable

You must set the **JAVA_HOME** environment variable for Tomcat before running JBoss Web Server.

In the **bin** directory of Tomcat (either ***JWS_HOME*/tomcat7/bin** or ***JWS_HOME*/tomcat8/bin**), create a file named **setenv.sh**, and insert the **JAVA_HOME** path definition.

For example: **export JAVA_HOME=/usr/lib/jvm/jre-1.7.0-openjdk.x86_64**

Creating a Tomcat User

Follow this procedure to create the **tomcat** user and its parent group:

1. In a shell prompt as the root user, change directory to ***JWS_HOME***.
2. Run the following command to create the **tomcat** user group:

```
# groupadd -g 53 -r tomcat
```

3. Run the following command to create the **tomcat** user in the **tomcat** user group:

```
# useradd -c "Tomcat" -u 53 -g tomcat -s /bin/sh -r tomcat
```

4. Check that the **tomcat** user group and the **tomcat** user were created correctly:

```
$ grep tomcat /etc/group

tomcat:x:53

$ grep tomcat /etc/passwd

tomcat:x:53:53:Tomcat:/home/tomcat:/bin/sh
```

5. From ***JWS_HOME***, run the following command to assign the ownership of the Tomcat directories to the **tomcat** user to allow the user to run the Tomcat service:

```
# chown -R tomcat:tomcat tomcat<VERSION>
```

Replace **<*VERSION*>** with the respective Tomcat version number (**7** or **8**).

You can use **ls -l** to verify that the **tomcat** user is the owner of the directory.

6. Ensure that the **tomcat** user has execute permissions to all parent directories. For example:

```
# chmod -R u+X tomcat<VERSION>
```

Enabling Apache Log4j Logging for Tomcat

To enable logging with Apache Log4j:

1. Change directory to ***JWS_HOME*/extras/**:

```
# cd JWS_HOME/extras/
```

2. Copy **log4j-eap6.jar** and **log4j.properties** from ***JWS_HOME*/extras/** to ***JWS_HOME*/lib**.

```
# cp log4j.properties log4j-eap6.jar ../tomcat<VERSION>/lib/
```

Replace **<*VERSION*>** with the Tomcat version number (**7** or **8**).

3. Change directory to ***JWS_HOME*/tomcat<VERSION>/extras/**:

```
# cd ../tomcat<VERSION>/extras/
```

4. Copy **tomcat-juli-adapters.jar** from ***JWS_HOME*/tomcat<VERSION>/extras** to ***JWS_HOME*/tomcat<VERSION>/lib**.

```
# cp tomcat-juli-adapters.jar ../lib/
```

5. Replace ***JWS_HOME*/tomcat<VERSION>/bin/tomcat-juli.jar** with ***JWS_HOME*/tomcat<VERSION>/extras/tomcat-juli.jar**:

```
# cp tomcat-juli.jar ../bin/
```

2.2.3. Starting JBoss Web Server

To start JBoss Web Server, you must start the following:

- Tomcat (7 or 8).

Before starting Tomcat, ensure that the following prerequisites are met:

- The Tomcat user is created.
- JAVA_HOME is set correctly.

Starting Tomcat

1. Run the following command as the **tomcat** user with your respective Tomcat version (**7** or **8**):

```
$ sh JWS_HOME/tomcat<VERSION>/bin/startup.sh
```

IMPORTANT

Although there are multiple methods of starting Tomcat, it is recommended that you use the **`startup.sh`** script. To start Tomcat as a service using Jsvc, see the Jsvc chapter.

2.2.4. Stopping JBoss Web Server

To stop JBoss Web Server, you must stop the following:

- Tomcat (7 or 8).

Stopping Tomcat

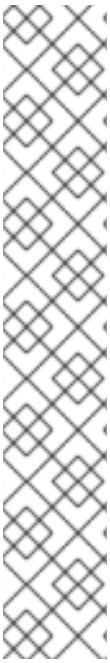
1. To stop Tomcat, run the following command as the root user with your respective Tomcat version (**7** or **8**):

```
# sh JWS_HOME/tomcat<VERSION>/bin/shutdown.sh
```

2.3. RPM INSTALLATION

Installing JBoss Web Server from RPM packages installs Tomcat as service, and installs its resources into absolute paths. The RPM installation option is only available for Red Hat Enterprise Linux 6 and Red Hat Enterprise Linux 7.

RPM installation packages for JBoss Web Server are available from Red Hat Subscription Management.



NOTE

For users wanting to manage JBoss Web Server installations using Red Hat Satellite: although Red Hat Satellite 6 is recommended for managing JBoss Web Server 3.1 installations, the following Red Hat Network (RHN) channels are also provided specifically for Satellite 5 users:

- For Red Hat Enterprise Linux 6:
  - jws-3-x86_64-server-6-rpm
  - jws-3-i386-server-6-rpm
- For Red Hat Enterprise Linux 7:
  - jws-3-x86_64-server-7-rpm

Red Hat Satellite 6 users can use the Red Hat Content Delivery Network (CDN) repositories.

WARNING

Installing JBoss Web Server from RPM packages shares Java library files with other applications. Library version conflicts occur when using RPM packages to install both JBoss Web Server 3 and JBoss EAP 6 on the same machine. To workaround the issue, you can install either JBoss Web Server 3 or JBoss EAP 6 using the RPM installation method, and the other using the ZIP installation method.

2.3.1. Installing JBoss Web Server from RPM packages

Prerequisites

- Install a Java Development Kit (JDK).
- Ensure that the tomcatjss package is removed.

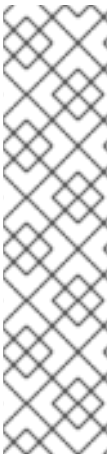
Before downloading and installing the RPM packages, you must register your system with Red Hat Subscription Management and subscribe to the respective Content Delivery Network (CDN) repositories.

For information on registering Red Hat Enterprise Linux, see Configuring the Subscription Service for Red Hat Enterprise Linux 6 or The Subscription Manager for Red Hat Enterprise Linux 7.

Attaching subscriptions to Red Hat Enterprise Linux (if required)

If the system does not have a subscription attached that provides JBoss Web Server:

1. Log in to the Red Hat Subscription Manager.
2. Click on the **Systems** tab.
3. Click on the **`Name`** of the system to add the subscription to.
4. Change from the **Details** tab to the **Subscriptions** tab, then click **`Attach Subscriptions`**.
5. Select the check box beside the subscription to attach, then click **`Attach Subscriptions`**.



NOTE

To verify that a subscription provides the required CDN repositories:

1. Log in to: https://access.redhat.com/management/subscriptions.
2. Click the **`Subscription Name`**.
3. Under **Products Provided**, you require:
   - JBoss Enterprise Web Server.
   - Red Hat JBoss Core Services.

Installing JBoss Web Server from RPM packages using YUM

1. On a command line, subscribe to the JBoss Web Server CDN repositories for your operating system version using **`subscription-manager`**:

```
# subscription-manager repos --enable <repository>
```

   - For Red Hat Enterprise Linux 6:
     - jws-3-for-rhel-6-server-rpms
     - jb-coreservices-1-for-rhel-6-server-rpms
   - For Red Hat Enterprise Linux 7:
     - jws-3-for-rhel-7-server-rpms
     - jb-coreservices-1-for-rhel-7-server-rpms

2. Issue the following command as the root user to install JBoss Web Server:

```
# yum groupinstall jws3
```

NOTE

- Although not recommended, instead of using the group install, you can install each of the packages and their dependencies individually.
- The Red Hat JBoss Core Services repositories above are required for the installation of JBoss Web Server.

2.3.2. Installing the JBoss Web Server Plus Group

The JBoss Web Server Plus group contains additional packages, mainly for the addition of Hibernate and its dependencies.

To install the JBoss Web Server Plus group of packages, run the following command as the root user:

```
# yum groupinstall jws3plus
```

2.3.3. Starting JBoss Web Server

To start JBoss Web Server, you must start the following:

- Tomcat (7 or 8)

Starting Tomcat

1. In a shell prompt as the root user, start the Tomcat service. Replace ***<VERSION>*** with the desired Tomcat version (**7** or **8**):

   a. For Red Hat Enterprise Linux 6:

```
# service tomcat<VERSION> start
```

   b. For Red Hat Enterprise Linux 7:

```
# systemctl start tomcat<VERSION>.service
```

This is the only supported method of starting Tomcat for an RPM installation.

1. To verify that Tomcat is running, the output of the service **`status`** command should be reviewed. This can be executed as any user.
   a. For Red Hat Enterprise Linux 6:

   ```
   # service tomcat<VERSION> status
   ```

   b. For Red Hat Enterprise Linux 7:

   ```
   # systemctl status tomcat<VERSION>.service
   ```

2.3.4. Stopping JBoss Web Server

To stop JBoss Web Server, stop the Tomcat services.

Stopping Tomcat

1. In a shell prompt as the root user, stop the Tomcat service. Replace ***<VERSION>*** with the desired Tomcat version (**7** or **8**):
   a. For Red Hat Enterprise Linux 6:

   ```
   # service tomcat<VERSION> stop
   ```

   b. For Red Hat Enterprise Linux 7:

   ```
   # systemctl stop tomcat<VERSION>.service
   ```

2. To verify that Tomcat is no longer running, the output of the service **`status`** command should be reviewed. This can be executed as any user.
   a. For Red Hat Enterprise Linux 6:

   ```
   # service tomcat<VERSION> status
   ```

   b. For Red Hat Enterprise Linux 7:

   ```
   # systemctl status tomcat<VERSION>.service
   ```

2.3.5. Configuring JBoss Web Server Services to Start at Boot

You can configure JBoss Web Server to start at boot.

Use the following commands to enable the JBoss Web Server services to start at boot. Replace ***<VERSION>*** with the desired Tomcat version (**7** or **8**).

- For Red Hat Enterprise Linux 6:

```
# chkconfig tomcat<VERSION> on
```

- For Red Hat Enterprise Linux 7:

```
# systemctl enable tomcat<VERSION>.service
```

2.4. SELINUX POLICIES

2.4.1. SELinux Policy Information

The following table contains information about the SELinux policies provided in the tomcat*<version>*-selinux packages.

Table 2.1. RPMs and Default SELinux Policies

Name	Port Information	Policy Information
tomcat*<version>*	Four ports in **`http_port_t`** (TCP ports **`8080`**, **`8005`**, **`8009`**, and **`8443`**) to allow the tomcat process to use them.	The Tomcat**`<VERSION>`** policy is installed, which sets the appropriate SELinux domain for the process when Tomcat executes. It also sets the appropriate contexts to allow tomcat to write to **`/var/lib/tomcat<VERSION>`**, **`/var/log/tomcat<VERSION>`**, **`/var/cache/tomcat<VERSION>`**, and **`/var/run/tomcat<VERSION>.pid`**.

For more information about using SELinux and other Red Hat Enterprise Linux security information, see the *Red Hat Enterprise Linux Security Guide*.

2.4.2. SELinux Policies for an RPM Installation

SELinux policies for each Tomcat are provided via their own Tomcat sub-packages: **`tomcat7-selinux`** and **`tomcat8-selinux`**. These packages are available in the JWS channel.

- To enable SELinux policies on Tomcat 7, install the **`tomcat7-selinux`** package.
- To enable SELinux policies on Tomcat 8, install the **`tomcat8-selinux`** package.

2.4.3. SELinux Policies for a ZIP Installation

In this release, SELinux policies are provided in the ZIP packages. The SELinux security model is enforced by the kernel and ensures applications have limited access to resources such as file system locations and ports. This helps ensure that the errant processes (either compromised or poorly configured) are restricted and in some cases prevented from running. The **`.postinstall.selinux`** file is included in each **`tomcat`** folder. If required, you can run the **`.postinstall.selinux`** script.

To install the SELinux policies using ZIP:

1. Install the **selinux-policy-devel** package:

```
yum install -y selinux-policy-devel
```

2. Execute the **.postinstall.selinux** script:

```
cd <JWS_home>/tomcat<version>
sh .postinstall.selinux
```

Where **tomcat<*version*>** refers to **tomcat7** or **tomcat8**.

3. Make and install the SELinux module:

```
cd selinux
make -f /usr/share/selinux/devel/Makefile
semodule -i tomcat<version>.pp
```

4. Apply the SELinux contexts for JBoss Web Server:

```
restorecon -r <JWS_home>/tomcat<version>/
```

5. Add access permissions to the required ports for JBoss Web Server:

```
semanage port -a -t http_port_t -p tcp 8005
semanage port -a -t http_port_t -p tcp 8080
semanage port -a -t http_port_t -p tcp 8009
semanage port -a -t http_port_t -p tcp 8443
```

6. Start the Tomcat service:

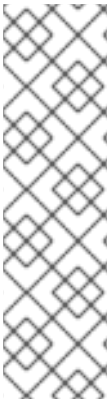
```
<JWS_home>/bin/startup.sh
```

7. Check the context of the running process expecting **tomcat_<version>__t**:

```
ps -eZ | grep tomcat | head -n1
```

8. To verify the contexts of the Tomcat directories, for example:

```
ls -lZ <JWS_home>/tomcat<version>/logs/
```



NOTE

By default, the selinux policy provided is not active and the Tomcat processes run in the **unconfined_java_t** domain. This domain does not confine the processes, and it is recommended that you undertake the following security precautions if you chose not to enable the selinux policy provided:

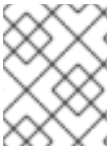
- Restrict file access for the **tomcat** user to only the files and directories that are necessary to the JBoss Web Server runtime.
- Do not run Tomcat as the **root** user.

CHAPTER 3. INSTALLING JBOSS WEB SERVER ON MICROSOFT WINDOWS

3.1. INSTALLING A JAVA DEVELOPMENT KIT (JDK)

Before installing JBoss Web Server on Microsoft Windows, you must first install a supported Java Development Kit (JDK).

For a list of supported configurations, see the Red Hat Customer Portal article: JBoss Web Server 3 Supported Configurations.



NOTE

For instructions on installing the IBM JDK, visit: https://www.ibm.com/developerworks/java/jdk/

To install the Oracle Java Development Kit:

1. Download the Oracle JDK 1.7 or 1.8 for your operating system and architecture. You can download the JDK installation file from the Oracle website: http://www.oracle.com/technetwork/java/javase/downloads/index.html.
2. Double-click the downloaded file to start the installation.
3. Proceed as instructed in the installation window.

3.2. DOWNLOADING AND EXTRACTING JBOSS WEB SERVER

To install JBoss Web Server, download and extract the installation ZIP files.

Downloading JBoss Web Server

1. Open a browser and log in to the Red Hat Customer Portal.
2. Click **Downloads**.
3. Click **Red Hat JBoss Web Server** in the **Product Downloads** list.
4. Select the correct JBoss Web Server version from the **Version** drop-down menu.
5. Click **Download** for each of the following files, ensuring that you select the correct platform and architecture for your system:
   - Red Hat JBoss Web Server 3.1 Application Server (**`jws-application-servers-3.1.0-<platform>-<architecture>.zip`**)

Extracting JBoss Web Server

1. Unzip the downloaded ZIP files to your installation directory.
   The directory created by extracting the ZIP archives is the top-level directory for JBoss Web Server. This is referred to as ***`JWS_HOME`***.

3.3. CONFIGURING THE JBOSS WEB SERVER INSTALLATION

Some configuration is required before running JBoss Web Server. This section includes the following configuration procedures:

- Setting Environment Variables
- Running the Post-Installation Scripts
- Installing the Tomcat Service
- Enabling log4j Logging for Tomcat
- Configuring Folder Permissions for the JBoss Web Server Services

Setting Environment Variables

1. Log in to an account with local administrator permissions.
2. Go to **Control Panel** → **System**.
3. Click on the **Advanced** tab.
4. Click the **Environment Variables** button.
5. Click the **New** button for **System Variables**.
6. For **`JAVA_HOME`**, **`TMP`**, and **`TEMP`**, enter the appropriate name-value pairs for your system.
7. For the SSL Connector to work, you will also need to add ***`JWS_HOME`*`\bin`** to the **`PATH`** environment variable of the user that the services will run under. This user is **`SYSTEM`** by default.

Running the Post-Installation Scripts

1. Open a command prompt with administrator privileges and change to the **`etc`** folder of your JBoss Web Server installation:

   ```
   cd /D "JWS_HOME\etc"
   ```

2. Run the Tomcat post-installation script using the following command:

   ```
   call postinstall.tomcat.bat
   ```

   The script creates the required symbolic links (junction points) for temporary logging and configuration directories.

Installing the Tomcat Service

1. Open a command prompt with administrator privileges and change to the **`bin`** folder for your Tomcat version:

   ```
   cd /D "JWS_HOME\share\tomcat<VERSION>\bin"
   ```

2. Install the Tomcat service with the following command:

```
call service.bat install
```

Enabling log4j Logging for Tomcat

1. Open a command prompt with administrator privileges and change to ***JWS_HOME*\share\extras**.
2. Copy the log4j files to the **lib** folder for your Tomcat version:

```
copy log4j-eap6.jar log4j.properties tomcat-juli-adapters.jar
..\tomcat<VERSION>\lib
```

3. Replace **tomcat-juli.jar** file in your Tomcat **bin** directory with the **tomcat-juli.jar** file from ***JWS_HOME*\share\extras**:

```
copy tomcat-juli.jar ..\tomcat<VERSION>\bin
```

Configuring Folder Permissions for the JBoss Web Server Services

Follow this procedure to ensure that the account used to run the services has full control over the ***JWS_HOME*** folder and all of its subfolders:

1. Right-click the ***JWS_HOME*** folder and click **Properties**.
2. Select the **Security** tab.
3. Click the **Edit** button.
4. Click the **Add** button.
5. In the text box, enter **LOCAL SERVICE**.
6. Select the **Full Control** check box for the **LOCAL SERVICE** account.
7. Click **OK**.
8. Click the **Advanced** button.
9. Inside the **Advanced Security Settings** dialog, select **LOCAL SERVICE** and click **Edit**.
10. Select the check box next to the **Replace all existing inheritable permissions on all descendants with inheritable permissions from this object** option.
11. Click **OK** through all the open folder property windows to apply the settings.

3.4. STARTING JBOSS WEB SERVER

To start JBoss Web Server, you must start the following:

- Tomcat (7 or 8)

You can start the services from a command prompt, or with the Computer Management tool.

Starting JBoss Web Server from a Command Prompt

1. Open a command prompt with administrator privileges.
2. Start the Tomcat service:

```
net start tomcat<VERSION>
```

Starting JBoss Web Server from the Computer Management Tool

1. Go to **Start** → **Administrative Tools** → **Services**.
2. In the **Services** list, right-click the name of the service (**tomcat**) and click **Start**.

NOTE

Some third-party applications add libraries to the system directory in Windows. These take precedence over Tomcat libraries when looked-up. This means that if those third-party libraries have the same name as the those used by Tomcat native libraries, they are loaded instead of the libraries distributed with JBoss Web Server.

In this situation, Tomcat may not start, and does not log any error messages in the Windows Event Log, or Tomcat log files. Errors can only be seen by using **catalina.bat run**.

If this behavior occurs, inspect the contents of the **C:\windows\System32** directory and other **PATH** directories, and ensure that there are no DLLs conflicting with those delivered with JBoss Web Server. In particular, look for **libeay32.dll**, **ssleay32.dll**, and **libssl32.dll**.

3.5. STOPPING JBOSS WEB SERVER

To stop JBoss Web Server, you must stop the following:

- Tomcat

You can stop the services from a command prompt, or with the Computer Management tool.

Stopping JBoss Web Server from a Command Prompt

1. Open a command prompt with administrator privileges.
2. Stop the Tomcat service:

```
net stop tomcat<VERSION>
```

Stopping JBoss Web Server from the Computer Management Tool

1. Go to **Start** → **Administrative Tools** → **Services**.
2. In the **Services** list, right-click the name of the service (**tomcat**) and click **Stop**.

CHAPTER 4. INSTALLING JBOSS WEB SERVER ON SOLARIS

4.1. INSTALLING A JAVA DEVELOPMENT KIT (JDK)

Before installing JBoss Web Server on Solaris, you must first install a supported Java Development Kit (JDK).

For a list of supported configurations, see the Red Hat Customer Portal article: JBoss Web Server 3 Supported Configurations.

Installing a Java Development Kit (JDK)

Install the Oracle JDK on a command line as the root user:

```
# pkg install jdk-<version>
```

Where <version> is the version of the JDK to install, such as **`jdk-8`**

Alternative: Download and Install a Java Development Kit on Solaris

1. Download the Oracle JDK for your operating system and architecture. You can download the JDK installation file from the Oracle website: http://www.oracle.com/technetwork/java/javase/downloads/index.html.
2. Run the JDK installation file.
3. Open **`/usr`** at a shell prompt, and run the following command to display the current Java symbolic link:

   ```
   ls -lad java
   ```

4. Remove the link:

   ```
   rm java
   ```

5. Create a new Java symbolic link to the newly installed JDK:

   ```
   ln -sf /usr/jdk/<JDK>
   ```

4.2. DOWNLOADING AND EXTRACTING JBOSS WEB SERVER

To install JBoss Web Server, download and extract the installation ZIP files.

Downloading JBoss Web Server

1. Open a browser and log in to the Red Hat Customer Portal.
2. Click **Downloads**.
3. Click **Red Hat JBoss Web Server** in the **Product Downloads** list.
4. Select the correct JBoss Web Server version from the **Version** drop-down menu.

5. Click **Download** for each of the following files, ensuring that you select the correct platform and architecture for your system:
   - Red Hat JBoss Web Server 3.1 Application Server (**jws-application-servers-3.1.0-*<platform>*-*<architecture>*.zip**)

Extracting JBoss Web Server

1. Unzip the downloaded ZIP files to your installation directory.
   The directory created by extracting the ZIP archives is the top-level directory for JBoss Web Server. This is referred to as ***JWS_HOME***.

4.3. CONFIGURING THE JBOSS WEB SERVER INSTALLATION

Some configuration is required before running JBoss Web Server. This section includes the following configuration procedures:

- Running the Post-Installation Scripts
- Setting the JAVA_HOME Environment Variable
- Enabling log4j Logging for Tomcat

Running the Post-Installation Scripts

1. Open a shell prompt, and change directory to ***JWS_HOME*/etc**.
2. As the root user, run the post-installation scripts:

   ```
   # sh .postinstall.tomcat
   ```

   The post-installation script creates the Tomcat user and group.

Setting the JAVA_HOME Environment Variable

You must set the **JAVA_HOME** environment variable for Tomcat before running JBoss Web Server.

1. Open the Tomcat configuration file:
   - For Tomcat 7: ***JWS_HOME*/etc/sysconfig/tomcat7**
   - For Tomcat 8: ***JWS_HOME*/etc/sysconfig/tomcat8**
2. Remove the hash (#) at the beginning of the following line:

   ```
   # JAVA_HOME="/usr/java"
   ```

Enabling log4j Logging for Tomcat

If required, follow this procedure to add log4j logging to Tomcat.

1. Open a shell prompt and change directory to ***JWS_HOME*/share/extras/**.
2. Copy the **log4j-eap6.jar**, **log4j.properties**, and **tomcat-juli-adapters.jar** files to the **lib** directory of the Tomcat directory.

For example:

```
# cp log4j.properties ../tomcat<VERSION>/lib/
```

```
# cp log4j-eap6.jar ../tomcat<VERSION>/lib/
```

```
# cp tomcat-juli-adapters.jar ../tomcat<VERSION>/lib/
```

Replace **<VERSION>** with the respective Tomcat version number (**7** or **8**).

3. Replace **tomcat-juli.jar** file in your Tomcat **bin** directory with the **tomcat-juli.jar** file from ***JWS_HOME*/share/extras/**:

```
# cp tomcat-juli.jar ../tomcat<VERSION>/bin/
```

4.4. STARTING JBOSS WEB SERVER

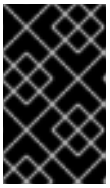
To start JBoss Web Server, you must start the following:

- Tomcat (7 or 8)

Starting Tomcat

1. As the root user, run the following command with your respective Tomcat version (**7** or **8**):

```
# sh JWS_HOME/share/apache-tomcat-<VERSION>/bin/daemon.sh start
```



IMPORTANT

Although there are multiple methods of starting Tomcat, it is recommended that you use the **daemon.sh** script. To start Tomcat as a service using Jsvc, see the Jsvc chapter.

4.5. STOPPING JBOSS WEB SERVER

To stop JBoss Web Server, you must stop the following:

- Tomcat (7 or 8)

Stopping Tomcat

1. To stop Tomcat, in a shell prompt as the root user, run the following command with your respective Tomcat version (**7** or **8**):

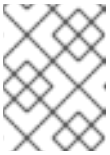
```
# sh JWS_HOME/share/apache-tomcat-<VERSION>/bin/daemon.sh stop
```

CHAPTER 5. USING JSVC TO START TOMCAT

Jsvc is a set of libraries and applications that facilitates running Java applications on Linux, UNIX, and similar operating systems. Using Jsvc with JBoss Web Server allows Tomcat to switch identities. Using Jsvc, Tomcat can perform root-level operations and then revert to a non-privileged user. Jsvc is primarily used for running Tomcat as a service.

Jsvc files are available at the following locations:

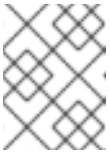
- For Red Hat Enterprise Linux:
  - ***JWS_HOME*/extras/jsvc**
  - ***JWS_HOME*/tomcat<*VERSION*>/bin/jsvc**



NOTE

***JWS_HOME*/tomcat<*VERSION*>/bin/jsvc** is a symlink to ***JWS_HOME*/extras/jsvc**.

- For Solaris:
  - ***JWS_HOME*/sbin/jsvc**
  - ***JWS_HOME*/share/apache-tomcat-<*VERSION*>/bin/jsvc**



NOTE

***JWS_HOME*/share/apache-tomcat-<*VERSION*>/bin/jsvc** is a symlink to ***JWS_HOME*/sbin/jsvc**.

5.1. STARTING TOMCAT USING JSVC

Start Tomcat Using Jsvc on Red Hat Enterprise Linux

Run the following command to start Tomcat using Jsvc on Red Hat Enterprise Linux:

```
JWS_HOME/tomcat<VERSION>/bin/daemon.sh start
```

Start Tomcat Using Jsvc on Solaris

Run the following command to start Tomcat using Jsvc on Solaris:

```
JWS_HOME/share/tomcat<VERSION>/bin/daemon.sh start
```

5.2. STOPPING TOMCAT USING JSVC

Stop Tomcat Using Jsvc on Red Hat Enterprise Linux

Run the following command to stop Tomcat that was started using Jsvc on Red Hat Enterprise Linux:

```
JWS_HOME/tomcat<VERSION>/bin/daemon.sh stop
```

Stop Tomcat Using Jsvc on Solaris

Run the following command to stop Tomcat that was started using Jsvc on Solaris:

```
JWS_HOME/share/tomcat<VERSION>/bin/daemon.sh stop
```

5.3. JSVC PARAMETERS

The following parameters can be configured when running the **daemon.sh** script:

Table 5.1. daemon.sh Startup Parameters

Parameter Name	Environment Variable	Default Value	Description
--java-home	**JAVA_HOME**	Based on the value of the **PATH** variable.	The Java home directory location.
--catalina-home	**CATALINA_HOME**	Determined by the location of the script.	The Tomcat installation directory location.
--catalina-base	**CATALINA_BASE**	Based on the value of the **PATH** variable.	The directory that contains the specific configuration and setup information if multiple servers are using the same installation.
--catalina-pid	-	**$CATALINA_BASE/logs/catalina-daemon.pid**	The file where the process ID (PID) for the running instance of Tomcat is stored.
--tomcat-user	-	**tomcat**	The user Tomcat uses.
--service-start-wait-time	-		This is a wrapper to the **--wait** parameter. The **--wait** parameter accepts values in seconds.

CHAPTER 6. HIBERNATE ON JBOSS WEB SERVER

Hibernate is an object-relational mapping framework. It is packaged independently from JBoss Web Server. This packaged version is used on all supported platforms.

Hibernate is used in the same way it is used for a regular Tomcat installation: the Hibernate JAR files are added into a deployment WAR file. Tomcat provides a default connection pooling mechanism, which is defined in **`context.xml`**. However, **`persistence.xml`** and **`web.xml`** are also required. The example below shows a configuration with the Tomcat connection pooling mechanism.

- **`/META-INF/context.xml`** defines the connection pools Tomcat should create.

  context.xml

  ```
  <Context>
    <Resource
      name="jdbc/DsWebAppDB"
      auth="Container"
      type="javax.sql.DataSource"
      username="sa"
      password=""
      driverClassName="org.h2.Driver"
      url="jdbc:h2:mem:target/test/db/h2/hibernate"
      maxActive="8"
      maxIdle="4"/>
  </Context>
  ```

- **`/WEB-INF/classes/META-INF/persistence.xml`** is a JPA configuration file. It defines how the application configures Hibernate to consume connections from the Tomcat pool. If you are using the Hibernate API directly, use a similar configuration to that shown in **`hibernate.cfg.xml`**.

  persistence.xml

  ```
  <persistence version="1.0"
    xmlns="http://java.sun.com/xml/ns/persistence"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/persistence
  http://java.sun.com/xml/ns/persistence/persistence_1_0.xsd">

    <persistence-unit name="dswebapp">
      <provider>org.hibernate.ejb.HibernatePersistence</provider>
      <properties>
        <property name="hibernate.dialect"
  value="org.hibernate.dialect.H2Dialect" />
        <property name="hibernate.connection.datasource"
  value="java:comp/env/jdbc/DsWebAppDB"/>
      </properties>
    </persistence-unit>
  </persistence>
  ```

- **`/WEB-INF/web.xml`** is a regular web application deployment file, which instructs Tomcat which datasource to consume. In the example below, the datasource is **`jdbc/DsWebAppDB`**.

  web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="2.5" xmlns="http://java.sun.com/xml/ns/javaee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
  http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd">

 <resource-env-ref>
  <resource-env-ref-name>jdbc/DsWebAppDB</resource-env-ref-name>
  <resource-env-ref-type>javax.sql.DataSource</resource-env-ref-
type>
 </resource-env-ref>
</web-app>
```

For details, see the Hibernate documentation for JBoss Web Server.

CHAPTER 7. MONITORING JBOSS WEB SERVER WITH JBOSS OPERATIONS NETWORK (ON)

JBoss Web Server can be monitored and managed by Red Hat JBoss Operations Network (JBoss ON).

To monitor JBoss Web Server using JBoss ON, you must install the *Web Server Plugin Pack* on your JBoss ON server, and then configure Tomcat in JBoss Web Server to be monitored.

7.1. DOWNLOADING THE WEB SERVER PLUGIN PACK FOR JBOSS ON

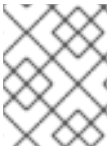
To monitor JBoss Web Server using JBoss ON, you must first download and install the *Web Server Plugin Pack* on your JBoss ON server.

1. Open a web browser, and log in to the Red Hat Customer Portal: http://access.redhat.com.
2. Click **Downloads**.
3. Click **Red Hat JBoss Web Server** in the **Product Downloads** list.
4. Select **JBoss ON for Web Server** in the **Product** drop-down menu.
5. Find **Web Server Plugin Pack for Red Hat JBoss Operations Network** in the list and click **Download**.

Consult the JBoss ON documentation for instructions on installing the plugin on your JBoss ON server.

7.2. CONFIGURING TOMCAT FOR JBOSS ON MONITORING

To allow JBoss ON to monitor Tomcat in JBoss Web Server, you must configure Tomcat to allow JBoss ON discovery, as well as providing JBoss ON the required access.



NOTE

For Microsoft Windows, skip this step and proceed to Configuring Tomcat for JBoss ON Monitoring

The JBoss ON agent requires read and write permission to the Tomcat directories. As a user with root privileges, run the following command to add the user which runs the JBoss ON Agent to the **`tomcat`** group:

```
# usermod -aG tomcat <JBOSSON_AGENT_USER>
```

Configuring Tomcat for JBoss ON Monitoring

JBoss Web Server instances are auto-discovered on Linux and Unix platforms. However, you need to configure the instance's JMX to allow for proper handling of authentication and accurate Tomcat monitoring.

To configure JMX to handle authentication:

1. Open the **`startup`** or the **`setenv`** file of the respective JBoss Web Server instance for editing:
   - On Red Hat Enterprise Linux installed from a ZIP file, open

***JWS_HOME*/tomcat<*VERSION*>/bin/setenv.sh**.

- On Red Hat Enterprise Linux installed from RPMs, open **/usr/sbin/tomcat<*VERSION*>**.
- On Microsoft Windows open ***JWS_HOME*\share\tomcat<*VERSION*>\bin\setenv.bat**.
- On Solaris using **daemon.sh** to start Tomcat, open ***JWS_HOME*/tomcat<*VERSION*>/bin/setenv.sh**.

2. Define an available port for JMX monitoring. Ensure the port is not blocked by any firewall.
   - On Red Hat Enterprise Linux and Solaris:

```
JAVA_OPTS="${JAVA_OPTS} -
Dcom.sun.management.jmxremote.port=PORT_NUMBER -
Djava.rmi.server.hostname=IP_ADDRESS"
```

   - On Microsoft Windows:

```
set "JAVA_OPTS=%JAVA_OPTS% -
Dcom.sun.management.jmxremote.port=PORT_NUMBER -
Djava.rmi.server.hostname=IP_ADDRESS"
```

3. In production environments, add the following lines to the JAVA_OPTS variable in the startup file to secure JMX with SSL, and restrict the access with a firewall:
   - On Red Hat Enterprise Linux and Solaris:

```
JAVA_OPTS="${JAVA_OPTS} -
Dcom.sun.management.jmxremote.access.file=JWS_HOME/jmxremote.acce
ss"
JAVA_OPTS="${JAVA_OPTS} -
Dcom.sun.management.jmxremote.password.file=JWS_HOME/jmxremote.pa
ssword"
```

   - On Microsoft Windows:

```
set "JAVA_OPTS=%JAVA_OPTS% -
Dcom.sun.management.jmxremote.access.file=JWS_HOME\jmxremote.acce
ss"
set "JAVA_OPTS=%JAVA_OPTS% -
Dcom.sun.management.jmxremote.password.file=JWS_HOME\jmxremote.pa
ssword"
```

NOTE

If you want to disable authentication and SSL for development purposes, add the following lines to the JAVA_OPTS variable in the startup file:

```
JAVA_OPTS="${JAVA_OPTS} -
Dcom.sun.management.jmxremote.ssl=false"
JAVA_OPTS="${JAVA_OPTS} -
Dcom.sun.management.jmxremote.authenticate=false"
```

4. Once the Tomcat server resource is discovered and imported into the JBoss ON inventory, it may be necessary to update the new resource's connection settings.

   a. In the JBoss ON interface, **Connection Settings** for the newly imported Tomcat server resource.

   b. Verify the value of the **Manager URL** property to the RMI URL, to ensure it uses the correct JMX host name and port number as defined in the Tomcat server startup file. An example for this value is shown below:

```
service:jmx:rmi:///jndi/rmi://$IP_ADDRESS:$PORT/jmxrmi
```

7.2.1. Configuring JBoss ON Monitoring for Tomcat Installed from RPMs

1. In a shell prompt become the root user.

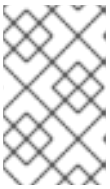
2. Configure the JMX JAVA_OPTS properties in the **/usr/sbin/tomcat<*VERSION*>** file in the **start** and **start-security** sections.

```
if [ "$1" = "start" ]; then
JAVA_OPTS="${JAVA_OPTS} -Dcom.sun.management.jmxremote.port=8100 -
Dcom.sun.management.jmxremote.ssl=false -
Dcom.sun.management.jmxremote.authenticate=true -
Dcom.sun.management.jmxr
emote.access.file="/etc/tomcat<VERSION>/jmxremote.access" -
Dcom.sun.management.jmxremote.password.file="/etc/tomcat<VERSION>/jm
xremote.password""

  ${JAVACMD} $JAVA_OPTS $LOGGING_CONFIG $CATALINA_OPTS \
    -classpath "$CLASSPATH" \
    -Dcatalina.base="$CATALINA_BASE" \
    -Dcatalina.home="$CATALINA_HOME" \
    -Djava.endorsed.dirs="$JAVA_ENDORSED_DIRS" \
    -Djava.io.tmpdir="$CATALINA_TMPDIR" \
    org.apache.catalina.startup.Bootstrap start \
     >> ${CATALINA_BASE}/logs/catalina.out 2>&1 &
    if [ ! -z "$CATALINA_PID" ]; then
      echo $! > $CATALINA_PID
    fi
elif [ "$1" = "start-security" ]; then
JAVA_OPTS="${JAVA_OPTS} -Dcom.sun.management.jmxremote.port=8100 -
Dcom.sun.management.jmxremote.ssl=false -
Dcom.sun.management.jmxremote.authenticate=true -
Dcom.sun.management.jmxr
emote.access.file="/etc/tomcat<VERSION>/jmxremote.access" -
Dcom.sun.management.jmxremote.password.file="/etc/tomcat<VERSION>/jm
xremote.password""

  ${JAVACMD} $JAVA_OPTS $LOGGING_CONFIG $CATALINA_OPTS \
    -classpath "$CLASSPATH" \
    -Dcatalina.base="$CATALINA_BASE" \
```

3. Run the following command to start Tomcat:

```
service tomcat<VERSION> start
```

4. Start the JBoss ON agent.
5. In the JBoss ON Web UI, import the JBoss ON agent and Tomcat.
6. In the JBoss ON Web UI, setup the Tomcat connection configuration (principal and credentials).
7. In the JBoss ON Web UI, set the **Tomcat Control** method configuration to **RPM System V init script**.

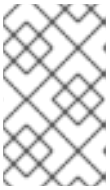


NOTE

The **Start** and **Shutdown** script may not be set because the Tomcat plugin always runs the **`service tomcat<VERSION> start/stop`** command for the **RPM System V init script** configuration setting.

7.2.2. Configuring JBoss ON Monitoring for Tomcat Installed as a Windows Service

1. Create the **`jmxremote.access`** file with **`controlRole readwrite`** in the **`C:\jmx`** directory.
2. Create the **`jmxremote.password`** file with **`controlRole pwd`** in the **`C:\jmx`** directory.



NOTE

Set the owner of **`jmxremote.access`** and **`jmxremote.password`** to SYSTEM, and restrict the access of **`jmxremote.password`** only to SYSTEM. The SYSTEM user must only have read access.

3. Enable JMX for the Tomcat Windows service:

```
JWS_HOME\sbin\tomcat<VERSION>.exe //US//Tomcat<VERSION>
++JvmOptions="-Dcom.sun.management.jmxremote.port=8100;-
Dcom.sun.management.jmxremote.access.file="C:\jmx\jmxremote.access";
-
Dcom.sun.management.jmxremote.password.file="C:\jmx\jmxremote.passwo
rd";-Dcom.sun.management.jmxremote.ssl=false;-
Dcom.sun.management.jmxremote.authenticate=true"
```

4. Start the Tomcat Windows service, and verify that it is running.
5. Install and configure the JBoss ON agent. Type **`discovery`** in the agent prompt to discover the Tomcat Windows service.
6. In the JBoss ON Web UI, click **Inventory**, then click **Discovery Queue**, and select **import Tomcat and RHQ agent**.
7. In the JBoss ON Web UI, go to **Platforms** and search for the agent name. Click on your agent.
8. On the **Agent** page, Tomcat Servers are listed. Select your Tomcat server by clicking it.
9. In the JBoss ON Web UI, click on the **Inventory** tab and then configure the Tomcat Server in **Connection Settings**.

10. Enter the Principal and Credentials information. Use the **`controlRole`** and password set in the **`jmxremote`** files.

11. Set the control method to **RPM System V init script**.

NOTE

You cannot set **Start** and **Shutdown** script fields.

12. Click **Save**.

13. Update the connection settings of the Tomcat Server JVM, and set Principal and Credentials.

CHAPTER 8. USING A PASSWORD VAULT WITH RED HAT JBOSS WEB SERVER 3

8.1. USING A PASSWORD VAULT WITH RED HAT JBOSS WEB SERVER 3

A password vault is used to mask passwords and other sensitive strings, and store them in an encrypted Java keystore. This allows you to eliminate storing clear-text passwords in your Tomcat configuration files, as Tomcat can lookup passwords and other sensitive strings from a keystore using the vault.

The examples and commands below use **JWS_HOME** as the JBoss Web Server installation directory. Replace **JWS_HOME** with the path to your JBoss Web Server installation. Also, the paths below use **/** for directory separators.

8.1.1. Installing the JBoss Web Server password vault

There are two methods of installing the JBoss Web Server password vault:

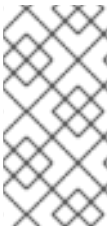
- Installing the JBoss Web Server password vault on Red Hat Enterprise Linux from an RPM(For Red Hat Enterprise Linux systems where JBoss Web Server was installed using **yum** only).
- Downloading and Extracting the Vault Files from a .zip archive (Available for all JBoss Web Server installations).

8.1.1.1. Installing the JBoss Web Server password vault on Red Hat Enterprise Linux from an RPM

Where the JBoss Web Server has been installed from RPMs on Red Hat Enterprise Linux, install the password vault as the root user by executing:

```
yum install tomcat-vault tomcat-vault-tomcat<VERSION>
```

Where **<VERSION>** is either **7** for tomcat 7 (**tomcat-vault-tomcat7**) or **8** for tomcat 8 (**tomcat-vault-tomcat8**).



NOTE

In the tomcat-vault RPM installation, the vault jar is located in **/usr/share/java/vault-tomcat-<VERSION>-jar-with-dependencies.jar**. This jar can be used in JWS zip installation: **JWS_HOME/tomcat_<VERSION>_/lib/** For JWS RPM installation: **/usr/share/tomcat<VERSION>/lib**.

8.1.1.2. Downloading and Extracting the Vault Files from a .zip archive

1. Stop Tomcat if it is running.
2. Extract the contents of the vault zip to your **JWS_HOME** directory. In this topic, **JWS_HOME/tomcat-vault** will refer to the extracted vault directory.
3. Copy **JWS_HOME/tomcat-vault/modules/system/layers/base/tomcat-vault/main/tomcat-vault.jar** to **JWS_HOME/tomcat<VERSION>/lib/**. Edit **JWS_HOME/tomcat<VERSION>/conf/catalina.properties**, and add the following line:

```
org.apache.tomcat.util.digester.PROPERTY_SOURCE=org.apache.tomcat.vault.ut
il.PropertySourceVault
```

8.1.2. Creating a Java Keystore

To use a password vault, you must first create a Java keystore. You can do this using the **`keytool -genseckey`** command. For example:

```
$ keytool -genseckey -keystore JWS_HOME/tomcat/vault.keystore -alias
my_vault  -storetype jceks -keyalg AES -keysize 128 -storepass
<vault_password> -keypass <vault_password> -validity 730
```

IMPORTANT

The values above are examples only. Replace them with values specific to your environment.

For an explanation of the parameters, use the **`keytool -genseckey -help`** command.

8.1.3. Storing the tomcat-vault `vault.properties` file outside of the JWS_HOME directory

This feature was introduced by JBoss Web Server 3.1 Service Pack 2.

The **`vault.properties`** file for the **`tomcat-vault`** can be stored outside of ***`JWS_HOME`*`/tomcat<VERSION>/conf/`** in a ***`CATALINA_BASE`*`/conf/`** directory (if set).

To set the *CATALINA_BASE* directory, follow the instructions in the section '**Advanced Configuration - Multiple Tomcat Instances**' in the Running The Apache Tomcat 8.0 Servlet/JSP Container document found on the Apache Tomcat Website.

NOTE

The default location for *CATALINA_BASE* is ***`JWS_HOME`*`/tomcat<VERSION>/`** (also known as *CATALINA_HOME*).

For more information on setting *CATALINA_BASE*, see:

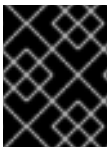
- Apache Tomcat 8: Introduction - Directories and Files
- Running The Apache Tomcat 8.0 Servlet/JSP Container: Advanced Configuration - Multiple Tomcat Instances

8.1.4. Initializing the Password Vault

The vault must be initialized before it can be used to store sensitive strings. This is done using the **`JWS_HOME/tomcat-vault/bin/tomcat-vault.sh`** vault script. For Microsoft Windows, the script is **`tomcat-vault.bat`**.

The script can be run interactively or non-interactively. Below is an example of an interactive execution of the script to initialize a password vault, with the values shown below using the example keystore from the previous step.

8.1.4.1. Initializing the Vault for Apache Tomcat interactively



IMPORTANT

The values below are examples only. Replace them with values appropriate for your environment.

```
# JWS_HOME/tomcat-vault/bin/tomcat-vault.sh

WARNING JBOSS_HOME may be pointing to a different installation -
unpredictable results may occur.

=========================================================================

  JBoss Vault

  JBOSS_HOME: JWS_HOME/tomcat-vault

  JAVA: java

=========================================================================

*********************************
****  JBoss Vault  ***************
*********************************
Please enter a Digit::
0: Start Interactive Session
1: Remove Interactive Session
2: Exit

0

Starting an interactive session
Enter directory to store encrypted files: JWS_HOME/tomcat-vault/
Enter Keystore URL: JWS_HOME/tomcat-vault/vault.keystore
Enter Keystore password: <vault_password>
Enter Keystore password again: <vault_password>
Values match
Enter 8 character salt: 1234abcd
Enter iteration count as a number (Eg: 44): 120
Enter Keystore Alias: my_vault
Initializing Vault
Jun 16, 2018 10:24:27 AM
org.apache.tomcat.vault.security.vault.PicketBoxSecurityVault init
INFO: PBOX000361: Default Security Vault Implementation Initialized and
Ready
Vault Configuration in tomcat properties file:
********************************************
...
KEYSTORE_URL=JWS_HOME/tomcat-vault/vault.keystore
KEYSTORE_PASSWORD=MASK-3CuP21KMHn7G6iH/A3YpM/
KEYSTORE_ALIAS=my_vault
SALT=1234abcd
ITERATION_COUNT=120
ENC_FILE_DIR=JWS_HOME/tomcat-vault/
```

```
...
**
Vault is initialized and ready for use
Handshake with Vault complete
Please enter a Digit::
0: Store a secured attribute
1: Check whether a secured attribute exists
2: Exit

2
```

Note the output for the Tomcat properties file, as you will need this to configure Tomcat to use the vault.

Configuring Tomcat to Use the Password Vault
In `JWS_HOME/tomcat<VERSION>/conf/`, create a file named `vault.properties` containing the vault configuration produced when initializing the vault. The values provided below use the example vault initialized in the previous steps.

NOTE

For `KEYSTORE_PASSWORD`, you must use the masked value that was generated when initializing the vault.

```
KEYSTORE_URL=JWS_HOME/tomcat-vault/vault.keystore
KEYSTORE_PASSWORD=MASK-3CuP21KMHn7G6iH/A3YpM/
KEYSTORE_ALIAS=my_vault
SALT=1234abcd
ITERATION_COUNT=120
ENC_FILE_DIR=JWS_HOME/tomcat-vault/
```

8.1.4.2. Initializing the Vault for Apache Tomcat non-interactively (silent setup)

The Vault for Apache Tomcat can be created non-interactively by providing the required input as arguments to the `tomcat-vault.sh` script. The `vault.properties` file is also created as output of the `tomcat-vault.sh` script when the `-g, --generate-config` option is used.

IMPORTANT

The values below are examples only. Replace them with values appropriate for your environment.

```
$ JWS_HOME/tomcat-vault/bin/tomcat-vault.sh \
 --keystore JWS_HOME/tomcat-vault/vault.keystore \
 --keystore-password <vault_password> \
 --alias my_vault \
 --enc-dir JWS_HOME/tomcat-vault/ \
 --iteration 120 \
 --salt 1234abcd \
 --generate-config JWS_HOME/tomcat<VERSION>/conf/vault.properties
```

8.1.5. Storing a Sensitive String in the Password Vault

The vault script used in the previous steps is also used to store sensitive strings in the password vault. The script can be run interactively or non-interactively.

When adding a string to a password vault, the sensitive string needs a name that it will be referred by. For a password vault, this name is called an `attribute name`, and the password itself is called a `secured attribute`.

The example below demonstrates using the vault script non-interactively to store a password. It uses the vault that was initialized in the previous steps, and stores the sensitive string `P@SSW0#D` with the attribute name `manager_password`.

```
$ JWS_HOME/tomcat-vault/bin/tomcat-vault.sh --keystore JWS_HOME/tomcat-
vault/vault.keystore --keystore-password <vault_password> --alias my_vault
--enc-dir JWS_HOME/tomcat-vault/ --iteration 120 --salt 1234abcd --vault-
block my_block --attribute manager_password --sec-attr P@SSW0#D
```

NOTE

You can optionally specify a vault block to store the password in. If you don't specify a block, one will be automatically created for you. In the above example, my_block is used.

8.1.6. Using a Stored Sensitive String in Your Tomcat Configuration

After storing a sensitive string in the password vault, you can refer to it in your configuration files by entering the stored string's attribute as `${VAULT::block_name::attribute_name::}`.

For example, to use the password stored in the previous steps, replace:

```
<user username="manager" password="P@SSW0#D" roles="manager-gui"/>
```

with:

```
<user username="manager" password="${VAULT::my_block::manager_password::}"
roles="manager-gui"/>
```

As a result, only a reference to the password is visible in the Tomcat configuration file, and the actual password is only stored in the password vault.

CHAPTER 9. CONFIGURING JWS CLIENT-SERVER COMMUNICATION WITH WEBSOCKET

9.1. ABOUT WEBSOCKET

WebSocket is a web technology that provides bi-directional, full duplex messages to be instantly distributed between a client and server over a single TCP socket connection. A full duplex communication allows two-way communication simultaneously.

The container provides an implementation of the WebSockets 1.0 JSR 356 API. To use the API, you must run Java 7 or later, and configure the APR or NIO2 HTTP/1.1 connectors of the web container.

JSR 356 is the standard for WebSocket API for Java. Developers can use the JSR 356 API for creating WebSocket applications independent of the implementation. The WebSocket API is purely event driven.

Developers can use the JSR 356 Java API for WebSocket to integrate WebSockets in applications on the server side as well as on the client side. Tomcat 7 and 8 implement the WebSocket protocol, which adheres to JSR-356 standard.

A Java client uses a JSR 356-compliant client implementation to connect to a WebSocket server. For web clients, WebSocket JavaScript API can be used to communicate with WebSocket server. The only difference between a WebSocket client and a WebSocket server is the method in which they are connected. A WebSocket client is a WebSocket point from which the connection to a peer originates. A WebSocket server is WebSocket endpoint which is already published and awaits connections from peers.

Some examples where WebSocket can be used include banking, chat, multiplayer, and social networking applications.

9.2. IMPLEMENTING WEBSOCKET ON TOMCAT

Configuring WebSocket on Tomcat requires individual configuration of the following:

- write timeout
- incoming binary messages
- incoming text messages
- additional programmatic deployment
- callbacks for asynchronous writes
- timeout for IO operations while establishing the connections

9.2.1. Configuring Write Timeout

You can change the write timeout in blocking mode by using the `org.apache.tomcat.websocket.BLOCKING_SEND_TIMEOUT` property. The property accepts values in milliseconds. The default value is `20000` (20 seconds).

9.2.2. Configuring Incoming Binary Messages

To configure incoming binary messages, `MessageHandler.Partial` must be defined. If `MessageHandler.Partial` is not defined, then incoming binary messages must be buffered so that the entire message is delivered in a single call to `MessageHandler.Whole`.

The default buffer size for binary messages is 8192 bytes. You can change the buffer size for a web application by changing the value of the servlet context initializing parameter `org.apache.tomcat.websocket.binaryBufferSize`.

9.2.3. Configuring Incoming Text Messages

To configure incoming text messages, `MessageHandler.Partial` must be defined. If `MessageHandler.Partial` is not defined then incoming text messages must be buffered so that the entire message is delivered in a single call to `MessageHandler.Whole`.

The default buffer size for text messages is 8192 bytes. You can change the buffer size for a web application by changing the value of the servlet context initializing parameter `org.apache.tomcat.websocket.textBufferSize`.

9.2.4. Configuring Additional Programmatic Deployment

The Java WebSocket 1.0 specification does not allow programmatic deployment after the first endpoint has started a WebSocket handshake. However, Tomcat by default allows additional programmatic deployment. Additional programmatic deployment can be done by using the servlet context initialization parameter `org.apache.tomcat.websocket.noAddAfterHandshake`.

Set the system property `org.apache.tomcat.websocket.STRICT_SPEC_COMPLIANCE` to `true` to change the default setting.

9.2.5. Configuring Callbacks for Asynchronous Writes

Callbacks for asynchronous writes need to be performed on a different thread to the thread that initiated the write. The container thread pool is not exposed via the Servlet API. Thus the WebSocket implementation has to provide its own thread pool.

The following servlet context initialization parameters control the thread pool:

`org.apache.tomcat.websocket.executorCoreSize`

The core size of the executor thread pool. If not set, the default of `0` (zero) is used.

`org.apache.tomcat.websocket.executorMaxSize`

The maximum permitted size of the executor thread pool. If not set, the default of `10` is used.

`org.apache.tomcat.websocket.executorKeepAliveTimeSeconds`

The maximum time an idle thread will remain in the executor thread pool until it is terminated. If not specified, the default of `60` seconds is used.

9.2.6. Configuring Timeout for IO Operations While Establishing the Connections

The timeout for IO operations while establishing the connections is controlled by the `userProperties` of the provided `javax.websocket.ClientEndpointConfig`. You can change timeout by changing the `org.apache.tomcat.websocket.IO_TIMEOUT_MS` property. The property accepts the values in milliseconds. The default value is `5000` (5 seconds).

To connect a WebSocket client to secure server endpoints, the client SSL configuration is controlled by the `userProperties` of the provided `javax.websocket.ClientEndpointConfig`.

The following user properties are supported:

- **`org.apache.tomcat.websocket.SSL_CONTEXT`**
- **`org.apache.tomcat.websocket.SSL_PROTOCOLS`**
- **`org.apache.tomcat.websocket.SSL_TRUSTSTORE`**
- **`org.apache.tomcat.websocket.SSL_TRUSTSTORE_PWD`**

The default truststore password is `changeit`. The `org.apache.tomcat.websocket.SSL_TRUSTSTORE` and `org.apache.tomcat.websocket.SSL_TRUSTSTORE_PWD` properties are ignored if the `org.apache.tomcat.websocket.SSL_CONTEXT` property is set.

APPENDIX A. JAVA IPV4/IPV6 PROPERTIES

Configuring Java Properties

In Java ther are 2 properties that are used to configure IPv4 and IPv6. These are `java.net.preferIPv4Stack` and `java.net.preferIPv6Addresses`.

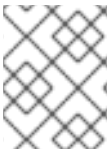
java.net.preferIPv4Stack (default: false)

If IPv6 is available then the underlying native socket, by default, is an IPv6 socket. This socket lets applications connect and accept connections from IPv4 and IPv6 hosts. If application use only IPv4 sockets, then set this property to `true`. However, it will not be possible for the application to communicate with IPv6 only hosts.

java.net.preferIPv6Addresses (default: false)

If a host has both IPv4 and IPv6 addresses, and IPv6 is available, then the default behavior is to use IPv4 addresses over IPv6. This allows backward compatibility. If applications that depend on an IPv4 address representation, for example: 192.168.1.1. Then, set this property to `true` to change the preference and use IPv6 addresses over IPv4 where possible.

To pass these properties to Tomcat, set `CATALINA_OPTS` in the *`CATALINA_HOME`*`/bin/setenv.*` file.



NOTE

If the *`CATALINA_HOME`*`/bin/setenv.sh` or *`CATALINA_HOME`*`/bin/setenv.bat` file does not exist, then you need to create one.

On Linux:

```
export "CATALINA_OPTS=-Djava.net.preferIPv4Stack=YOUR_VALUE -
Djava.net.preferIPv6Addresses=YOUR_VALUE"
```

On Windows:

```
set "CATALINA_OPTS=-Djava.net.preferIPv4Stack=YOUR_VALUE -
Djava.net.preferIPv6Addresses=YOUR_VALUE"
```

Configuring Tomcat Bindings

The Tomcat bindings can be set in `CATALINA_HOME/conf/server.xml` with the IPv6 address:

- **Specify the Tomcat binding address:**
  `<Server … address="`*`TOMCAT_BINDING_ADDRESS`*`">`
- **Specify the HTTP connector address:**
  `<Connector protocol="HTTP/1.1" … address="`*`HTTP_CONNECTOR_ADDRESS`*`">`
- **Specify the AJP connector address:**
  `<Connector protocol="AJP/1.3" … address="`*`AJP_CONNECTOR_ADDRESS`*`">`