# Red Hat JBoss Portal 6.2 Installation Guide

for use with Red Hat JBoss Portal 6.2 and its patch releases.

Jared Morgan Aakanksha Singh

# Red Hat JBoss Portal 6.2 Installation Guide

for use with Red Hat JBoss Portal 6.2 and its patch releases.

Jared Morgan
Red Hat, Ltd. Customer Content Services

Aakanksha Singh
Red Hat, Ltd. Customer Content Services

## Legal Notice



## Abstract

This book provides information about obtaining, installing and running Red Hat JBoss Portal. It forms part of the complete document suite available at

# Table of Contents

# Preface

## 1. Document Conventions

This manual uses several conventions to highlight certain words and phrases and draw attention to specific pieces of information.

### 1.1. Typographic Conventions

Four typographic conventions are used to call attention to specific words and phrases. These conventions, and the circumstances they apply to, are as follows.

**`Mono-spaced Bold`**

Used to highlight system input, including shell commands, file names and paths. Also used to highlight keys and key combinations. For example:

> To see the contents of the file **`my_next_bestselling_novel`** in your current working directory, enter the **`cat my_next_bestselling_novel`** command at the shell prompt and press **`Enter`** to execute the command.

The above includes a file name, a shell command and a key, all presented in mono-spaced bold and all distinguishable thanks to context.

Key combinations can be distinguished from an individual key by the plus sign that connects each part of a key combination. For example:

> Press **`Enter`** to execute the command.
>
> Press **`Ctrl`**+**`Alt`**+**`F2`** to switch to a virtual terminal.

The first example highlights a particular key to press. The second example highlights a key combination: a set of three keys pressed simultaneously.

If source code is discussed, class names, methods, functions, variable names and returned values mentioned within a paragraph will be presented as above, in **`mono-spaced bold`**. For example:

> File-related classes include **`filesystem`** for file systems, **`file`** for files, and **`dir`** for directories. Each class has its own associated set of permissions.

**Proportional Bold**

This denotes words or phrases encountered on a system, including application names; dialog-box text; labeled buttons; check-box and radio-button labels; menu titles and submenu titles. For example:

> Choose **System** → **Preferences** → **Mouse** from the main menu bar to launch **Mouse Preferences**. In the **Buttons** tab, select the **`Left-handed mouse`** check box and click **`Close`** to switch the primary mouse button from the left to the right (making the mouse suitable for use in the left hand).
>
> To insert a special character into a **`gedit`** file, choose **Applications** → **Accessories** → **Character Map** from the main menu bar. Next, choose **Search** → **Find…** from the **Character Map** menu bar, type the name of the character in the **Search** field and click **`Next`**. The character you sought will be highlighted in the

> **Character Table**. Double-click this highlighted character to place it in the **Text to copy** field and then click the **Copy** button. Now switch back to your document and choose **Edit** → **Paste** from the **gedit** menu bar.

The above text includes application names; system-wide menu names and items; application-specific menu names; and buttons and text found within a GUI interface, all presented in proportional bold and all distinguishable by context.

***Mono-spaced Bold Italic*** or ***Proportional Bold Italic***

Whether mono-spaced bold or proportional bold, the addition of italics indicates replaceable or variable text. Italics denotes text you do not input literally or displayed text that changes depending on circumstance. For example:

> To connect to a remote machine using ssh, type **ssh *username*@*domain.name*** at a shell prompt. If the remote machine is **example.com** and your username on that machine is john, type **ssh john@example.com**.
>
> The **mount -o remount *file-system*** command remounts the named file system. For example, to remount the **/home** file system, the command is **mount -o remount /home**.
>
> To see the version of a currently installed package, use the **rpm -q *package*** command. It will return a result as follows: ***package-version-release***.

Note the words in bold italics above: username, domain.name, file-system, package, version and release. Each word is a placeholder, either for text you enter when issuing a command or for text displayed by the system.

Aside from standard usage for presenting the title of a work, italics denotes the first use of a new and important term. For example:

> Publican is a *DocBook* publishing system.

## 1.2. Pull-quote Conventions

Terminal output and source code listings are set off visually from the surrounding text.

Output sent to a terminal is set in **mono-spaced roman** and presented thus:

```
books        Desktop   documentation  drafts  mss    photos   stuff  svn
books_tests  Desktop1  downloads      images  notes  scripts  svgs
```

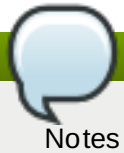
Source-code listings are also set in **mono-spaced roman** but add syntax highlighting as follows:

```
static int kvm_vm_ioctl_deassign_device(struct kvm *kvm,
                  struct kvm_assigned_pci_dev *assigned_dev)
{
         int r = 0;
         struct kvm_assigned_dev_kernel *match;

         mutex_lock(&kvm->lock);

         match = kvm_find_assigned_dev(&kvm->arch.assigned_dev_head,
                                       assigned_dev->assigned_dev_id);
         if (!match) {
                 printk(KERN_INFO "%s: device hasn't been assigned
```

```
before, "
                      "so cannot be deassigned\n", __func__);
                    r = -EINVAL;
                    goto out;
            }

            kvm_deassign_device(kvm, match);

            kvm_free_assigned_device(kvm, match);

out:
            mutex_unlock(&kvm->lock);
            return r;
}
```

### 1.3. Notes and Warnings

Finally, we use three visual styles to draw attention to information that might otherwise be overlooked.



**Note**

Notes are tips, shortcuts or alternative approaches to the task at hand. Ignoring a note should have no negative consequences, but you might miss out on a trick that makes your life easier.

**Important**

Important boxes detail things that are easily missed: configuration changes that only apply to the current session, or services that need restarting before an update will apply. Ignoring a box labeled “Important” will not cause data loss but may cause irritation and frustration.

**Warning**

Warnings should not be ignored. Ignoring warnings will most likely cause data loss.

## 2. Getting Help and Giving Feedback

### 2.1. Do You Need Help?

If you experience difficulty with a procedure described in this documentation, visit the Red Hat Customer Portal at http://access.redhat.com. From the Customer Portal, you can:

- Search or browse through a knowledge base of technical support articles about Red Hat products.
- Submit a support case to Red Hat Global Support Services (GSS).
- Access other product documentation.

Red Hat also hosts a large number of electronic mailing lists for discussion of Red Hat software and technology. You can find a list of publicly available mailing lists at https://www.redhat.com/mailman/listinfo. Click the name of any mailing list to subscribe to that list or to access the list archives.

## 2.2. We Need Feedback

If you find a typographical error in this manual, or if you have thought of a way to make this manual better, we would love to hear from you. Please submit a report in Bugzilla: http://bugzilla.redhat.com/ against the product Red Hat JBoss Portal.

When submitting a bug report, be sure to mention the manual's identifier: *Installation_Guide*

If you have a suggestion for improving the documentation, try to be as specific as possible when describing it. If you have found an error, please include the section number and some of the surrounding text so we can find it easily.

# Chapter 1. Documentation Advice

## 1.1. Cross-product content disclaimer

While this document has been created to assist with a particular Red Hat JBoss product, the interoperability of Red Hat's middleware range may mean that content written for another product's documentation is suitably applicable to this document as well.

In these instances, all care is taken to ensure the information provided is usable and relevant although the content may mention a different product by name.

Report a bug

# Chapter 2. Introduction

## 2.1. Overview

The Red Hat JBoss Portal (JBoss Portal) *Installation Guide* is for system administrators who need to download, install and configure JBoss Portal in a production environment.

The guide includes instructions for installing a new Portal. It also includes instructions for configuring the mandatory settings required to start a portal and have it operate correctly.

The guide does not contain instructions for migrating from extant JBoss Portal instances. See the JBoss Portal *Migration Guide* for considerations and guidance in performing this task.

Report a bug

## 2.2. New and Changed Features

See the Release Notes for the latest features and improvements to Red Hat JBoss Portal. The Release Notes are hosted on the Red Hat Customer Portal, which can be accessed from https://access.redhat.com/site/documentation/Red_Hat_JBoss_Portal/

Report a bug

# Chapter 3. Platform Requirements

## 3.1. Java Requirements

The definitive list of supported Java environments is available from https://access.redhat.com/site/articles/119833.

JBoss Portal requires a working Java Development Kit (JDK) 1.6 or 1.7 installation. Both 32-bit and 64-bit JDKs are supported.

Report a bug

## 3.2. Hardware Requirements

To install Red Hat JBoss Portal (JBoss Portal) requires at least 550 MB of available storage. The mandatory Java Development Kit (JDK) requires another 150 MB

To run JBoss Portal, the minimum operating requirements are:

**Disk space: 1.5 GB**

This includes the 700 MB required for a base install, including JDK; 500 MB for the server log file (the default configuration for the log file); and 330 MB for deployed JBoss applications.

**CPU: 1 GHz Intel Pentium Processor**

Intel Core 2 Duo, Intel Core 2 Quad and Intel Xeon CPUs improve server performance for high demand applications.

**RAM: 1.5 GB**

This is the bare minimum required to run JBoss Portal.

To deploy and run small to medium applications, 2 GB or more is required.

4 GB or more is recommended to run larger applications, or to run a server managed via a Graphical User Interface (GUI).

Specific information on supported JBoss Portal technology combinations is available in the Supported Configuration resource, located at https://access.redhat.com/site/articles/119833.

Report a bug

## 3.3. Operational Requirements

As the minimum requirements note, multiple factors affect the server hardware required to effectively support the Red Hat JBoss Portal (JBoss Portal).

Broadly considered, JBoss Portal server performance is tied to three distinct but related attributes: the deployed applications; the client load; and the server configuration. Specific factors to consider include:

- application size.
- application complexity.

- client numbers.
- client request frequency.
- log file size.
- post-installation server tuning and configuration.

Report a bug

# Chapter 4. Download the Platform

## 4.1. About the Red Hat Customer Portal

The *Red Hat Customer Portal* is the centralized platform for Red Hat knowledge and subscription resources. Use the *Red Hat Customer Portal* to do the following:

- Manage and maintain Red Hat entitlements and support contracts.
- Download officially-supported software.
- Access product documentation and the Red Hat Knowledgebase.
- Contact Global Support Services.
- File bugs against Red Hat products.

The Customer Portal is available here: https://access.redhat.com.

Report a bug

## 4.2. Available Downloads

Red Hat JBoss Portal (JBoss Portal) includes a number of different installation types and optional components, which are available for download on the Red Hat Customer Portal at https://access.redhat.com/.

Table 4.1, “Available Downloads” describes the different options, including components which are appropriate for certain operating systems or architectures.

**Table 4.1. Available Downloads**

| Name | Description | Operating Systems |
|---|---|---|
| Documentation | The documents are available at https://access.redhat.com/site/documentation/Red_Hat_JBoss_Portal/ | Platform-independent |
| Source Code | The Java source code for JBoss Portal is provided. | Platform-independent |
| Portal Platform | The ZIP installation package which can be installed and run on every supported platform. | Platform-independent |
| Quickstarts and wolf repo | Both quickstarts and maven repository ZIP packages are provided and can be used on every supported platform. | Platform-independent |

Report a bug

## 4.3. Download the Platform and Related Plug-ins

### 4.3.1. Download Red Hat JBoss Portal

Before starting the download procedure, log onto the Red Hat Customer Portal with valid user credentials.

1. Load https://access.redhat.com in a web browser.
2. Click the **Downloads** option, and then select **Red Hat JBoss Portal** from the alphabetical list.
3. Select the latest released version from the **Version** field to display the available downloads
4. Click the **Download** link next to the Red Hat JBoss Portal 6.2 item to begin the download.
5. Repeat the previous step for any supplemental packages required (for example, the Quickstarts archive).

**Next Step in Download the Platform and Related Plug-ins**

- Section 4.3.2, “Download JBoss Operations Network Plug-in”

Report a bug

### 4.3.2. Download JBoss Operations Network Plug-in

**Previous Step in Download the Platform and Related Plug-ins**

- Section 4.3.1, “Download Red Hat JBoss Portal”

Install the JBoss Operations Network JBoss Portal plug-in to manage the production server through a graphical interface. You must be logged onto the portal with valid user entitlements to access JBoss Operations Network downloads.

1. Load https://access.redhat.com in a web browser.
2. Click the **Downloads** option, and then select **Red Hat JBoss Operations Network** from the alphabetical list.
3. Select **JBoss ON for Portal Platform** from the **Product** drop-down menu.
4. Select the latest version available from the **Version** field.
5. Download the following files by clicking the **Download** link beside each item:
   - Base Distribution
   - Portal Platform Plug-in Pack
   - All Patches applicable to the version you are downloading.

     Patches are applied to JBoss Operations Network after installing the base distribution and plug-in.

Report a bug

# Chapter 5. Install the Platform

## 5.1. Install OpenJDK on Red Hat Enterprise Linux

### Download and Install OpenJDK

Complete this task to install OpenJDK on Red Hat Enterprise Linux, and configure the system to use it as the default Java installation. OpenJDK is one of many Java Development Kits (JDKs) supported in Red Hat Enterprise Linux for use with Red Hat JBoss products.

**Note**

It is possible to install multiple JDKs on a single system.

### Prerequisites

Your system must meet the following conditions before continuing with this task:

- Red Hat Enterprise Linux is installed and running on the server hosting the Red Hat JBoss server.
- The server running Red Hat Enterprise Linux must be registered with RHN and subscribed to the base channel. See https://access.redhat.com/site/documentation/en-US/Red_Hat_Enterprise_Linux/7/html/System_Administrators_Guide/part-Package_Management.html for more information about managing subscriptions and entitlements on Red Hat Enterprise Linux systems.

1. **Install the OpenJDK RPM**

   Install OpenJDK using the yum command-line interface as a root user. Install the latest compatible version of OpenJDK for your Red Hat Enterprise Linux installation.

   **OpenJDK 6**

   ```
   yum install java-1.6.0-openjdk-devel
   ```

   **OpenJDK 7**

   ```
   yum install java-1.7.0-openjdk-devel
   ```

2. **Optional: Install and configure the alternatives utility**

   Red Hat Enterprise Linux includes a utility called **alternatives**, which lets you change the default Java version for applications which allow multiple versions to be installed. OpenJDK is one such application.

   To use the **alternatives** utility, perform the following steps.

   a. Log in as the **root** user, or prefix the commands below with the **sudo** command.

   b. The **alternatives** command is not installed by default, but may already be installed on your system. If not, install the **alternatives** package by running the following command:

```
yum install alternatives
```

c. Issue the following command:

```
/usr/sbin/alternatives --config java
```

d. Follow the prompts to set the default version of OpenJDK.

3. **Optional: Set the `JAVA_HOME` environment variable**

   Some applications, such as Apache Maven and Apache Ant, and utilities, such as **`alternatives`** require you to set the **`JAVA_HOME`** environment variable. The **`JAVA_HOME`** variable points to the **`bin`** directory containing the required Java executable. To set the **`JAVA_HOME`** environment variable perform the following steps.

   a. Determine the correct value for **`JAVA_HOME`**. For example, Red Hat Enterprise Linux installs OpenJDK 1.6 into either **`/usr/lib/jvm/java-1.6.0-openjdk-1.6.0.0/`** or **`/usr/lib/jvm/java-1.6.0-openjdk-1.6.0.0.x86_64/`**, depending on whether your system is a 32-bit or 64-bit architecture.

   b. As the user who will use OpenJDK, open the shell configuration file. For the Bash shell, this file is **`/home/`*`username`*`/.bashrc`**.

   c. At the bottom of the file, type the following line, replacing the hypothetical path with the actual path to use on your own system: **`export JAVA_HOME="`*`/path/to/java/home`*`"`**

      If you are setting the environment variable for **`alternatives`** utility, then at the bottom of the file type the following line: **`export JAVA_HOME="/etc/alternatives/jre"`**

   d. Save the file, and log out of and back into your session.

OpenJDK is installed on the server and available for use. If necessary, the **`JAVA_HOME`** environment variable has been specified as well. If necessary, the default OpenJDK for your system has been set using the **`alternatives`** utility.

Report a bug

## 5.2. Install Red Hat JBoss Portal

1. Navigate to the parent directory where the Red Hat JBoss Portal (JBoss Portal) Zip archive will be extracted.

   Before continuing, verify the chosen directory, and it's access permissions, meet organizational security and deployment requirements.

2. Unzip **`jboss-portal-`*`[version]`*`.zip`** to extract the Zip archive contents to the directory.

   A command-line or GUI archive manager is suitable for this task, depending on the server environment.

   The **`jboss-portal-`*`[version]`*** directory is created with an installation of JBoss Portal in its default configuration.

**See Also:**

- Section 4.3.1, “Download Red Hat JBoss Portal”

Report a bug

## 5.3. Install and Configure JBoss Operations Network Plug-in

Red Hat JBoss Portal provides a JBoss Operations Network agent plug-in pack, which is available for download for customers with a "JPP with Management" entitlement.

Information about what the agent plug-in offers is located in the *Administration and Monitoring* section of the *Administration and Configuration Guide*, which is hosted at https://access.redhat.com/site/documentation/en-US/Red_Hat_JBoss_Portal/6.1/html/Administration_and_Configuration_Guide/part-Administration_and_Monitoring.html.

You must download and install the Red Hat JBoss Operations Network server before you can install the Agent Plug-in. See the Red Hat JBoss Operations Network Installation Guide for instructions relating to installing and configuring Red Hat JBoss Operations Network.

See Installing JBoss Agent Plug-in Packs in the JON Installation Guide for the procedure to install the agent plug-in pack downloaded earlier.

**See Also:**

- Section 4.3.2, “Download JBoss Operations Network Plug-in”

Report a bug

## 5.4. Installation Structure and Details

Red Hat JBoss Portal (JBoss Portal) includes a simplified directory structure, compared to previous versions. Following is a listing of the directory structure, and a description of what the directory contains.

**Example 5.1. Top Level Directories**

The top level directories include the gatein-management and gatein-sso directories that contain gatein war files and gatein sso, saml,josso, opensso and cas related directories. The top level directories also include the jboss-jpp-6.2 directory that consist of directories related to domain mode configuration and standalone mode configuration, other binaries, logging properties files and so on.

```
jboss-jpp-6.2
├── appclient
│   └── configuration
├── bin
│   ├── add-user.bat
│   ├── add-user.sh
│   ├── appclient.bat
│   ├── appclient.conf
│   ├── appclient.conf.bat
│   ├── appclient.sh
│   ├── client
│   ├── demo-domain-setup.sh
```

```
|   ├── domain.bat
|   ├── domain.conf
|   ├── domain.conf.bat
|   ├── domain.sh
|   ├── init.d
|   ├── jboss-cli.bat
|   ├── jboss-cli-logging.properties
|   ├── jboss-cli.sh
|   ├── jboss-cli.xml
|   ├── jconsole.bat
|   ├── jconsole.sh
|   ├── jdr.bat
|   ├── jdr.sh
|   ├── portal-setup.bat
|   ├── portal-setup.sh
|   ├── product.conf
|   ├── run.bat
|   ├── run.sh
|   ├── standalone.bat
|   ├── standalone.conf
|   ├── standalone.conf.bat
|   ├── standalone.sh
|   ├── vault.bat
|   ├── vault.sh
|   ├── wsconsume.bat
|   ├── wsconsume.sh
|   ├── wsprovide.bat
|   └── wsprovide.sh
├── bundles
|   └── system
├── docs
|   ├── examples
|   ├── licenses
|   └── schema
├── domain
|   ├── configuration
|   ├── data
|   └── tmp
├── gatein
|   ├── extensions
|   └── gatein.ear
├── JBossEULA.txt
├── jboss-modules.jar
├── LICENSE.txt
├── modules
|   ├── layers.conf
|   └── system
|       └── layers
|           ├── base
|           └── gatein
├── standalone
|   ├── configuration
|   ├── data
|   ├── deployments
|   ├── lib
|   ├── log
```

```
│   └── tmp
├── version.txt
└── welcome-content
    ├── eap.css
    ├── favicon.ico
    ├── images
    ├── index.html
    ├── index_noconsole.html
    ├── jpp.css
    ├── noconsole.html
    └── noredirect.html
```

```
gatein-management
└── gatein-management-cli.war
```

```
gatein-sso
├── cas
│   └── plugin
├── josso
│   ├── gatein-josso-181
│   ├── gatein-josso-182
│   ├── josso-181
│   └── josso-182
├── opensso
│   └── plugin
├── README
└── saml
    └── idp-sig.war
```

**Example 5.2. Directories within the gatein/ directory**

```
gatein
├── extensions
│   ├── gatein-wsrp-integration.ear
│   │   ├── extension-war.war
│   │   ├── lib
│   │   ├── META-INF
│   │   ├── wsrp-admin-gui.war
│   │   └── wsrp-producer.war
│   ├── jpp-branding-extension.ear
│   └── jpp-mobile-integration.ear
│       ├── gatein-mobile-login.war
│       ├── jpp-information-portlet.war
│       ├── jpp-mobile-configuration.war
│       ├── jpp-responsive-skin.war
│       ├── META-INF
│       ├── redirect-portlet.war
│       ├── responsive-banner-portlet.war
│       ├── responsive-features-portlet.war
│       ├── responsive-footer-portlet.war
```

```
│       ├── responsive-header-portlet.war
│       └── responsive-navigation-portlet.war
└── gatein.ear
    ├── dashboard.war
    ├── exoadmin.war
    ├── eXoGadgetServer.war
    ├── eXoGadgets.war
    ├── eXoResources.war
    │   ├── javascript
    │   ├── META-INF
    │   ├── skin
    │   └── WEB-INF
    ├── gwtGadgets.war
    ├── integration.war
    │   └── WEB-INF
    ├── META-INF
    │   ├── application.xml
    │   └── jboss-deployment-structure.xml
    ├── portal.war
    │   ├── device
    │   ├── error
    │   ├── favicon.ico
    │   ├── groovy
    │   ├── login
    │   ├── META-INF
    │   ├── setup
    │   ├── templates
    │   ├── WEB-INF
    │   └── welcome.jsp
    ├── redirect-admin.war
    ├── rest.war
    └── web.war
```

**Example 5.3. Directories within the standalone/ directory**

```
standalone
├── configuration
│   ├── application-roles.properties
│   ├── application-users.properties
│   ├── gatein
│   │   ├── codec
│   │   ├── configuration.properties
│   │   ├── configuration.xml
│   │   ├── controller.xml
│   │   ├── gadgets
│   │   ├── portlet.xml
│   │   └── wsrp
│   ├── logging.properties
│   ├── mgmt-users.properties
│   ├── standalone-full-ha.xml
│   ├── standalone-full.xml
│   ├── standalone-ha.xml
│   ├── standalone-osgi.xml
```

```
│   ├── standalone.xml
│   └── standalone_xml_history
│       ├── current
│       ├── snapshot
│       ├── standalone.boot.xml
│       ├── standalone.initial.xml
│       └── standalone.last.xml
├── data
│   ├── content
│   ├── gatein
│   │   ├── jcr
│   │   └── portal
│   ├── timer-service-data
│   ├── tx-object-store
│   │   └── ShadowNoFileLockStore
│   └── wsdl
│       └── gatein-wsrp-integration.ear
├── deployments
│   └── README.txt
├── lib
│   └── ext
├── log
│   ├── server.log
│   └── server.log.2013-09-04
└── tmp
    ├── auth
    ├── vfs
    │   ├── deployment1e5dbb2debad1ef9
    │   ├── deploymentaa52d43feb530be9
    │   ├── deploymentbbd242b840f00e6
    │   ├── temp16524ed851e8e554
    │   ├── temp44c44c4d38f5c74c
    │   └── tempac089482639a2df5
    └── work
        └── jboss.web
```

Report a bug

## 5.5. Run as an Operating System Service

Red Hat JBoss Portal is based on Red Hat JBoss Enterprise Application Platform 6. It can be configured to run as a service, allowing the server to start in Standalone configuration at system runtime. The server instance will continue to run after logging out of the local system.

Report a bug

### 5.5.1. Install as a Service in Red Hat Enterprise Linux

**Install Red Hat JBoss Portal as a Red Hat Enterprise Linux Service**

Complete this task to configure Red Hat JBoss Portal (JBoss Portal) as a service in Red Hat Enterprise Linux. When completed, JBoss Portal starts automatically when the Red Hat Enterprise Linux system reaches its default run-level, and stops automatically when the operating system executes its shutdown routine.

### Prerequisites

You must have administrator access to perform this task.

1. **Copy the startup script to the /etc/jboss-as/ directory**

   The startup script, and an associated configuration file, are located in the ***$JPP_HOME*/bin/init.d/** directory. Copy each file to the required location.

   ```
   [user@host init.d]$ sudo cp jboss-as.conf /etc/jboss-as
   ```

   ```
   [user@host init.d]$ sudo cp jboss-as-standalone.sh /etc/init.d/
   ```

2. **Add the startup script as a service**

   Add the new **jboss-as-standalone.sh** service to list of automatically started services, using the **chkconfig** service management command.

   ```
   [user@host init.d]$ sudo chkconfig --add jboss-as-standalone.sh
   ```

3. **Configure script options and variables**

   Edit the **jboss-as.conf** file to customize startup options for JBoss Portal and the JVM. Use the comments in the file as general guidance.

   Because the start-up script makes default assumptions about the name of the start-up file and the location of the JBoss Portal instance, some customization is required to the script before it can be run as a service. Customize the script, paying special attention to the following variables:

   **JBOSS_HOME**

   Location where JBoss Portal is extracted.

   It is recommended to set the **JBOSS_HOME** variable to point to the directory where you extracted JBoss Portal. Some applications require this variable in order to function correctly. Do not add a trailing slash (/) at the end of the directory name.

   **JBOSS_USER**

   User with the ability to run JBoss Portal as a non-privileged user (superuser privileges are not required to run JBoss Portal).

   **JBOSS_CONFIG**

   Name of the configuration file used to start JBoss Portal, such as **standalone.xml**

   **JBOSS_SCRIPT**

Script used to start JBoss Portal, such as **standalone.sh**.

4. **Optional: Start the service.**

   If desired, start the new service using the standard syntax for starting Red Hat Enterprise Linux services.

   ```
   [user@host bin]$ sudo service jboss-as-standalone start
   ```

**See Also:**

- Section 1.1, “Cross-product content disclaimer”

Report a bug

## 5.6. Start as a Standalone Server

**Red Hat Enterprise Linux.**

Run the following command to start the server in standalone mode:

***JPP_HOME*/bin/standalone.sh**

**Optional: Specify additional parameters.**

To print a list of additional parameters to pass to the start-up scripts, use the ***-h*** parameter.

**See Also:**

- Section 7.2, “About JBoss Portal Domain Mode”

Report a bug

## 5.7. Test the Installation

1. **Look for error messages in the log file**

   After you start the server, view the log file in ***JPP_HOME*/standalone/log/**.

   **Result:**

   If the server started properly, there will be no errors, and you will see output similar to the following:

   **Example 5.4. Example of a successful start-up**

   ```
   14:36:32,632 INFO  [org.jboss.as] (Controller Boot Thread)
   JBAS015874: JBoss Portal Platform 6.0.0.ER04.2 (AS 7.1.3.Final-
   redhat-4) started in 23519ms - Started 919 of 1041 services (116
   services are passive or on-demand)
   ```

2. **Browse to the Management Console.**

If the installation worked properly and your server is running, you should be able to access the Management Console by pointing your web browser at an address similar to **http://*YOUR_SERVER*:9990/**, replacing *YOUR_SERVER* with a valid value.

**Result:**

The front page of the Management Console appears.

**Result:**

The Management Console is a deployable service. If you are able to reach it after starting the server, your installation is working properly and is able to deploy services.

**See Also:**

- Chapter 6, *Common Configuration*
- Section 5.2, “Install Red Hat JBoss Portal”

Report a bug

## 5.8. Uninstall Red Hat JBoss Portal (Zip Installation)

This section covers the steps required to uninstall a Zip installation of Red Hat JBoss Portal (JBoss Portal) 6.

**Prerequisites**

Backup any modified configuration files and deployments that may be reused in a later instance.

1. Navigate to the directory where the JBoss Portal 6 folder from the Zip file is extracted.
2. JBoss Portal 6 installs in a single directory. Delete the installation directory to uninstall JBoss Portal 6.
3. Delete any initialization scripts, or other scripts which depended upon JBoss Portal 6 being installed on the server.

JBoss Portal 6 is now uninstalled from the server.

**See Also:**

- Section 5.2, “Install Red Hat JBoss Portal”

Report a bug

# Chapter 6. Common Configuration

Red Hat JBoss Portal (JBoss Portal) is a layered Middleware product. It runs atop Red Hat JBoss Enterprise Application Platform (JBoss EAP), therefore common configuration relating to JBoss EAP is required. The following tasks relate to configuring the underlying application platform.

Report a bug

## 6.1. Network Ports Used By JBoss EAP 6

The ports used by the JBoss EAP 6 default configuration depend on several factors:

- Whether your server groups use one of the default socket binding groups, or a custom group.
- The requirements of your individual deployments.

**Numerical port offsets**

A numerical port offset can be configured, to alleviate port conflicts when you run multiple servers on the same physical server. If your server uses a numerical port offset, add the offset to the default port number for its server group's socket binding group. For instance, if the HTTP port of the socket binding group is **8080**, and your server uses a port offset of **100**, its HTTP port is **8180**.

Unless otherwise stated, the ports use the TCP protocol.

### The default socket binding groups

- **full-ha-sockets**
- **full-sockets**
- **ha-sockets**
- **standard-sockets**

These socket binding groups are available only in **domain.xml**. The standalone server profiles contain only standard socket binding group. This group corresponds to standard-sockets in **standalone.xml**, **ha-sockets** for **standalone-ha.xml**, **full-sockets** for **standalone-full.xml**, and **full-ha-sockets** for **standalone-full-ha.xml**. Standalone profiles contain some more socket bindings, for example, management-{native,http,https}.

**Table 6.1. Reference of the default socket bindings**

| Name | Port | Multicast Port | Description | full-ha-sockets | full-sockets | ha-socket | standard-socket |
|---|---|---|---|---|---|---|---|
| **ajp** | 8009 | | Apache JServ Protocol. Used for HTTP clustering and load balancing. | Yes | Yes | Yes | Yes |

| Name | Port | Multicast Port | Description | full-ha-sockets | full-sockets | ha-socket | standard-socket |
|---|---|---|---|---|---|---|---|
| **http** | 8080 | | The default port for deployed web applications. | Yes | Yes | Yes | Yes |
| **https** | 8443 | | SSL-encrypted connection between deployed web applications and clients. | Yes | Yes | Yes | Yes |
| **jacorb** | 3528 | | CORBA services for JTS transactions and other ORB-dependent services. | Yes | Yes | No | No |
| **jacorb-ssl** | 3529 | | SSL-encrypted CORBA services. | Yes | Yes | No | No |
| **jgroups-diagnostics** | | 7500 | Multicast. Used for peer discovery in HA clusters. Not configurable using the Management Interfaces. | Yes | No | Yes | No |
| **jgroups-mping** | | 45700 | Multicast. Used to discover initial membership in a HA cluster. | Yes | No | Yes | No |
| **jgroups-tcp** | 7600 | | Unicast peer discovery in HA clusters using TCP. | Yes | No | Yes | No |
| **jgroups-tcp-fd** | 57600 | | Used for HA failure detection over TCP. | Yes | No | Yes | No |
| **jgroups-udp** | 55200 | 45688 | Multicast peer discovery in HA clusters using UDP. | Yes | No | Yes | No |
| **jgroups-udp-fd** | 54200 | | Used for HA failure detection over UDP. | Yes | No | Yes | No |
| **messaging** | 5445 | | JMS service. | Yes | Yes | No | No |
| **messaging-group** | | | Referenced by HornetQ JMS broadcast and discovery groups. | Yes | Yes | No | No |
| **messaging-throughput** | 5455 | | Used by JMS Remoting. | Yes | Yes | No | No |
| **mod_cluster** | | 23364 | Multicast port for communication between JBoss EAP 6 and the HTTP load balancer. | Yes | No | Yes | No |

| Name | Port | Multicast Port | Description | full-ha-sockets | full-sockets | ha-socket | standard-socket |
|---|---|---|---|---|---|---|---|
| **remoting** | 4447 | | Used for remote EJB invocation. | Yes | Yes | Yes | Yes |
| **txn-recovery-environment** | 4712 | | The JTA transaction recovery manager. | Yes | Yes | Yes | Yes |
| **txn-status-manager** | 4713 | | The JTA / JTS transaction manager. | Yes | Yes | Yes | Yes |

### Management Ports

In addition to the socket binding groups, each host controller opens two more ports for management purposes:

- **9990** - The Web Management Console port
- **9999** - The port used by the Management Console and Management API

Additionally, if HTTPS is enabled for the Management Console, 9443 is also opened as the default port.

**See Also:**

- Section 1.1, “Cross-product content disclaimer”

Report a bug

## 6.2. Configure Network Firewalls to Work with JBoss EAP 6

### Summary

Most production environments use firewalls as part of an overall network security strategy. If you need multiple server instances to communicate with each other or with external services such as web servers or databases, your firewall must take this into account. A well-managed firewall only opens the ports which are necessary for operation, and limits access to the ports to specific IP addresses, subnets, and network protocols.

A full discussion of firewalls is out of the scope of this documentation.

**Prerequisites**

- Determine the ports you need to open.
- An understanding of your firewall software is required. This procedure uses the **system-config-firewall** command in Red Hat Enterprise Linux 6. Microsoft Windows Server includes a built-in firewall, and several third-party firewall solutions are available for each platform. On Microsoft Windows Server, you can use PowerShell to configure the firewall.

### Assumptions

This procedure configures a firewall in an environment with the following assumptions:

- The operating system is Red Hat Enterprise Linux 6.
- JBoss EAP 6 runs on host **10.1.1.2**. Optionally, the server has its own firewall.
- The network firewall server runs on host **10.1.1.1** on interface **eth0**, and has an external interface **eth1**.
- You want traffic on port **5445** (a port used by JMS) forwarded to JBoss EAP 6. No other traffic should be allowed through the network firewall.

**Procedure 6.1. Manage Network Firewalls and JBoss EAP 6 to work together**

1. **Log into the Management Console.**

   Log into the Management Console. By default, it runs on http://localhost:9990/console/.

2. **Determine the socket bindings used by the socket binding group.**

   a. Click the **Configuration** label at the top of the Management Console.

   b. Expand the **General Configuration** menu. Select the **Socket Binding**.

   c. The **Socket Binding Declarations** screen appears. Initially, the **standard-sockets** group is shown. Choose a different group by selecting it from the combo box on the right-hand side.

   **Note**

   If you use a standalone server, it has only one socket binding group.

   The list of socket names and ports is shown, eight values per page. You can go through the pages by using the arrow navigation below the table.

3. **Determine the ports you need to open.**

   Depending on the function of the particular port and the requirements of your environment, some ports may need to be opened on your firewall.

4. **Configure your firewall to forward traffic to JBoss EAP 6.**

   Perform these steps to configure your network firewall to allow traffic on the desired port.

   a. Log into your firewall machine and access a command prompt, as the root user.

   b. Issue the command **system-config-firewall** to launch the firewall configuration utility. A GUI or command-line utility launches, depending on the way you are logged into the firewall system. This task makes the assumption that you are logged in via SSH and using the command-line interface.

   c. Use the **TAB** key on your keyboard to navigate to the **Customize** button, and press the **ENTER** key. The **Trusted Services** screen appears.

   d. Do not change any values, but use the **TAB** key to navigate to the **Forward** button, and press **ENTER** to advanced to the next screen. The **Other Ports** screen appears.

e. Use the **TAB** key to navigate to the **<Add>** button, and press **ENTER**. The **Port and Protocol** screen appears.

f. Enter **5445** in the **Port / Port Range** field, then use the **TAB** key to move to the **Protocol** field, and enter **tcp**. Use the **TAB** key to navigate to the **OK** button, and press **ENTER**.

g. Use the **TAB** key to navigate to the **Forward** button until you reach the **Port Forwarding** screen.

h. Use the **TAB** key to navigate to the **<Add>** button, and press the **ENTER** key.

i. Fill in the following values to set up port forwarding for port **5445**.

- Source interface: **eth1**
- Protocol: **tcp**
- Port / Port Range: **5445**
- Destination IP address: **10.1.1.2**
- Port / Port Range: **5445**

Use the **TAB** key to navigate to the **OK** button, and press **ENTER**.

j. Use the **TAB** key to navigate to the **Close** button, and press **ENTER**.

k. Use the **TAB** key to navigate to the **OK** button, and press **ENTER**. To apply the changes, read the warning and click **Yes**.

5. **Configure a firewall on your JBoss EAP 6 host.**

Some organizations choose to configure a firewall on the JBoss EAP 6 server itself, and close all ports that are not necessary for its operation. See Section 6.1, “Network Ports Used By JBoss EAP 6” and determine which ports to open, then close the rest. The default configuration of Red Hat Enterprise Linux 6 closes all ports except **22** (used for Secure Shell (SSH) and **5353** (used for multicast DNS). While you are configuring ports, ensure you have physical access to your server so that you do not inadvertently lock yourself out.

### Result

Your firewall is configured to forward traffic to your internal JBoss EAP 6 server in the way you specified in your firewall configuration. If you chose to enable a firewall on your server, all ports are closed except the ones needed to run your applications.

### Procedure 6.2. Configuring Firewall on Microsoft Windows using PowerShell

1. Switch off firewall for debug purpose to determine whether the current network behavior is related to the firewall configuration.

```
Start-Process "$psHome\powershell.exe" -Verb Runas -ArgumentList '-
command "NetSh Advfirewall set allprofiles state off"'
```

2. Allow UDP connections on port 23364. For example:

```
Start-Process "$psHome\powershell.exe" -Verb Runas -ArgumentList '-
command "NetSh Advfirewall firewall add rule name="UDP Port 23364"
```

```
dir=in  action=allow protocol=UDP localport=23364"'
Start-Process "$psHome\powershell.exe" -Verb Runas -ArgumentList '-
command "NetSh Advfirewall firewall add rule name="UDP Port 23364"
dir=out action=allow protocol=UDP localport=23364"'
```

**Procedure 6.3. Configure the Firewall on Red Hat Enterprise Linux 7 to Allow mod_cluster Advertising**

- To allow mod_cluster advertising on Red Hat Enterprise Linux 7, you must enable the UDP port in the firewall as follows:

```
firewall-cmd --permanent --zone=public --add-port=23364/udp
```

**Note**

224.0.1.105:23364 is the default address and port for mod_cluster balancer advertising UDP multicast.

**See Also:**

- Section 1.1, “Cross-product content disclaimer”

Report a bug

## 6.3. Add the User for the Management Interfaces

### Overview

JBoss EAP 6's management instances are secured by default as there are no user accounts available initially, (unless you have installed the platform using the graphical installer.) This is a precautionary measure designed to prevent security breaches that can arise from simple configuration errors.

HTTP communication with JBoss EAP 6 is considered to be remote access, even if the traffic originates on the localhost. Therefore, you must create at least one user in order to be able to use the management console. If you attempt to access the management console before adding a user, you will receive an error because it does not deploy until a user is created.

Follow these steps to create the initial administrative user, which can use the web-based Management Console and remote instances of the Management CLI to configure and administer JBoss EAP 6 from remote systems.

**Procedure 6.4. Create the Initial Administrative User for the Remote Management Interfaces**

1. **Run the `add-user.sh` or `add-user.bat` script.**

   Change to the ***`EAP_HOME`*`/bin/`** directory. Invoke the appropriate script for your operating system.

   **Red Hat Enterprise Linux**

   ```
   [user@host bin]$ ./add-user.sh
   ```

**Microsoft Windows Server**

```
C:\bin>  add-user.bat
```

2. **Choose to add a Management user.**

   Press **ENTER** to select the default option **a** to add a Management user.

   This user is added to the **ManagementRealm** and is authorized to perform management operations using the web-based Management Console or command-line based Management CLI. The other choice, **b**, adds a user to the **ApplicationRealm**, and provides no particular permissions. That realm is provided for use with applications.

3. **Enter the desired username and password.**

   When prompted, enter the username and password. You will be prompted to confirm the password.

4. **Enter group information.**

   Add the group or groups to which the user belongs. If the user belongs to multiple groups, enter a comma-separated list. Leave it blank if you do not want the user to belong to any groups.

5. **Review the information and confirm.**

   You are prompted to confirm the information. If you are satisfied, type **yes**.

6. **Choose whether the user represents a remote JBoss EAP 6 server instance.**

   Besides administrators, the other type of user which occasionally needs to be added to JBoss EAP 6 in the **ManagementRealm** is a user representing another instance of JBoss EAP 6, which must be able to authenticate to join a cluster as a member. The next prompt allows you to designate your added user for this purpose. If you select **yes**, you will be given a hashed **secret** value, representing the user's password, which would need to be added to a different configuration file. For the purposes of this task, answer **no** to this question.

7. **Enter additional users.**

   You can enter additional users if desired, by repeating the procedure. You can also add them at any time on a running system. Instead of choosing the default security realm, you can add users to other realms to fine-tune their authorizations.

8. **Create users non-interactively.**

   You can create users non-interactively, by passing in each parameter at the command line. This approach is not recommended on shared systems, because the passwords will be visible in log and history files. The syntax for the command, using the management realm, is:

   ```
   [user@host bin]$ ./add-user.sh username password
   ```

   To use the application realm, use the **-a** parameter.

   ```
   [user@host bin]$ ./add-user.sh -a username password
   ```

9. You can suppress the normal output of the add-user script by passing the **--silent** parameter. This applies only if the minimum parameters **username** and **password** have been specified. Error messages will still be shown.

### Result

Any users you add are activated within the security realms you have specified. Users active within the **ManagementRealm** realm are able to manage JBoss EAP 6 from remote systems.

**See Also:**

- Section 6.4, “Default User Security Configuration”
- Section 1.1, “Cross-product content disclaimer”

Report a bug

## 6.4. Default User Security Configuration

### Introduction

All management interfaces in JBoss EAP 6 are secured by default. This security takes two different forms:

- Local interfaces are secured by a SASL contract between local clients and the server they connect to. This security mechanism is based on the client's ability to access the local filesystem. This is because access to the local filesystem would allow the client to add a user or otherwise change the configuration to thwart other security mechanisms. This adheres to the principle that if physical access to the filesystem is achieved, other security mechanisms are superfluous. The mechanism happens in four steps:

**Note**

HTTP access is considered to be remote, even if you connect to the localhost using HTTP.

  - The client sends a message to the server which includes a request to authenticate with the local SASL mechanism.
  - The server generates a one-time token, writes it to a unique file, and sends a message to the client with the full path of the file.
  - The client reads the token from the file and sends it to the server, verifying that it has local access to the filesystem.
  - The server verifies the token and then deletes the file.
- Remote clients, including local HTTP clients, use realm-based security. The default realm with the permissions to configure the JBoss EAP 6 instance remotely using the management interfaces is **ManagementRealm**. A script is provided which allows you to add users to this realm (or realms you create). For more information on adding users, see the *User Management* chapter of the *JBoss EAP 6 Administration and Configuration Guide*. For each user, the username and a hashed password are stored in a file.

**Managed domain**

***EAP_HOME*/domain/configuration/mgmt-users.properties**

**Standalone server**

***EAP_HOME*/standalone/configuration/mgmt-users.properties**

Even though the contents of the **mgmt-users.properties** are masked, the file must still be treated as a sensitive file. It is recommended that it be set to the file mode of **600**, which gives no access other than read and write access by the file owner.

**See Also:**

- Section 1.1, “Cross-product content disclaimer”

Report a bug

## 6.5. Adjust Memory Settings

If the server is running out of memory, you can adjust the memory settings before deploying the applications.

You can do this by updating **JAVA_OPTS** settings in the file ***JPP_HOME*/bin/standalone.conf** on Linux, or ***JPP_HOME*/bin/standalone.conf.bat** on Windows.

The default values do not take into account the memory requirements of your applications:

```
-Xms1303m -Xmx1303m -XX:MaxPermSize=256m ....
```

**See Also:**

- Section 1.1, “Cross-product content disclaimer”

Report a bug

# Chapter 7. Platform-specific Configuration

Configuration relating specifically to the technology and features of Red Hat JBoss Portal is required before JBoss Portal can be used in a production environment. The following tasks detail the extra configuration required to prepare JBoss Portal for a production environments.

Report a bug

## 7.1. Configure the Portal Root Password

For security reasons, Red Hat JBoss Portal (JBoss Portal) does not ship with default user accounts. It is necessary to configure a password for the portal root user. This user is created for first-run access to the portal in order to configure an Administrator user account.

**Procedure 7.1. Setting the Root Password Through the Command-line**

1. Open a terminal and execute the **`portal-setup.sh`** script.
   - For Linux, **`$JPP_HOME/bin/portal-setup.sh`**
   - For Microsoft Windows, **`%JPP_HOME%\bin\portal-setup.bat`**
2. Type a value for the root password when prompted and press **`Enter`**. Re-type the password and press **`Enter`**.

   The password is encrypted and stored in the **`$JPP_HOME/standalone/configuration/gatein/configuration.properties`** file.
3. Start JBoss Portal.
4. Open the Users Management interface and create the required accounts, including an Administrator account.

**Procedure 7.2. Setting the Root Password Through the Web Interface**

If JBoss Portal detects there is no Root password configured, the user is redirected to a Set Root Password page.

**Important**

The portal-setup scripts are for single use only. After initially running the script, the password in **`configuration.properties`** has no effect.

The password generated by the script can be removed after configuring accounts in the Users Management interface.

1. Once all configuration in this user guide has been completed, start JBoss Portal.
2. Open JBoss Portal in a browser.
3. In the Set Root Password page, type a Root password in the Password field, and repeat the same password in the Repeat password field.

4. Click Setup to set the Root password.

   To modify the Root password on subsequent occasions, use the portal administration interface for Users Management.

**See Also:**

- Section 5.6, “Start as a Standalone Server”

Report a bug

## 7.2. About JBoss Portal Domain Mode

Red Hat JBoss Portal (JBoss Portal) supports standalone mode as its primary operation mode. JBoss Portal can be configured to operate in domain mode, however there are some caveats which are important to understand when considering running JBoss Portal in a managed domain.

JBoss Portal domain configuration guidance and examples are described in detail in the Domain Mode part of the JBoss Portal *Administration and Configuration Guide*. Read this information before deciding how to configure JBoss Portal's primary operation mode.

A general introduction to domain mode is provided in the About JBoss EAP 6 Operating Modes chapter in the Red Hat JBoss Enterprise Application Platform (JBoss EAP) *Administration and Configuration Guide*.

Report a bug

## 7.3. Provision JCR and IDM Databases

JCR and IDM services are configured to use a container-provided datasource, configured through Red Hat JBoss Enterprise Application Platform 6 configuration.

See the Datasource Management chapter in the Red Hat JBoss Enterprise Application Platform 6 *Administration and Configuration Guide* for detailed information about managing and configuring datasources for use with JBoss Portal. This chapter includes direct examples for datasource connectors

Datasource configuration is located in **`JPP_DIST/standalone/configuration/standalone.xml`** and for clustered configuration in the **`standalone-ha.xml`** file.

**Procedure 7.3. Configuring the Java Content Repository (JCR)**

1. Open **`JPP_DIST/standalone/configuration/standalone.xml`** in edit mode.
2. Bind the JCR datasource to JNDI under **`java:/jdbcjcr_portal`**.

```
<datasource jndi-name="java:/jdbcjcr_portal" pool-
name="JCRPortalDS" enabled="true" use-java-context="true">

    <connection-
url>jdbc:h2:file:${jboss.server.data.dir}/gatein/portal/jdbcjcr_por
tal;DB_CLOSE_DELAY=-1</connection-url>
    <driver>h2</driver>

    <security>
```

```
        <user-name>sa</user-name>
        <password>sa</password>
    </security>

</datasource>
```

3. If additional portals are deployed, additional datasources must be configured and bound in JNDI using a separate **java:/jdbcjcr_*PORTAL-NAME*** directive.

   Ensure the user has rights to create tables on jdbcjcr_portal, and to update them as they will automatically be created during the first startup.

**Procedure 7.4. Configuring Identity Management (IDM)**

1. Open **JPP_DIST/standalone/configuration/standalone.xml** in edit mode.
2. Bind the IDM datasource to JNDI under **java:/jdbcidm_portal**.

```
<datasource jndi-name="java:/jdbcidm_portal" pool-
name="JCRPortalDS" enabled="true" use-java-context="true">

    <connection-
url>jdbc:h2:file:${jboss.server.data.dir}/gatein/portal/jdbcjcr_por
tal;DB_CLOSE_DELAY=-1</connection-url>
    <driver>h2</driver>

    <security>
        <user-name>sa</user-name>
        <password>sa</password>
    </security>

</datasource>
```

3. If additional portals are deployed, additional datasources must be configured and bound in JNDI using a separate **java:/jdbcjcr_*PORTAL-NAME*** directive.

   Ensure the user has rights to create tables on jdbcjcr_portal, and to update them as they will automatically be created during the first startup.

**See Also:**

- Section 7.4.1, “JTA Support”

Report a bug

## 7.4. JTA Support Configuration

### 7.4.1. JTA Support

Red Hat JBoss Portal uses separate Java Content Repository (JCR) and Identity Management (IDM) databases to store identity information, and portal content. If your provisioned databases are compatible, it may be advantageous to configure the Java Transaction API (JTA) to provide a distributed transaction system across the application server, the resource manager, and the transactional applications.

With JTA configured, each HTTP request to the portal is encapsulated inside a JTA transaction. The

advantage of this approach is access to IDM and JCR datasources and other portal resources are managed in one request rather than multiple requests. This approach also allows local resources to participate in global transactions.

Report a bug

### 7.4.2. JDBC Driver Download Locations

The following table gives the standard download locations for JDBC drivers of common databases used with JBoss EAP 6. These links point to third-party websites which are not controlled or actively monitored by Red Hat. For the most up-to-date drivers for your database, check your database vendor's documentation and website.

**Table 7.1. JDBC driver download locations**

| Vendor | Download Location |
|---|---|
| MySQL | http://www.mysql.com/products/connector/ |
| PostgreSQL | http://jdbc.postgresql.org/ |
| Oracle | http://www.oracle.com/technetwork/database/features/jdbc/index-091264.html |
| IBM | http://www-306.ibm.com/software/data/db2/java/ |
| Sybase | http://www.sybase.com/products/allproductsa-z/softwaredeveloperkit/jconnect |
| Microsoft | http://msdn.microsoft.com/data/jdbc/ |

Report a bug

### 7.4.3. Install a JDBC Driver as a Core Module

#### Prerequisites

Before performing this task, you need to meet the following prerequisites:

- Select the correct JDBC driver from the information in the Supported Configurations document at https://access.redhat.com/site/articles/119833 and download the driver package.
- Extract the archive.

**Procedure 7.5. Install a JDBC Driver as a Core Module**

1. Create a file path structure under the ***JPP_HOME*/modules/system/layers/base** directory. For example, for a MySQL JDBC driver, create **JPP_HOME/modules/system/layers/base/com/mysql/main/**.
2. Copy the JDBC driver JAR into the **main/** subdirectory.
3. In the **main/** subdirectory, create a **module.xml** file. The following is an example of a **module.xml** file:

```
<?xml version="1.0" encoding="UTF-8"?>
<module xmlns="urn:jboss:module:1.0" name="com.mysql">
  <resources>
    <resource-root path="mysql-connector-java-5.1.15.jar"/>
  </resources>
  <dependencies>
```

```
    <module name="javax.api"/>
    <module name="javax.transaction.api"/>
  </dependencies>
</module>
```

4. Start the Server.
5. Start the Management CLI.
6. Run the CLI command to add the JDBC driver module to the server configuration.

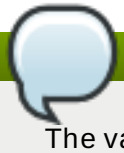
   The command you choose depends on the number of classes listed in the **/META-INF/services/java.sql.Driver** file located in the JDBC driver JAR. For example, the **/META-INF/services/java.sql.Driver** file in the MySQL 5.1.20 JDBC JAR lists two classes:

   - **com.mysql.jdbc.Driver**
   - **com.mysql.fabric.jdbc.FabricMySQLDriver**

   When there is more than one entry, you must also specify the name of the driver class. Failure to do so results in an error similar to the following:

   **Example 7.1. Driver class error**

   ```
   JBAS014749: Operation handler failed: Service jboss.jdbc-
   driver.mysql is already registered
   ```

   **Note**

   The value for *DRIVER_NAME* depends on the number of classes listed in the **/META-INF/services/java.sql.Driver** file located in the JDBC driver JAR. If there is only one class, the value is the name of the JAR. If there are multiple classes, the value is the name of the JAR + driverClassName + "_" + majorVersion +"_" + minorVersion. Failure to do so will result in the following error being logged:

   ```
   JBAS014775:    New missing/unsatisfied dependencies
   ```

   For example, the *DRIVER_NAME* value required for the MySQL 5.1.31 driver, is **mysql-connector-java-5.1.31-bin.jarcom.mysql.jdbc.Driver_5_1**.

   A. Run the CLI command for JDBC JARs containing one class entry.

   ```
   /subsystem=datasources/jdbc-driver=DRIVER_NAME:add(driver-
   name=DRIVER_NAME,driver-module-name=MODULE_NAME,driver-xa-
   datasource-class-name=XA_DATASOURCE_CLASS_NAME)
   ```

   **Example 7.2. CLI Command for Standalone Mode for JDBC JARs with one driver class**

```
/subsystem=datasources/jdbc-driver=mysql:add(driver-
name=mysql,driver-module-name=com.mysql,driver-xa-datasource-
class-name=com.mysql.jdbc.jdbc2.optional.MysqlXADataSource)
```

**Example 7.3. CLI Command for Domain Mode for JDBC JARs with one driver class**

```
/profile=ha/subsystem=datasources/jdbc-driver=mysql:add(driver-
name=mysql,driver-module-name=com.mysql,driver-xa-datasource-
class-name=com.mysql.jdbc.jdbc2.optional.MysqlXADataSource)
```

B. Run the CLI command for JDBC JARs containing multiple class entries.

```
/subsystem=datasources/jdbc-driver=DRIVER_NAME:add(driver-
name=DRIVER_NAME,driver-module-name=MODULE_NAME,driver-xa-
datasource-class-name=XA_DATASOURCE_CLASS_NAME, driver-class-
name=DRIVER_CLASS_NAME)
```

**Example 7.4. CLI Command for Standalone Mode for JDBC JARs with multiple driver class entries**

```
/subsystem=datasources/jdbc-driver=mysql:add(driver-
name=mysql,driver-module-name=com.mysql,driver-xa-datasource-
class-name=com.mysql.jdbc.jdbc2.optional.MysqlXADataSource,
driver-class-name=com.mysql.jdbc.Driver)
```

**Example 7.5. CLI Command for Domain Mode for JDBC JARs with multiple driver class entries**

```
/profile=ha/subsystem=datasources/jdbc-driver=mysql:add(driver-
name=mysql,driver-module-name=com.mysql,driver-xa-datasource-
class-name=com.mysql.jdbc.jdbc2.optional.MysqlXADataSource,
driver-class-name=com.mysql.jdbc.Driver)
```

**Result**

The JDBC driver is now installed and set up as a core module, and is available to be referenced by application datasources.

Report a bug

### 7.4.4. Configure Datasources for JTA Support

You can choose to configure the IDM and JCR databases to share the same datasource, or for those databases that support transactions with two-phase commits, create two separate XA datasources.

**Procedure 7.6. Shared Datasource Configuration**

This procedure describes how to set up the JCR and IDM datasources as a no-XA combined datasource. no-XA datasources are widely supported by most database vendors, which makes this configuration a good choice for small-scale testing requirements.

The JCR and IDM databases do not support two-phase commit because they are no-XA databases. JTA allows for one datasource of this type within one JTA transaction, and the datasources are handled as a Last Resource in a JTA transaction.

Configuring the IDM and JCR datasources in this way means that portlet applications can not use a different no-XA datasource, because they must access the JCR and IDM no-XA datasources of the portal. This approach is therefore not recommended for production environments due to scalability issues.

1. Open **`JPP_HOME/standalone/configuration/standalone.xml`**.
2. Comment the jdbcjcr_portal <datasource> directive to disable the JCR datasource.

```
<datasources>
  <!--<datasource jndi-name="java:/jdbcjcr_portal" pool-
name="JCRPortalDS" enabled="true" use-java-context="true">
  <connection-
url>jdbc:h2:file:${jboss.server.data.dir}/gatein/portal/jdbcjcr_por
tal;DB_CLOSE_DELAY=-1</connection-url>
  <driver>h2</driver>
  <security>
    <user-name>sa</user-name>
    <password>sa</password>
  </security>
  </datasource>-->
</datasources>
```

3. Configure jdbcidm_portal <datasource> directive to use the MySQL database.

```
<datasources>
  <!--<datasource jndi-name="java:/jdbcjcr_portal" pool-
name="JCRPortalDS" enabled="true" use-java-context="true">
    <connection-
url>jdbc:h2:file:${jboss.server.data.dir}/gatein/portal/jdbcjcr_por
tal;DB_CLOSE_DELAY=-1</connection-url>
    <driver>h2</driver>
    <security>
      <user-name>sa</user-name>
      <password>sa</password>
    </security>
  </datasource>-->
  <datasource jndi-name="java:/jdbcidm_portal" pool-
name="IDMPortalDS" enabled="true" use-java-context="true">
    <connection-url>jdbc:mysql://localhost/portal</connection-url>
    <driver>mysql</driver>
    <security>
      <user-name>portal</user-name>
      <password>portal</password>
    </security>
    <pool>
      <min-pool-size>10</min-pool-size>
      <max-pool-size>100</max-pool-size>
```

```
      <prefill>true</prefill>
    </pool>
  </datasource>
</datasources>
```

4. Add the MySQL module into the list of <drivers>.

```
<datasources>
  <!--<datasource jndi-name="java:/jdbcjcr_portal" pool-
name="JCRPortalDS" enabled="true" use-java-context="true">
    <connection-
url>jdbc:h2:file:${jboss.server.data.dir}/gatein/portal/jdbcjcr_por
tal;DB_CLOSE_DELAY=-1</connection-url>
    <driver>h2</driver>
    <security>
      <user-name>sa</user-name>
      <password>sa</password>
    </security>
  </datasource>-->
  <datasource jndi-name="java:/jdbcidm_portal" pool-
name="IDMPortalDS" enabled="true" use-java-context="true">
    <connection-url>jdbc:mysql://localhost/portal</connection-url>
    <driver>mysql</driver>
    <security>
      <user-name>portal</user-name>
      <password>portal</password>
    </security>
    <pool>
      <min-pool-size>10</min-pool-size>
      <max-pool-size>100</max-pool-size>
      <prefill>true</prefill>
    </pool>
  </datasource>
  <drivers>
    <driver name="mysql" module="com.mysql">
      <xa-datasource-class>com.mysql.jdbc.Driver</xa-datasource-
class>
    </driver>
  </drivers>
</datasources>
```

5. Open **JPP_HOME/standalone/configuration/gatein/configuration.properties**, and verify the gatein.jcr.datasource.name property has the value java:/jdbcidm. This value ensures the JCR database uses the IDM database as the datasource.

```
gatein.jcr.datasource.name=java:/jdbcidm
gatein.idm.datasource.name=java:/jdbcidm
```

**Procedure 7.7. XA Datasource Configuration**

This procedure describes how to configure separate transactional JCR and IDM databases. Unlike no-XA databases, it is not necessary to share the same datasource to ensure commit integrity: JTA handles datasource synchronization using two-phase commit logic.

**Important**

In most cases, a separate datasource is required for the JCR and IDM database, running on separate processes (separate hosts). Some databases such as Sybase and Oracle support shared datasources using the same process. Verify the database used for the production installation supports shared XA datasources.

If both datasources use the same database, an issue with enlisting two XA resources for the same database and transaction will most likely occur and cause transactions to fail.

Customers that are restricted to using shared datasources on a supported database are recommended to use XA datasources instead of no-XA datasources. It is strongly recommended to implement a datasource solution on two separate hosts to take advantage of XA Datasource performance and remove the LRCO limitation.

1. Open **`JPP_HOME/standalone/configuration/standalone.xml`**.
2. Configure jdbcidm_portal <xa-datasource> directive to use the MySQL database.

```
<datasources>
  <xa-datasource jndi-name="java:/jdbcidm_portal" pool-
name="IDMPortalDS" enabled="true" use-java-context="true">
    <xa-datasource-property name="ServerName">localhost</xa-
datasource-property>
    <xa-datasource-property name="DatabaseName">portalidm</xa-
datasource-property>
    <driver>mysql</driver>
    <xa-pool>
      <min-pool-size>10</min-pool-size>
      <max-pool-size>100</max-pool-size>
      <prefill>true</prefill>
    </xa-pool>
    <security>
      <user-name>portal</user-name>
      <password>portal</password>
    </security>
  </xa-datasource>
</datasources>
```

3. Configure jdbcjcr_portal <xa-datasource> directive to use the MySQL database.

```
<datasources>
  <xa-datasource jndi-name="java:/jdbcidm_portal" pool-
name="IDMPortalDS" enabled="true" use-java-context="true">
    <xa-datasource-property name="ServerName">localhost</xa-
datasource-property>
    <xa-datasource-property name="DatabaseName">portalidm</xa-
datasource-property>
    <driver>mysql</driver>
    <xa-pool>
      <min-pool-size>10</min-pool-size>
      <max-pool-size>100</max-pool-size>
      <prefill>true</prefill>
```

```
    </xa-pool>
    <security>
      <user-name>portal</user-name>
      <password>portal</password>
    </security>
  </xa-datasource>
  <xa-datasource jndi-name="java:/jdbcjcr_portal" pool-
name="JCRPortalDS" enabled="true" use-java-context="true">
    <xa-datasource-property name="ServerName">localhost</xa-
datasource-property>
    <xa-datasource-property name="DatabaseName">portaljcr</xa-
datasource-property>
    <driver>mysql</driver>
    <xa-pool>
      <min-pool-size>10</min-pool-size>
      <max-pool-size>100</max-pool-size>
      <prefill>true</prefill>
    </xa-pool>
    <security>
      <user-name>portal</user-name>
      <password>portal</password>
    </security>
  </xa-datasource>
</datasources>
```

4. Add the MySQL module into the list of <drivers>.

```
<datasources>
  <xa-datasource jndi-name="java:/jdbcidm_portal" pool-
name="IDMPortalDS" enabled="true" use-java-context="true">
    <xa-datasource-property name="ServerName">localhost</xa-
datasource-property>
    <xa-datasource-property name="DatabaseName">portalidm</xa-
datasource-property>
    <driver>mysql</driver>
    <xa-pool>
      <min-pool-size>10</min-pool-size>
      <max-pool-size>100</max-pool-size>
      <prefill>true</prefill>
    </xa-pool>
    <security>
      <user-name>portal</user-name>
      <password>portal</password>
    </security>
  </xa-datasource>
  <xa-datasource jndi-name="java:/jdbcjcr_portal" pool-
name="JCRPortalDS" enabled="true" use-java-context="true">
    <xa-datasource-property name="ServerName">localhost</xa-
datasource-property>
    <xa-datasource-property name="DatabaseName">portaljcr</xa-
datasource-property>
    <driver>mysql</driver>
    <xa-pool>
      <min-pool-size>10</min-pool-size>
      <max-pool-size>100</max-pool-size>
      <prefill>true</prefill>
```

```
    </xa-pool>
    <security>
      <user-name>portal</user-name>
      <password>portal</password>
    </security>
  </xa-datasource>
  <drivers>
    <driver name="mysql" module="com.mysql">
      <xa-datasource-
class>com.mysql.jdbc.jdbc2.optional.MysqlXADataSource</xa-
datasource-class>
  </driver>
  </drivers>
</datasources>
```

5. Open **JPP_HOME/standalone/configuration/gatein/configuration.properties**, and verify the gatein.jcr.datasource.name property has the value java:/jdbcidm. This value ensures the JCR database uses the IDM database as the datasource.

```
gatein.jcr.datasource.name=java:/jdbcjcr
gatein.idm.datasource.name=java:/jdbcidm
```

**See Also:**

- Section 7.4.6, “Last Resource Commit Optimization (LRCO)”

Report a bug

### 7.4.5. Configure Hibernate and Picketlink

Hibernate and PicketLink require some configuration changes to support JTA.

**Procedure 7.8. Configuring Hibernate and Picketlink for JTA**

1. Open **JPP_HOME/gatein/gatein.ear/portal.war/WEB-INF/conf/organization/idm-configuration.xml**
2. In the hibernate.properties <init-params> group, comment the Non-JTA Setup directives, and uncomment the JTA Setup directives:

```
<init-params>
  <properties-param>
    <name>hibernate.properties</name>
    <description>Default Hibernate Service</description>
    <property name="hibernate.hbm2ddl.auto" value="update"/>
    <property name="hibernate.show_sql" value="false"/>
    <property name="hibernate.connection.datasource"
value="${gatein.idm.datasource.name}${container.name.suffix}"/>
    <property name="hibernate.connection.autocommit"
value="false"/>
    <!--  Non-JTA setup  -->
    <!--<property name="hibernate.current_session_context_class"
value="thread"/>-->
```

```
    <!--  JTA setup  -->
    <property name="hibernate.current_session_context_class"
value="jta"/>
    <property name="hibernate.transaction.factory_class"
value="org.hibernate.transaction.JTATransactionFactory" />
    <property name="hibernate.transaction.jta.platform"
value="org.exoplatform.services.organization.idm.UserTransactionJta
Platform" />

<!-- Remaining code removed for readability-->
```

3. In the org.exoplatform.services.organization.idm.Config <init-params> group, change useJTA <field> value to **true**.

```
<field name="useJTA">
  <boolean>true</boolean>
</field>
```

This setting forces PicketlinkIDMOrganizationServiceImpl to encapsulate each HTTP request within the JTA transaction, instead of the Hibernate transaction API. The bounds of the transaction are startRequest and endRequest.

**Important**

If managed transactions are used in portlet applications (for example, using EJB with Container-Managed transactions), be aware that some changes to transaction behavior may occur in the application.

Ensure that portlet applications do not attempt to start or commit JTA transactions from the application.

4. Open **JPP_HOME/gatein/gatein.ear/portal.war/WEB-INF/conf/organization/picketlink-idm/picketlink-idm-config.xml**

5. In the <stores> directives, change the lazyStartOfHibernateTransaction option to **false** to switch transaction management from PicketLink to JTA.

Report a bug

### 7.4.6. Last Resource Commit Optimization (LRCO)

The XA transaction protocol is designed to provide ACID properties by using a two-phase commit protocol. Sometimes it is necessary to allow a non-XA-aware resource manager to participate in a transaction. This is often the case with data stores that do not support distributed transactions.

In this situation, a technique known as Last Resource Commit Optimization (LRCO) is used. LRCO can also be referred to as the Last Resource Gambit. The one-phase-aware resource is processed last in the prepare phase of the transaction, at which time an attempt is made to commit it.

If the attempt is successful, the transaction log is written and the remaining resources go through the phase-two commit. If the last resource fails to commit, the transaction is rolled back. Although this protocol allows most transactions to complete normally, some errors can cause an inconsistent

transaction outcome. For this reason, use LRCO as a last resort. When a single <datasource> is used in a transaction, the LRCO is automatically applied to it. In other situations, you can designate a last resource by using a special marker interface.

Using more than a single one-phase resource in the same transaction is not transactionally safe, and is not recommended. JBoss Transaction Service sees an attempt to enlist a second such resource as an error and terminates the transaction. Whenever possible, <datasource> must be converted to an <xa-datasource> to prevent any potential transaction issues.

Report a bug

## 7.5. Email Service Configuration

Red Hat JBoss Portal (JBoss Portal) includes an email service. The service can be used to send authentication information to users, and alert administrators to invalid authentication attempts. The service must be configured before use.

**Procedure 7.9. Configuring the SMTP Email Service**

The service is configured to support GMail by default. Google Accounts with two-factor authentication activated must generate an application-specific password for the email service through the Google Account Security Console.

1. Open ***JPP_HOME*/standalone/configuration/gatein/configuration.properties** in edit mode.

2. Specify the Google Account information as indicated in the file.

```
# EMail
gatein.email.smtp.username=[user@gmail.com]
gatein.email.smtp.password=[password|app-specific password]
gatein.email.smtp.host=smtp.gmail.com
gatein.email.smtp.port=465
gatein.email.smtp.starttls.enable=true
gatein.email.smtp.auth=true
gatein.email.smtp.socketFactory.port=465
gatein.email.smtp.socketFactory.class=javax.net.ssl.SSLSocketFactory
```

3. When provisioning the Email Service for a production environment, it is recommended to specify a corporate SMTP gateway and port for the **smtp.host** and **smtp.port** variables.

   If using a SMTP gateway over SSL, a certificate trust store containing the SMTP server public certificate is required. Depending on the key sizes, Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy files are required for the Java Runtime Environment (JRE).

Report a bug

## 7.6. Email Notifications Configuration

**Prerequisites:**

- Section 7.5, “Email Service Configuration”

The email service can be configured to send notifications to a specified email address, typically to

the portal administrators. Two types of notifications are available:

- Notifications about new users, which are typically useful on portals that have a public registration page.
- Notifications about invalid login attempts, which can help in identifying security attacks.

Notification emails are disabled by default. To enable the notifications, perform the following configuration procedures.

**Procedure 7.10. Enabling Notifications about New User Registration**

1. Configure an SMTP account.
2. Configure the following mandatory configuration items located in the **`PostRegistrationService`** part of the **`JPP_HOME/gatein/gatein.ear/portal.war/WEB-INF/conf/admin/admin-configuration.xml`** configuration file.

   **sendMailAfterRegistration**

   Boolean value that specifies whether an email notification is sent for new user registrations.

   The default value is **`false`**. If set to **`true`**, email notifications are sent when users join.

   **mailTo**

   Specifies the email address to which notifications are sent.
3. Configure optional parameters to customize the content of the notification emails:

   **mailFrom**

   Specifies the email address from which the notification email is sent.

   **mailSubject**

   Specifies the subject of the notification email.

   **mailMessage**

   Specifies the body of the notification email.

   The **`${user.name}`**, **`${user.firstName}`**, **`${user.lastName}`** and **`${user.email}`** macros can be used in the text to dynamically include the respective values in each notification email.
4. Test the configuration by restarting the server and registering a new user in the JBoss Portal user interface. An email is sent to the specified email address, notifying the recipient that a user has been registered to the portal.

**Procedure 7.11. Enabling Notifications about Invalid Login Attempts**

1. Configure an SMTP account.

2. Configure the following mandatory configuration items located in the **InvalidLoginAttemptsService** part of the **JPP_HOME/gatein/gatein.ear/portal.war/WEB-INF/conf/admin/admin-configuration.xml** configuration file.

   **sendingMailsEnabled**

   boolean value determining if e-mail notifications about invalid login attempts are enabled. It is **false** by default and needs to be changed to **true** for the e-mail notifications to be sent.

   **mailTo**

   Specifies the email address to which notifications are sent.

3. Configure optional parameters to customize the content and policies associated with notification emails:

   **mailFrom**

   Specifies the email address from which the notification email is sent.

   **mailSubject**

   Specifies the subject of the notification email.

   **mailMessage**

   Specifies the body of the notification email.

   The **${user.name}**, **${user.firstName}**, **${user.lastName}** and **${user.email}** macros can be used in the text to dynamically include the respective values in each notification email.

   **numberOfFailedAttempts**

   Specifies the number of invalid login attempts after which an email notification is sent. The default value is **3**.

   **invalidLoginPolicy**

   Specifies the policy used to determine that invalid login attempts are coming from same source. The available values are

   **SESSION** - the default, indicating that invalid login attempts must originate from the same HTTP session.

   **SESSION_AND_USER** - indicates that invalid login attempts must originate from the same HTTP session, and contain the same user name.

   **SERVER** - indicates that invalid login attempts must come from the same remote server.

4. Test the configuration by restarting the server and attempting to login to JBoss Portal with invalid account credentials.

   Repeat the attempt for the number of times specified in the **numberOfFailedAttempts** parameter (the default is three attempts).

5. An email is sent to the specified email address, notifying the recipient that an invalid login attempt has been detected.

Report a bug

## 7.7. Clustering Configuration

Before configuring a clustered environment in Red Hat JBoss Portal (JBoss Portal), read and understand the HTTP Clustering and Load Balancing chapter of the Red Hat JBoss Enterprise Application Platform *Administration and Configuration Guide*.

In summary, clustering can be configured one of two ways:

**Using a single physical server**

One physical server using two virtual IP addresses.

This configuration is used during basic clustering testing in a development environment, and can use the pre-configured Hypersonic (H2) database (which is not supported in production environments).

For assistance configuring virtual IP addresses, consult the Operating System documentation for guidance.

**Using more than one physical server**

This method uses two or more servers, each configured to use a JBoss Portal binary. For example, a "node1" directory available on server one and a "node2" directory available on server two.

This configuration is recommended for production environments. A supported databased documented in Tested Configurations is required for this configuration.

### Configuring a Red Hat JBoss Portal Virtual Cluster for Basic Testing

Implement the following configuration to enable a cluster of JBoss Portal instances hosted on the same machine, bound to the virtual IPs 192.168.210.101 (for node1) and 192.168.210.102 (for node2).

All configuration required for basic clustering in JBoss Portal is included in **`standalone-ha.xml`** file. For this example, no modification is required to the file. For production environments, changes to this file are required and are documented in this guide.

1. Configure two virtual IP addresses:
    a. 192.168.210.101, which is used for "node1" in this procedure.
    b. 192.168.210.102, which is used for "node2" in this procedure.
2. Create two cluster nodes from the JBoss Portal binary:

```
$ cp -r jboss-jpp-6.2/ node1
$ cp -r jboss-jpp-6.2/ node2
```

3. Copy the Hypersonic database JAR into the JRE to act as the data store for the virtual servers.

```
$ java -cp modules/system/layers/base/com/h2database/h2/main/h2-
1.3.168.redhat-4.jar org.h2.tools.Server
```

4. Start the servers from the node1 and node2 directories:

   From node1 directory.

   ```
   $ ./bin/standalone.sh --server-config=standalone-ha.xml -
   Djboss.node.name=node1 -b 192.168.210.101 -u 239.23.42.2 -
   Djboss.bind.address.management=192.168.210.101
   ```

   From node2 directory.

   ```
   $ ./bin/standalone.sh --server-config=standalone-ha.xml -
   Djboss.node.name=node2 -b 192.168.210.102 -u 239.23.42.2 -
   Djboss.bind.address.management=192.168.210.102
   ```

   Both nodes start, and can be accessed through their web interfaces:

   - node1 on http://192.168.210.101:8080/portal
   - node2 on http://192.168.210.102:8080/portal

**See Also:**

- Section 7.3, “Provision JCR and IDM Databases”

Report a bug

## 7.8. Setting mod_jk for Cluster Configuration

**Procedure 7.12. To setup the loadbalancer server which uses Apache HTTPD+Mod_jk:**

1. **Install apache server and mod_jk module**

   The package that contains Apache HTTP Server is known as *httpd*. To build and install mod_jk from source code use the package *httpd-devel*.

2. **Setup apache to use mod_jk**

   Copy the file **`mod-jk.conf`** to **`/etc/httpd/conf.d`** and append the following line to **`/etc/httpd/conf/httpd.conf`** to load the module.

   ```
   LoadModule jk_module modules/mod_jk.so
   ```

   File **`mod-jk.conf`** is as follows:

   ```
   # Where to find workers.properties
   JkWorkersFile workers.properties

   # Where to put jk logs
   JkLogFile /var/log/apache2/mod_jk.log

   # Set the jk log level [debug/error/info]
   JkLogLevel debug

   # Select the log format
   ```

```
JkLogStampFormat "[%a %b %d %H:%M:%S %Y]"

# JkOptions indicates to send SSK KEY SIZE
JkOptions +ForwardKeySize +ForwardURICompat -ForwardDirectories

# JkRequestLogFormat
#JkRequestLogFormat "%w %V %T"

JkMountFile uriworkermap.properties

# Add shared memory.
# This directive is present with 1.2.10 and
# later versions of mod_jk, and is needed for load balancing to
work properly
JkShmFile /var/log/apache2/jk.shm

# Add jkstatus for managing runtime data
<Location> /jkstatus/>
JkMount status
</Location>
```

3. **Setup Workers:**

   Create **workers.properties** file in /etc/httpd/.

**Note**

The Balanced workers must have the same name as the name of jboss.node.name as shown in Section 7.7, “Clustering Configuration”. The example uses the names node1 and node2. The values for balanced workers in **workers.properties** file must also have the same name as node1 and node2.

The file **workers.properties** is as follows:

```
# Define list of workers that will be used for mapping requests
worker.list=loadbalancer,status

# modify the host as your host IP or DNS name
worker.node1.port=8009
worker.node1.host=192.168.210.101
worker.node1.type=ajp13
worker.node1.lbfactor=1

## modify the host as your host IP or DNS name
worker.node2.port=8009
worker.node2.host=192.168.210.102
worker.node2.type=ajp13
worker.node2.lbfactor=1

# Load-balancing behaviour
worker.loadbalancer.type=lb
worker.loadbalancer.method=Session
worker.loadbalancer.balance_workers=node1,node2
```

```
worker.loadbalancer.sticky_session=1

#worker.list=loadbalancer
worker.status.type=status
```

The file **uriworkermap.properties** is as follows:

```
/portal=loadbalancer
/portal/*=loadbalancer
/eXo*=loadbalancer
/eXoResources*/*=loadbalancer
/exo*=loadbalancer
/exo*/*=loadbalancer
/web=loadbalancer
/web/*=loadbalancer
/integration=loadbalancer
/integration/*=loadbalancer
/dashboard=loadbalancer
/dashboard/*=loadbalancer
/rest=loadbalancer
/rest/*=loadbalancer
/jpp_branding_skin|/*=loadbalancer
/jpp-branding-skin|/*=loadbalancer
/jpp-branding-extension|/*=loadbalancer
/status=status
/status/*=status
```

**Troubleshooting error *"503 Service Temporarily Unavailable"***

**Problem Description**

When accessing JBoss Portal using Apache, a "503 Service Temporarily Unavailable" response is received when trying to access JBoss Portal using Apache. The cluster is working and the individual JBoss Portal nodes can be accessed directly. The logs mod_jk.log indicates mod_jk is working, but Tomcat is reported as not running on the specified port.

**Cause**

In Red Hat Enterprise Linux, SELinux prevents httpd from accessing an important resource, such as jk.shm. Check SELinux alerts to verify this fact.

**Solution**

Temporarily disable SELinux to allow Apache to initialize the mod_jk connector properly by executing the following command:

```
setenforce 0
```

Report a bug

## 7.9. HTTPS Configuration

Red Hat JBoss Portal (JBoss Portal) runs by default in HTTP mode. For security purposes, configure a production platform to run in HTTPS mode.

**Important**

Understanding the fundamentals of keystore and truststore configuration is critical to the tasks in this section.

See the About SSL Encryption section in the Red Hat JBoss Enterprise Application Platform 6 *Development Guide* for detailed encryption theory and procedures relevant to all JBoss Middleware platforms.

There are a number of steps required to enable HTTPS on the platform. In summary, the steps are:

1. Generate encryption keys and certificate.
2. Export the self-signed certificate.
3. Import the certificate to the trust store.
4. Define the keystore and truststore details in the HTTPS connector directive file.

Report a bug

## 7.10. Enable HTTPS Communication

### Generate a Keystore using the Java Keytool

Follow this procedure to create a key and certificate using the Java keytool utility.

**Note**

If you are using Tomcat Native Libraries, you must use openSSL to generate the key and certificate.

On Linux distributions, the Tomcat native libraries are enabled by default. To disable the libraries, set the **`native="false"`** flag on the web subsystem configuration.

### Prerequisites

- Read and understand how keypairs and certificates operate in the About SSL Encryption section of the JBoss Enterprise Application Platform 6 *Development Guide*.
- Understand the command-line parameters of **`keytool`** as documented in the Oracle Java Tools documentation.
- Obtain the JDK keystore password. For new installations, the default JDK password is "**`changeme`**".

1. Run the command to generate a simple certificate using the keytool command (if you do not have your own X.509 certificate). The certificate is stored in the **`server.keystore`** file.

```
keytool -genkey -alias serverkeys -keyalg RSA -keystore
server.keystore -storepass 123456 -keypass 123456 -dname
"CN=localhost, OU=MYOU, O=MYORG, L=MYCITY, ST=MYSTATE, C=MY"
```

2. Import the key into the Sun JDK keystore:

```
keytool -importkeystore -srckeystore server.keystore -destkeystore
$JAVA_HOME/jre/lib/security/cacerts
```

**Note**

On OS X, the **cacerts** file is located at **$JAVA_HOME/lib/security/cacerts**

3. Change the key password to match the new keystore password (in most instances this is the default JDK trustore password; **changeit**.

```
keytool -keypasswd -alias serverkeys --keystore
$JAVA_HOME/jre/lib/security/cacerts
```

### Generate a Keystore using OpenSSL

Follow this procedure to create a key and certificate using openSSL.

**Note**

If you are using Tomcat Native Libraries, you must use openSSL to generate the key and certificate.

On Linux distributions, the Tomcat native libraries are enabled by default. To disable the libraries, set the **native="false"** flag on the web subsystem configuration.

### Prerequisites

- Read and understand how keypairs and certificates operate in the "About SSL Encryption" section of the JBoss Enterprise Application Platform 6 *Development Guide*
- Understand the command-line parameters of **openssl** as documented in the command-line help.
- Obtain the JDK keystore password. For new installations, the default JDK password is "**changeme**".

1. Generate a simple key file using the openssl genrsa command:

```
openssl genrsa -des3 -out server.pem 1024
```

The above command will store the key in **server.keystore** .

2. Generate a certificate signing request using the openssl **-req** command:

```
openssl req -new -key server.pem -out server.csr
```

3. Generate a server certificate from the server.csr key using the openssl **-req** command:

```
openssl x509 -req -days 365 -in server.csr -signkey server.pem -out
servercert.pem
```

**Configure JBoss Portal to Use The Key**

Configure JBoss Portal to accept the key, for either OpenSSL or Java keytool.

1. Edit ***JPP_HOME*/standalone/configuration/standalone.xml** to add the HTTPS connector to the web subsystem configuration.

   Change **certificate-key-file** and **password** to values appropriate for the keystore. This procedure assumes the keystore password is the default: **changeit**):

   For Java keytool, as described in Generate a Keystore using the Java Keytool:

```
<subsystem xmlns="urn:jboss:domain:web:1.2" default-virtual-
server="default-host" native="false">
    ...
    <connector name="https" protocol="HTTP/1.1" socket-
binding="https" scheme="https" secure="true">
        <ssl name="https" key-alias="serverkeys" password="changeit"
certificate-key-file="${java.home}/jre/lib/security/cacerts"/>
    </connector>
    ...
</subsystem>
```

   For OpenSSL, as described in Generate a Keystore using OpenSSL:

```
<connector name="https" protocol="HTTP/1.1" scheme="https"
socket-binding="https" secure="true" enabled="true">
  <ssl password="mypassword" certificate-key-file="server.pem"
protocol="TLSv1"
verify-client="true" certificate-file="servercert.pem"/>
</connector>
```

2. Save and close the file.
3. Restart the portal.
4. JBoss Portal can now communicate using a secure connection.

Report a bug

## 7.11. Specify White-list and Black-list Gadget Proxy Resources

To work around web browser security mechanisms, gadget consumption requires a local anonymous proxy to route access to elements used by the gadgets.

The anonymous proxy is configured to accept or refuse certain hosts by specifying a white-list (of allowed hosts) and a black-list (of denied hosts). By default, the proxy is closed to any host except the domain on which the gadget server is installed.

### Task: Specify White-list and Black-list Gadget Proxy Resources

Complete this task to configure the gadget proxy filter to allow a specific list of gadget resources.

### Prerequisites

- You have installed the platform, and performed all configuration tasks up to this task.

1. Navigate to ***JPP_HOME*/gatein/gatein.ear/portal.war/WEB-INF/conf/common/**
2. Open **common-configuration.xml** in a text editor.
3. Append the ProxyFilterService <component> block to the file.

```
<component>
   <key>org.exoplatform.web.security.proxy.ProxyFilterService</key>

<type>org.exoplatform.web.security.proxy.ProxyFilterService</type>
    <init-params>
      <values-param>
        <!-- The white list -->
        <name>white-list</name>
        <value></value>
      </values-param>
      <values-param>
        <name>black-list</name>
        <value></value>
      </values-param>
    </init-params>
  </component>
```

4. In the <name>white-list</name> block, specify the name of the domains you want to grant access to by adding <value> directives for each domain name.

**Important**

A required resource must be explicitly defined in the white-list. Failure to do so will result in the resource being treated as black-listed.

Wildcard characters can be used to simplify configuration. For example, ***.example.com** would allow all domains with **example.com** as the suffix.

5. In the <name>black-list</name> block, specify the name of the domains you want to deny access to by adding <value> directives for each domain name.

   Wildcard characters can be used to simplify configuration. For example, ***.example.com** would deny all domains with **example.com** as the suffix.

6. Save and close **common-configuration.xml**.

7. You have specified the gadget domains required in your installation. This completes the procedure.

**Related Information**

**Example 7.6. Valid Proxy Configuration**

```
<component>
   <key>org.exoplatform.web.security.proxy.ProxyFilterService</key>
   <type>org.exoplatform.web.security.proxy.ProxyFilterService</type>
    <init-params>
      <values-param>
        <!-- The white list -->
        <name>white-list</name>
        <value>*.example.com</value>
        <value>www.example.net</value>
      </values-param>
      <values-param>
        <name>black-list</name>
        <value>forbidden.example.com</value>
      </values-param>
    </init-params>
</component>
```

Report a bug

## 7.12. Validator Configuration

A configurable validator can be applied to user name fields in the user account, user registration, and group membership portlets.

The validator enforces the validation mask for user name formats in these portlets, which results in consistent and valid user names.

The architecture does allow for configurable validation to be used in different contexts if needed, however this feature is not officially supported in Red Hat JBoss Portal (JBoss Portal).

The validator is configured through properties set in the ***JPP_HOME*/standalone/configuration/gatein/configuration.properties** file.

**Important**

Some components used in the portal depend on user names being all lowercase. Consider only accepting lowercase user names to maintain cross-compatibility.

The default validator behavior ensures a user name meets the following requirements:

- Length must be between 3 and 30 characters.
- Must start with a letter.

- Must end with a letter or number.
- Only lowercase letters, numbers, underscores (_) and period (.) can be used.
- No consecutive underscores (_) or period (.) can be used.

To alter the default behavior, the validator supports four aspects:

**gatein.validators.*[username|groupmembership]*.length.min**

Minimum length of the validated field.

**gatein.validators.*[username|groupmembership]*.length.max**

Maximum length of the validated field.

**gatein.validators.*[username|groupmembership]*.regexp**

Regular expression to which values of the validated field must conform.

**gatein.validators.*[username|groupmembership]*.format.message**

Information message that displays when the value of the validated field does not conform to the specified regular expression.

Within each aspect, ***`[username|groupmembership]`*** refers to the configuration type.

The **`username`** configuration type sets how user names are validated when created by the user registration portlet, or modified by the user account portlet.

The **`groupmembership`** configuration type sets how user names are validated when created or modified by the group membership portlet.

The **`email`** configuration type sets how email fields are validated.

The **`displayname`** configuration type sets how user's display name fields are validated.

The **`jobtitle`** configuration type sets how user profile's job title fields are validated.

The **`grouplabel`** configuration type sets how group membership label is validated.

**Example 7.7. Email as the User Name Mask**

Use the following configuration to ensure users configure an email address as their user name.

```
# validators
            gatein.validators.username.regexp=^[A-Za-z0-9._%+-]+@[A-Za-
z0-9.-]+\.[A-za-z]{2,4}$
            gatein.validators.username.format.message=Username must be
a valid email address.
```

**Example 7.8. Letter and Number Combination**

Use the following configuration to ensure users configure a user name starting with the letter "u", followed by four to nine digits.

```
# validators
```

```
gatein.validators.username.length.min=5

gatein.validators.username.length.max=10

gatein.validators.username.regexp=^u\d{4,9}$

gatein.validators.username.format.message=Username must start with
''u'' and be followed by 4 to 9 digits.
```

Report a bug

## 7.13. Custom Password Policy

Red Hat JBoss Portal Platform allows to modify the password policy using the ***passwordpolicy*** property.

The ***passwordpolicy*** property is defined in the **configuration.properties** file.

The code below shows how the ***passwordpolicy*** property defines the format, length, and valid expression for an acceptable password.

```
gatein.validators.passwordpolicy.format.message=Minimum of 1 digit, 1
lower case, 1 upper case, minimum of 6 chars, max of 20.
gatein.validators.passwordpolicy.regexp=((?=.*\\d)(?=.*[a-z])(?=.*[A-Z]).
{6,20})
gatein.validators.passwordpolicy.length.max=20
gatein.validators.passwordpolicy.length.min=6
```

**Important**

Restart the server after a password policy change to reload the policy.

Report a bug

## 7.14. Password Encryption

### 7.14.1. About Remember Me Password Encryption

The automatic login feature of Red Hat JBoss Portal (JBoss Portal) uses a cookie token mechanism to authenticate returning users. The tokens are not stored in plain text, but instead only their salted hash is stored. Due to an internal design limitation of JBoss Portal, the Remember Me feature requires that passwords are stored along with tokens. Passwords are encrypted symmetrically before they are stored.

Report a bug

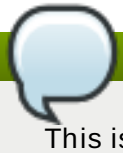
### 7.14.2. Symmetric Password Encryption

The implementation is based on the Java Cryptography Architecture (JCA) library. By default, the AES algorithm is used for password encryption. All secrets needed in the process must be created, maintained, and stored securely by the Portal operator.

**Example 7.9. JCA-based Configuration**

The JCA-based encryption is configured in the **configuration.properties** file.

```
gatein.codec.builderclass=org.exoplatform.web.security.codec.JCASymmetr
icCodecBuilder
gatein.codec.config=${gatein.conf.dir}/codec/jca-symmetric-
codec.properties
```



**Note**

This is an example of the directives required. There is no **jca-symmetric-codec.properties** file in the default JBoss Portal binary.

In this example, the builderclass configuration specifies **org.exoplatform.web.security.codec.JCASymmetricCodecBuilder** is used, and that it is configured in the **jca-symmetric-codec.properties** file.

**Example 7.10. Customized Properties File**

The **configuration.properties** values can be changed if the default settings require customization. Observe how the values in the file correspond to the **keytool** command parameters.

```
# Detailed information on JCA standard names can be found at
#
#
http://docs.oracle.com/javase/7/docs/technotes/guides/security/Standard
Names.html#KeyStore
#
# The file key.txt is generated using the keytool utility in the JDK
#
# keytool -genseckey -alias "gtnKey" -keypass "gtnKeyPass" -keyalg
"AES" -keysize 128 -keystore "key.txt" -storepass "gtnStorePass" -
storetype "JCEKS"
#
#
gatein.codec.jca.symmetric.alias=gtnKey
gatein.codec.jca.symmetric.keypass=gtnKeyPass
gatein.codec.jca.symmetric.keyalg=AES
gatein.codec.jca.symmetric.keystore=key.txt
gatein.codec.jca.symmetric.storepass=gtnStorePass
gatein.codec.jca.symmetric.storetype=JCEKS
```

Report a bug

### 7.14.3. Customization using JCASymmmetricCodecBuilder

You can customize many aspects of the default setup, such as algorithm, key storage, key size and so on.

**Example 7.11. Command to generate secret key**

To generate a secret key that suits your needs, you will need to issue something like the following:

```
$JAVA_HOME/bin/keytool -genseckey -alias "customAlias" -keypass
"customKeyPass" -keyalg "customAlgo" -keystore "customStore" -storepass
"customStorePass" -storetype "customStoreType"
```

**Note**

- The list of available algorithms can be found in Standard Algorithm Name Documentation .
- Some extra params for keytool might be required for some algorithms.
- In JCA, only JCEKS storetype supports symmetric encryption.

The **keytool** command stores the freshly generated secret key in a file named **customStore** . Copy this file to the **gatein/conf/codec** directory to be able to reference it in **jca-symmetric-codec.properties** file.

It remains to update the **jca-symmetric-codec.properties** file with parameters used in the above **keytool** invocation:

**Example 7.12. Update jca-symmetric-codec.properties file**

```
gatein.codec.jca.symmetric.alias=customAlias
gatein.codec.jca.symmetric.keypass=customKeyPass
gatein.codec.jca.symmetric.keyalg=customAlgo
gatein.codec.jca.symmetric.keystore=customStore
gatein.codec.jca.symmetric.storepass=customStorePass
gatein.codec.jca.symmetric.storetype=customStoreType
```

Report a bug

# Revision History

| | | |
|---|---|---|
| **Revision 6.2.0-4** | **Mon May 11 2015** | **Aakanksha Singh** |

Prepared the guide for 6.2.0 GA.