



# **Red Hat JBoss BPM Suite 6.1**

## **Installation Guide**

For Red Hat JBoss Administrators



# Red Hat JBoss BPM Suite 6.1 Installation Guide

---

For Red Hat JBoss Administrators

Kanchan Desai  
kadesai@redhat.com

Doug Hoffman

Eva Kopalova

B Long  
Red Hat Engineering Content Services  
belong@redhat.com

Petr Penicka

Red Hat Content Services

Gemma Sheldon  
Red Hat Engineering Content Services  
gsheldon@redhat.com

## Legal Notice

Copyright © 2015 Red Hat, Inc.

This document is licensed by Red Hat under the [Creative Commons Attribution-ShareAlike 3.0 Unported License](#). If you distribute this document, or a modified version of it, you must provide attribution to Red Hat, Inc. and provide a link to the original. If the document is modified, all Red Hat trademarks must be removed.

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

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

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

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

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

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

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

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

All other trademarks are the property of their respective owners.

## Abstract

This guide provides the steps necessary for administrators to install Red Hat JBoss BPM Suite, the plug-ins for Red Hat JBoss Developer Studio, and provides instructions for running example projects.

## Table of Contents

<b>CHAPTER 1. INTRODUCTION</b> .....	<b>3</b>
1.1. ABOUT RED HAT JBOSS BPM SUITE	3
1.2. SUPPORTED PLATFORMS	3
1.3. USE CASE: PROCESS-BASED SOLUTIONS IN THE LOAN INDUSTRY	3
<b>PART I. RED HAT JBOSS BPM SUITE INSTALLATION</b> .....	<b>5</b>
<b>CHAPTER 2. INSTALLATION OPTIONS</b> .....	<b>6</b>
2.1. THE RED HAT JBOSS BPM SUITE INSTALLER INSTALLATION	6
2.2. THE GENERIC DEPLOYABLE BUNDLE INSTALLATION	22
<b>CHAPTER 3. SPECIAL SETUPS</b> .....	<b>31</b>
3.1. SETTING UP PERSISTENCE FOR BUSINESS CENTRAL	31
3.2. SETTING UP PERSISTENCE FOR DASHBUILDER	32
3.3. SPECIAL SETUP FOR IBM DB2 DATABASE	33
<b>CHAPTER 4. ROLES AND USERS</b> .....	<b>34</b>
4.1. DEFINING ROLES	34
4.2. CREATING USERS	34
<b>CHAPTER 5. TESTING THE INSTALLATION</b> .....	<b>36</b>
5.1. STARTING THE SERVER	36
5.2. JAVA SECURITY MANAGER AND PERFORMANCE MANAGEMENT	37
5.3. LOGGING ON TO BUSINESS CENTRAL	38
<b>CHAPTER 6. CLUSTERING</b> .....	<b>39</b>
GIT REPOSITORY CLUSTERING MECHANISM	39
CLUSTERING MAVEN REPOSITORIES	40
6.1. SETTING UP A CLUSTER	41
6.2. SETTING UP QUARTZ	43
6.3. CONFIGURING CLUSTERING ON RED HAT JBOSS EAP	44
<b>CHAPTER 7. MAVEN REPOSITORIES</b> .....	<b>50</b>
7.1. ABOUT MAVEN	50
7.2. ABOUT THE PROVIDED MAVEN REPOSITORIES	50
7.3. CONFIGURING MAVEN TO USE THE FILE SYSTEM REPOSITORIES	50
7.4. CONFIGURING MAVEN TO USE THE ONLINE REPOSITORIES	54
7.5. DEPENDENCY MANAGEMENT	58
<b>CHAPTER 8. RED HAT JBOSS DEVELOPER STUDIO</b> .....	<b>59</b>
8.1. RED HAT JBOSS DEVELOPER STUDIO	59
8.2. INSTALLING THE JBOSS DEVELOPER STUDIO PLUG-INS	59
8.3. SETTING THE DROOLS RUNTIME	59
8.4. CONFIGURING THE JBOSS BPM SUITE SERVER	60
8.5. IMPORTING PROJECTS FROM A GIT REPOSITORY INTO JBOSS DEVELOPER STUDIO	60
<b>CHAPTER 9. PATCHING AND UPGRADING RED HAT JBOSS BPM SUITE</b> .....	<b>64</b>
9.1. ABOUT PATCHES AND UPGRADES	64
9.2. APPLYING PATCHES IN RED HAT JBOSS BPM SUITE 6.1	64
9.3. PATCHING OTHER PLATFORMS AND APPLICATIONS	66
<b>APPENDIX A. REVISION HISTORY</b> .....	<b>68</b>



# CHAPTER 1. INTRODUCTION

## 1.1. ABOUT RED HAT JBOSS BPM SUITE

Red Hat JBoss BPM Suite is an open source business process management suite that combines Business Process Management and Business Rules Management and enables business and IT users to create, manage, validate, and deploy Business Processes and Rules.

Red Hat JBoss BRMS and Red Hat JBoss BPM Suite use a centralized repository where all resources are stored. This ensures consistency, transparency, and the ability to audit across the business. Business users can modify business logic and business processes without requiring assistance from IT personnel.

To accommodate Business Rules component, Red Hat JBoss BPM Suite includes integrated Red Hat JBoss BRMS.

Business Resource Planner is included with this release.

Red Hat JBoss BPM Suite is supported for use with Red Hat Enterprise Linux 7 (RHEL7) .

## 1.2. SUPPORTED PLATFORMS

Red Hat JBoss BPM Suite and Red Hat JBoss BRMS are supported on the following containers:

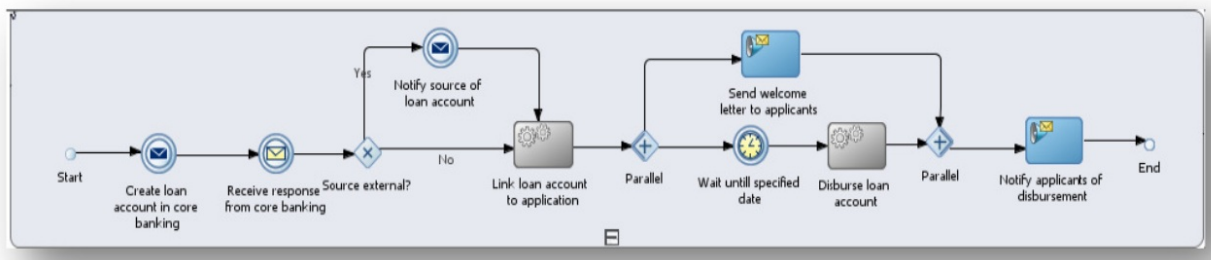
- Red Hat JBoss Enterprise Application Platform 6.4
- Red Hat JBoss Web Server 2.1 (Tomcat 7) on JDK 1.7
- IBM WebSphere Application Server 8.5.5.0
- Oracle WebLogic Server 12.1.3 (12c)

## 1.3. USE CASE: PROCESS-BASED SOLUTIONS IN THE LOAN INDUSTRY

This section describes a use case of deploying JBoss BPM Suite to automate business processes (such as loan approval process) at a retail bank. This use case is a typical process-based specific deployment that might be the first step in a wider adoption of JBoss BPM Suite throughout an enterprise. It leverages features of both business rules and processes of JBoss BPM Suite.

A retail bank offers several types of loan products each with varying terms and eligibility requirements. Customers requiring a loan must file a loan application with the bank. The bank then processes the application in several steps, such as verifying eligibility, determining terms, checking for fraudulent activity, and determining the most appropriate loan product. Once approved, the bank creates and funds a loan account for the applicant, who can then access funds. The bank must be sure to comply with all relevant banking regulations at each step of the process, and has to manage its loan portfolio to maximize profitability. Policies are in place to aid in decision making at each step, and those policies are actively managed to optimize outcomes for the bank.

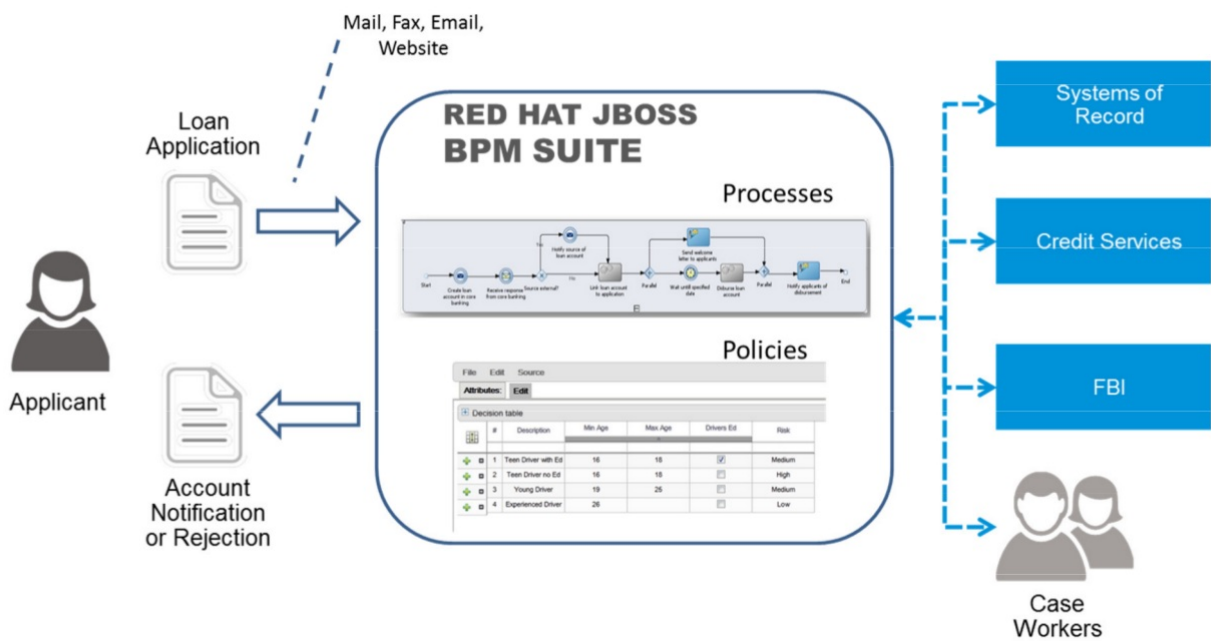
Business analysts at the bank model the loan application processes using the BPMN2 authoring tools (Process Designer) in JBoss BPM Suite. Here is the process flow:



**Figure 1.1. High-level loan application process flow**

Business rules are developed with the rule authoring tools in JBoss BPM Suite to enforce policies and make decisions. Rules are linked with the process models to enforce the correct policies at each process step.

The bank's IT organization deploys the JBoss BPM Suite so that the entire loan application process can be automated.



**Figure 1.2. Loan Application Process Automation**

The entire loan process and rules can be modified at any time by the bank's business analysts. The bank is able to maintain constant compliance with changing regulations, and is able to quickly introduce new loan products and improve loan policies in order to compete effectively and drive profitability.



# PART I. RED HAT JBOSS BPM SUITE INSTALLATION

## CHAPTER 2. INSTALLATION OPTIONS

Red Hat JBoss BPM Suite comes in two versions:

- Executable jar installer for installation on Red Hat JBoss Enterprise Application Platform (EAP) 6.4.
- Zip file install which itself comes in two versions:
  - `jboss-bpmsuite-6.MINOR_VERSION-deployable-eap6.x.zip`: version adapted for deployment on Red Hat JBoss Enterprise Application Platform (EAP 6.4).
  - `jboss-bpmsuite-6.MINOR_VERSION-deployable-generic.zip`: the deployable version with additional libraries adapted for deployment on Red Hat JBoss Web Server (EWS), Apache Tomcat 6, and Apache Tomcat 7.

Depending on your environment, you may choose the installation option best suited for your project needs.



### NOTE

Red Hat JBoss BPM Suite is designed to work with UTF-8 encoding. If a different encoding system is used by the underlying JVM, unexpected errors might occur. To ensure UTF-8 is used by the JVM, use the following system property `-Dfile.encoding=UTF-8`.



### IMPORTANT

From JBoss BPM Suite 6.1 onwards, you must have JBoss EAP 6.4 or better already installed before attempting to install JBoss BPM Suite.



### WARNING

In addition to the following install steps, we strongly recommend to set the system property `org.kie.tx.lock.enabled` to `false` in order to prevent unresponsiveness or deadlock issues. You can either start your server with the option `-Dorg.kie.tx.lock.enabled=false` or edit the `standalone.xml` file:

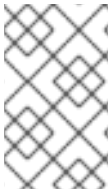
```
<system-properties>
  <property name="org.kie.tx.lock.enabled" value="false"/>
  ...
</system-properties>
```

For further details, please refer to this article:  
<https://access.redhat.com/solutions/1610723>.

## 2.1. THE RED HAT JBOSS BPM SUITE INSTALLER INSTALLATION

This section describes the steps required to install Red Hat JBoss BPM Suite using the jar file installer installation method. The jar file is an executable file that installs JBoss BPM Suite on an existing JBoss EAP 6 installation.

From JBoss BPM Suite 6.1 onwards, you must have JBoss EAP 6.4 or better already installed in order to install JBoss BPM Suite.



#### NOTE

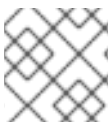
Due to IBM JDK not being able to use keystores generated on other JDKs, it is not possible to install JBoss BPM Suite into an existing JBoss EAP running on IBM JDK with a keystore generated on another JDK.

### 2.1.1. Downloading Red Hat JBoss BPM Suite for JBoss EAP

1. Go to the [Red Hat Customer Portal](#) and log in.
2. Click **Downloads** → **Products Downloads**.
3. In the **Product Downloads** page that opens, click **Red Hat JBoss BPM Suite**.
4. From the **Version** drop-down menu, select version **6.1**.
5. Select **Red Hat JBoss BPM Suite 6.1 Deployable for EAP 6.4** and then click **Download**.

### 2.1.2. Installing Red Hat JBoss BPM Suite Using the Installer

The installer for Red Hat JBoss BPM Suite is an executable Java jar file. You can use it to install JBoss BPM Suite on an existing EAP 6.4 installation.



#### NOTE

For security reasons, you should run the installer as a **non-root** user.

#### Prerequisite

Before attempting to install JBoss BPM Suite, ensure you have already installed Red Hat JBoss EAP 6.4 or better.

#### 1. Setup Location and Users

Navigate to the folder where you downloaded the installer file in a command prompt and execute the following command.

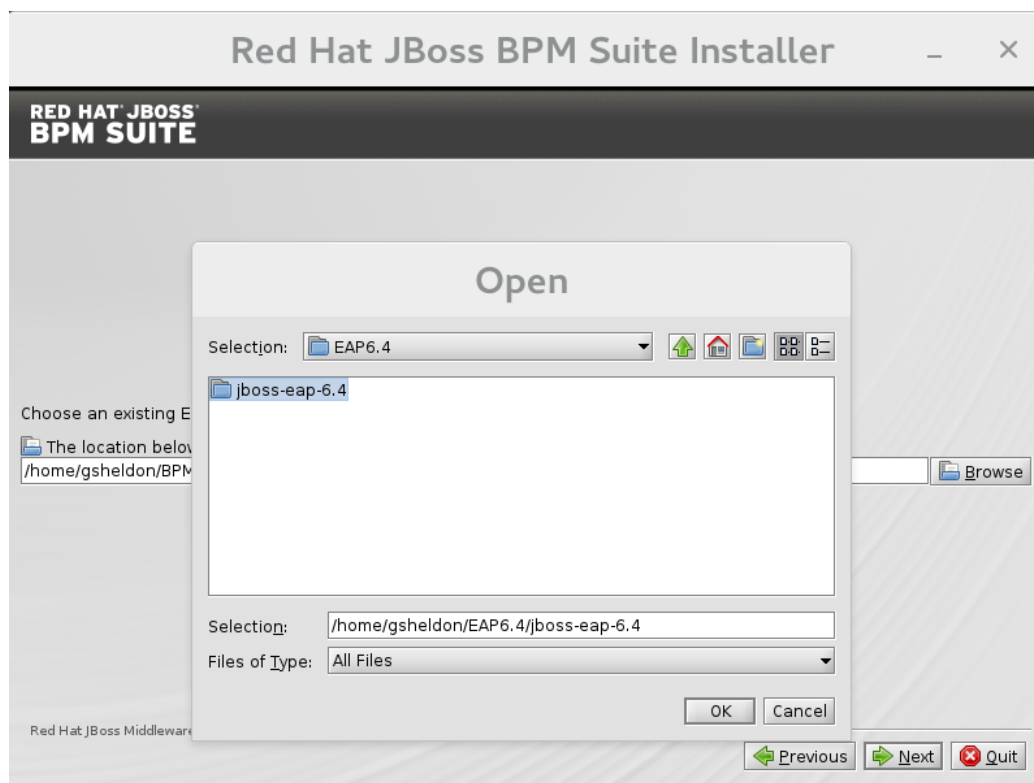
```
java -jar jboss-bpmsuite-6.1.0.GA-installer.jar
```



#### NOTE

When running the installer on Windows, you may be prompted to provide administrator credentials during the installation. To prevent this, add the `izpack.mode=privileged` option to the installation command: `java - Dizpack.mode=privileged -jar jboss-bpmsuite-6.1.0.GA-installer.jar`

- The graphical installer will execute and display a splash screen and a license agreement page. Accept the license to proceed.
- In the next screen, provide the parent location of an existing JBoss EAP where JBoss BPM Suite must be installed. The screenshot below depicts an example directory path:



**Figure 2.1. Red Hat JBoss BPM Suite for JBoss EAP Directory Path**

- In the next two screens, create two users: the first one for the management console of the EAP (ManagementRealm) and the second one for managing JBoss BPM Suite itself (ApplicationRealm).

Creation of the first user for the management console of JBoss EAP is optional and you may skip it if it is not required.

Make a note of these usernames and passwords as you will need them to access the JBoss EAP server (if you do decide to create it) and the JBoss BPM Suite application respectively.

Unless advanced configuration is performed, the JBoss BPM Suite user password will be used as the default password for both client and server JMS SSL keystores, as well as password vault keystores.



#### NOTE

The username that you create should *not* be the same as any of the pre-defined roles (See [Section 4.1, “Defining Roles”](#)).

The passwords that you create must have at least 8 characters and must contain at least one number and one non-alphanumeric character (not including the character &).

**NOTE**

The application role assigned to the second user that you create is the `admin` role. You can assign additional roles to this user at this stage.

**5. Setup Security Environment**

Next, you will setup the security environment of your new JBoss BPM Suite install. Decide to enable or disable the Java Security Manager in this step by clicking on the check box. The Java Security Manager makes your system more secure but may downgrade performance. You need to make a decision based on your environment.

- Choose whether you want to setup pure IPv6 configuration on the server that the installation is taking place. This will allow you to setup runtime IPv6 specific configurations later.

**7. Configure Runtime Environment**

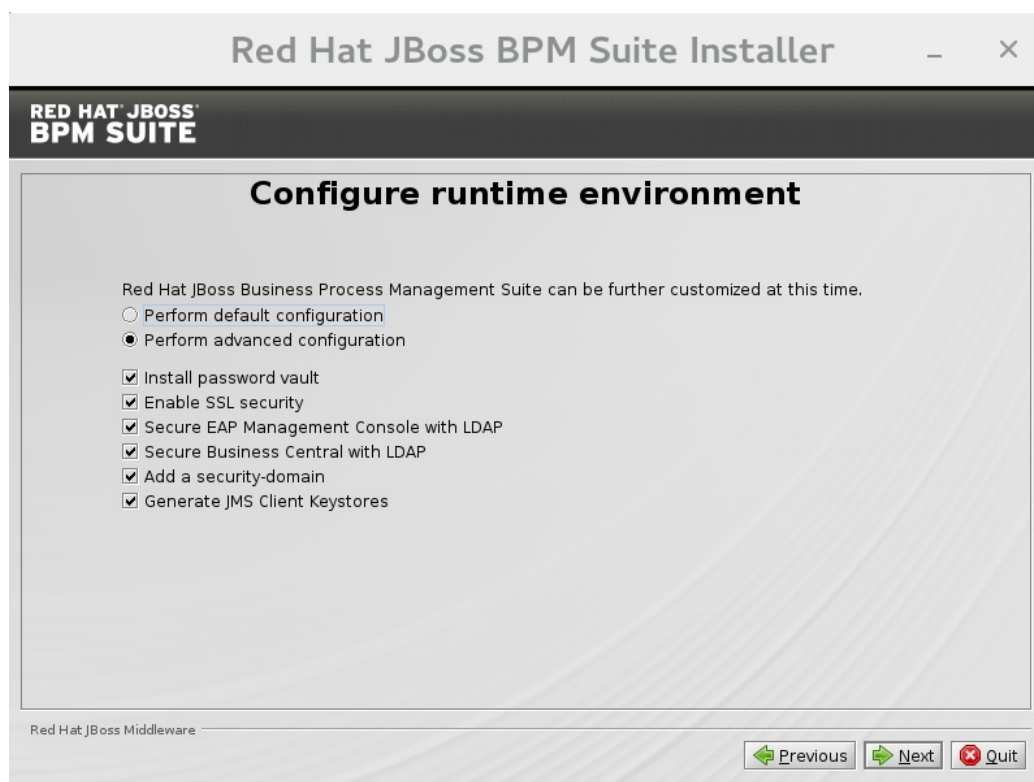
This step provides the option of using a default configuration or specifying an advanced configuration.

- o **Default Configuration**

Choose default configuration for the runtime environment in the next step and click **next** to review the installation details. If you are happy with the details, click **next** to start the actual installation or click **previous** to go back and make changes.

- o **Advanced Configuration**

Choose to enable advanced configuration options. Select "Perform advanced configuration" and choose the advanced configuration options you want to enable for your environment via the check boxes.



**Figure 2.2. Advanced Configuration Options**

- **Configure Vault Password**

Vault passwords are used to obfuscate passwords in the various server descriptors using a java secret key generated during the installation process, or manually using

the keytool. This prevents passwords from being stored as plain text in the descriptors. The *iteration count* and *salt* are both parameters to the encryption process.

In the case of JBoss BPM Suite, a vault is always installed, even if the user does not choose to install one with their own parameters. When this occurs, default values will be used.

For more information about vault passwords, see the *Red Hat JBoss EAP Security Guide*

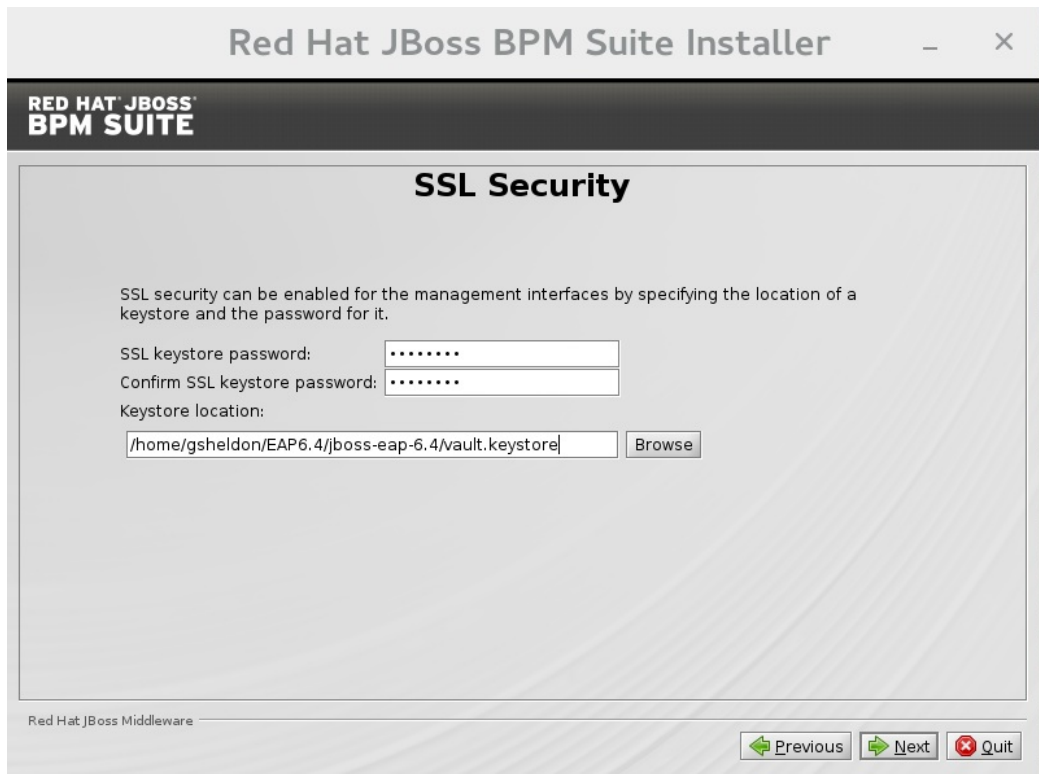
Figure 2.3. Configure vault password

#### ■ SSL Security

This screen allows you to add the `<ssl>` and `<truststore>` elements to the ManagementRealm security-realm using the provided keystore.

- The `<ssl>` element causes the server to present the certificate within the keystore as its identity, which allows the user to apply their official certificate.
- The `<truststore>` element enables "Client-Cert" authentication. This means that, if a remote client attempts to connect to any resource managed by the ManagementRealm, the client can present a certificate, and if an entry in the truststore matches, will be authenticated without needing to provide a username / password.

The end result is an encrypted connection that is secure between the client and the server for the ManagementRealm.



**Figure 2.4. SSL Security Configuration**

- **LDAP Security**

This step in the installer allows the user to define an LDAP server, which in turn defines users which should be allowed to authenticate with the ManagementRealm. This replaces the default configuration.

The **LDAP Connection** screen allows users to define how to connect to the LDAP server.

- The Distinguished Name(DN): the user that can connect to the LDAP server. Typically the DN will uniquely define a special user for this purpose.

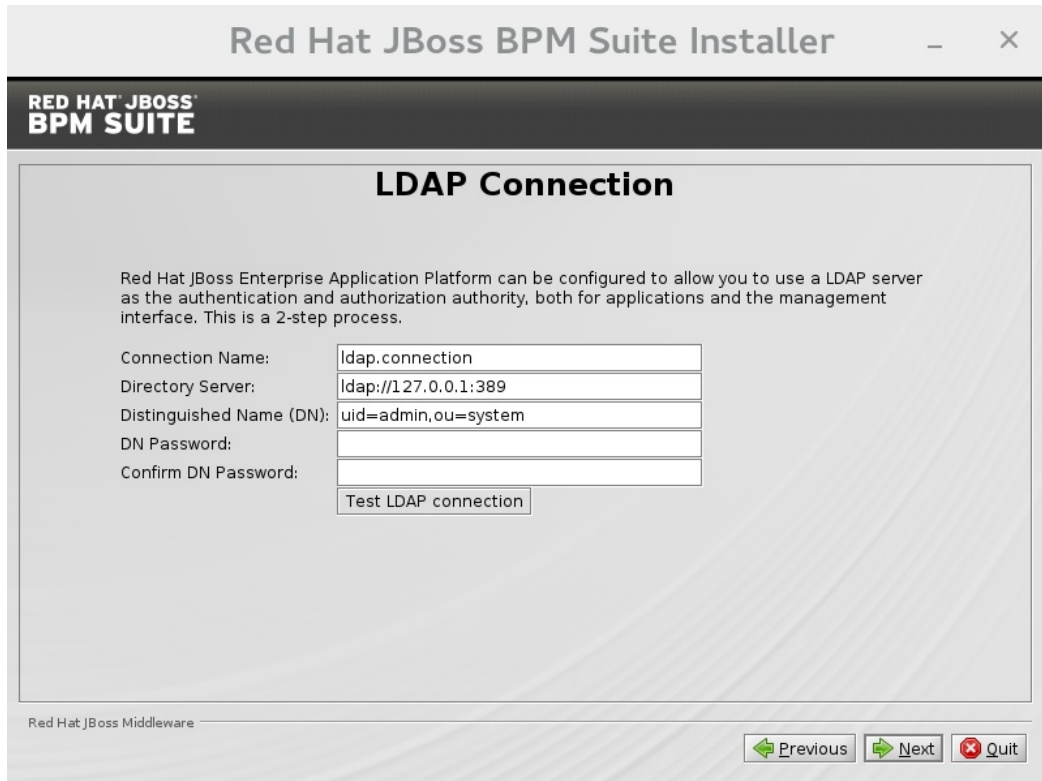


Figure 2.5. LDAP Connection Configuration

### LDAP Security (Management Console)

The **Management Console LDAP Configuration** screen allows you to set up a security realm. This defines the `<security-realm>` element to be added to the descriptors, and utilizes the connection defined previously.

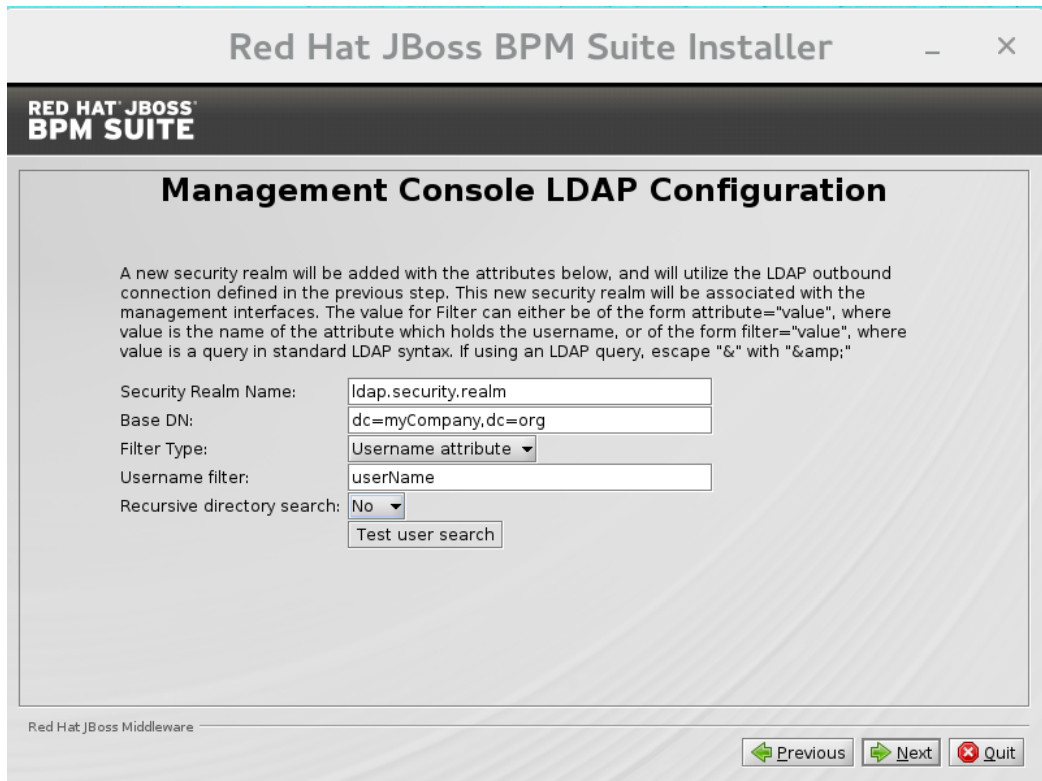


Figure 2.6. Management Console LDAP Configuration

- **Base DN:** Will typically define a 'base search' or 'root context' to begin searching for users.



- **Filter type:** Tells JBoss EAP how to find the LDAP attribute that defines a user; it can be a simple attribute, but can also be a complex "LDAP Filter".
- **Username attribute:** The LDAP attribute which holds the username values. A username entered in this field is used for search queries as a value of the 'uid' attribute. If a user chooses 'LDAP syntax query' as a filter type, this query must be specified in this field.
- **Recursive directory search:** If enabled, JBoss EAP will traverse the LDAP tree recursively, starting at Base DN. Otherwise, the search will be limited to Base DN.

### LDAP Security (Business Central)

Most of the following fields are similar to the Base DN. Contexts are used to search for roles, which allows it to perform authorization in addition to authentication. Otherwise, the `context` fields are analogous to the Base DN from the previous, and `attribute` fields are analogous to Username attribute. The filters allow fine grained control over which values of the given attribute will be accepted.

In JBoss BPM Suite, the `jbpm.usergroup.callback.properties` and `jbpm.user.info.properties` files used by `LDAPUserGroupInfo` and `LDAPUserInfo` components of Task Service, are also filled by values entered on the Business Central LDAP Configuration page.

Input values from Business Central LDAP Configuration page are used to configure a new security domain, which make use of `LdapExtended` login module. This security domain is set as default for Business Central web application. For more information about security domains and login modules, see the *Red Hat JBoss EAP Security Guide*

The screenshot shows a window titled "Red Hat JBoss BPM Suite Installer" with a sub-header "RED HAT JBOSS BPM SUITE". The main content area is titled "Business Central LDAP Configuration". Below the title, there is a paragraph of instructions: "Enter the details below to configure LDAP authentication on Business Central. The LDAP server URL, bind DN and bind password previously defined will be used. The contexts below define the users and roles that should be allowed access to Business Central. A security domain will be defined using these details." Below this text are several input fields: "User context:", "User filter:", "User roles context:", "User roles filter:", "Role context:", "Role filter:", "Role attribute id:", and "Role name id:". At the bottom left of the form area, there is a checkbox labeled "Role attribute is distinguished name". At the bottom right of the window, there are three buttons: "Previous", "Next", and "Quit".

Figure 2.7. Business Central LDAP Configuration

- **Security Domain and JSSE**

The Security Domain screen allows you to configure all of the elements of the `<security-domain>` security subsystem for managing security information,

including JSSE configuration. For more detailed information about configuring security domains, see the *Red Hat JBoss EAP Security Guide*

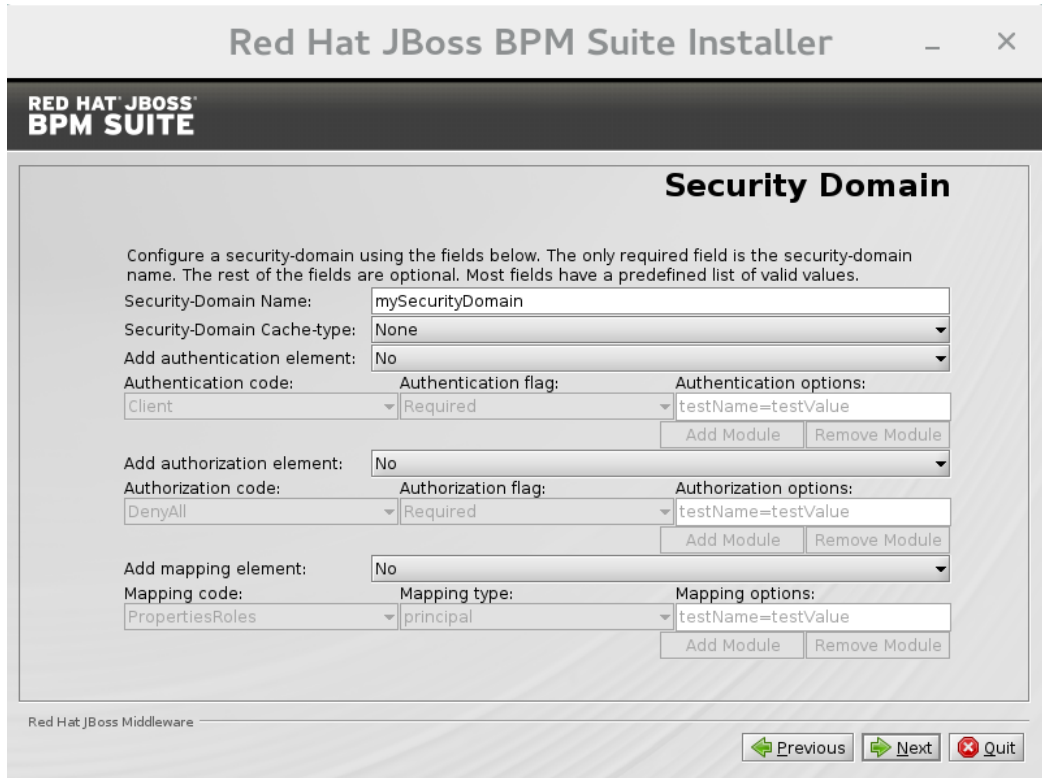


Figure 2.8. Security Domain

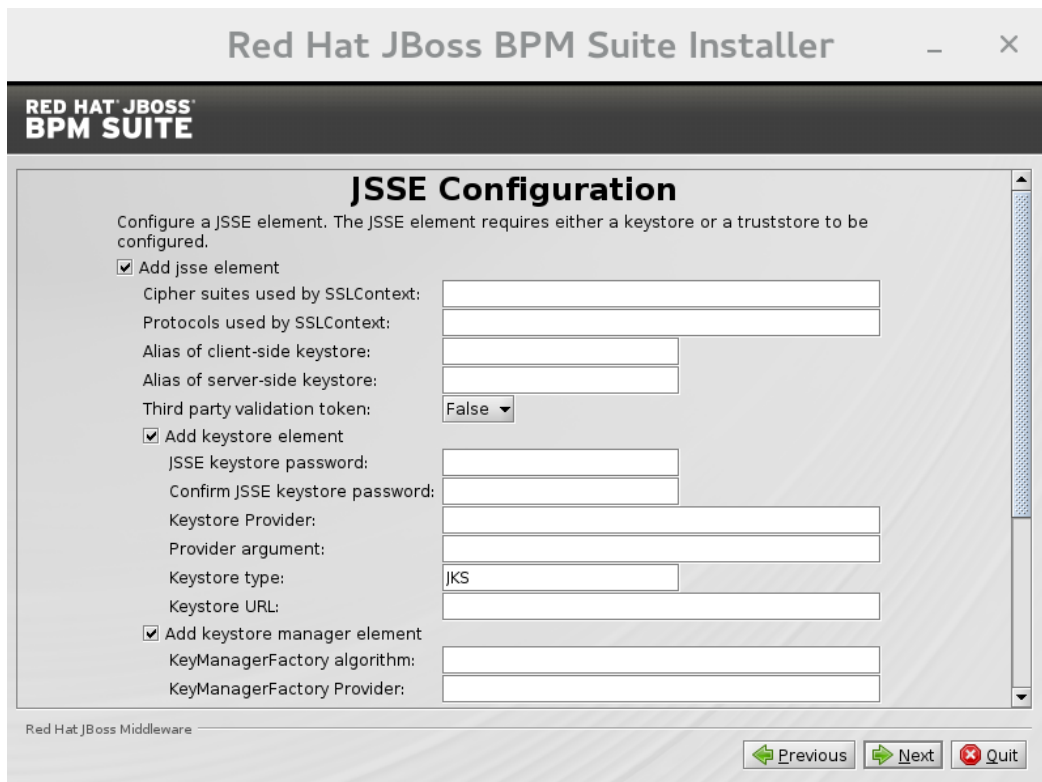
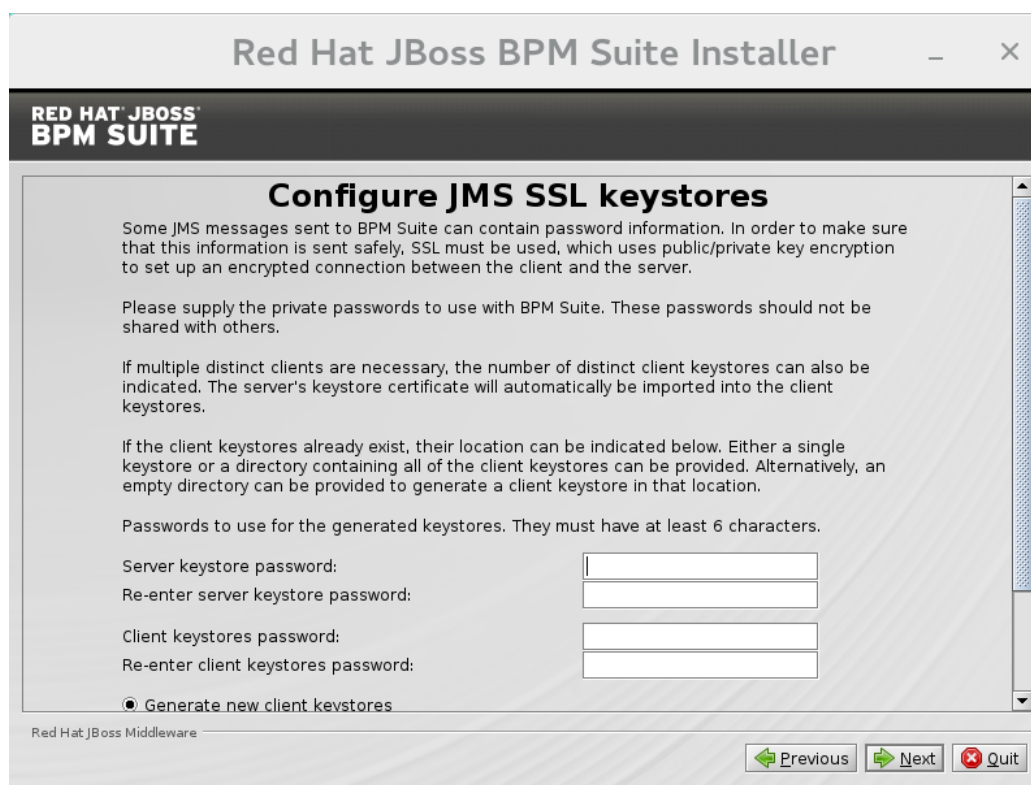


Figure 2.9. JSSE Configuration

- **Configure JMS SSL Keystores**

This screen allows the encryption of JMS messages sent to Business Central. The client keystores are distributed to systems that need to communicate with the server in order to facilitate encrypted communications. Users can use their pre-existing

keystores or generate new ones.



**Figure 2.10. Configure JMS SSL Keystores**

8. The installer will go through the steps to install JBoss BPM Suite and will perform post installation configuration steps when you click **next**. The installer will also start the JBoss BPM Suite server and connect to it to validate the installation. Click **next** to get to the last screen where you can generate the installation script and properties file. Click **done** to quit the installer.

You have successfully installed Red Hat JBoss BPM Suite using the installer.

### 2.1.3. Installing Red Hat JBoss BPM Suite using the Installer in CLI Mode

The installer for Red Hat JBoss BPM Suite can also be executed through the command-line interface (CLI). The procedure below demonstrates the steps that you are likely to encounter using this option to install JBoss BPM Suite.

#### Prerequisite

Before attempting to install JBoss BPM Suite, ensure you have already installed Red Hat JBoss EAP 6.4 or better.

1. Navigate to the folder where you downloaded the installer file in a command prompt and execute the following command.

```
java -jar jboss-bpmsuite-6.1.0.GA-installer.jar -console
```

2. The command-line interactive process will start and display the End-User license agreement. You will be prompted to select an option at the end of this license:

```
press 1 to continue, 2 to quit, 3 to redisplay.
```

- 
3. Enter 1 to begin the installation and type in the parent directory of an existing EAP installation.

The location below must specify the JBOSS\_HOME of an existing EAP installation.

```
[/home/user/BPMSuite-6.1.0/jboss-eap-6.4]
```

The installer will verify the location of the JBoss EAP installation at the provided location. Enter 1 to confirm and continue.

4. Optional: Create a user for the management console of JBoss EAP (Management Realm):

Create an administrative user

This user will be added to the host container's management realm for administrative purposes. It can be used to access the management console, the management CLI or other applications secured in this realm.

Enter 0 to skip creating a new administrative user or 1 to create one. If you do decide to create one, then follow these steps:

1. Enter a username:

```
Admin username: [admin]
```

2. Create and confirm a password for the user of the EAP management console:

The password must have at least 8 characters, and contain at least one number and one non-alphanumeric symbol (not including the character &).

```
Admin password: []
```

```
Confirm admin password: [*****]
```

After this user has been created successfully, continue to the next step.

5. Create a JBoss BPM Suite admin user

Create a Business Process Management Suite Admin User

Create a BPM Suite admin user. The user will be added to the ApplicationRealm, and can be used to access the Business Central Console. The User will be assigned the 'admin' application roles. The BPM Suite username cannot be any of the following: 'admin', 'analyst', 'user', 'manager' or 'developer'.

```
BPM Suite username: [bpmsAdmin]
```

6. Enter a username for this user and then create and confirm a password.

The password must have at least 8 characters, and contain at least one number and one non-alphanumeric symbol (not including the character &).

BPM Suite password: []

Confirm BPM Suite password: [\*\*\*\*\*]

- After the passwords have been entered and confirmed, you will be given an optional step to define other roles for this user (besides the `admin` role). Enter these roles in a comma separated list or just press enter to skip this part.

(Optional) You can add this user to additional roles that will be used for task management. These roles are custom named and used again when building your processes with human tasks. Add your custom named roles in a comma separated list below.

Additional user roles: []

- Configure the Java Security Manager by either pressing 1 to select it or 0 to deselect it.

Configure the Java Security Manager

A Java security manager offers JVM level security beyond what is provided by the application container. It enforces access rules at the JVM runtime based on one or more security policies.

This installer will place two security policies in the installation directory with the filenames 'security.policy' and 'kie.policy' regardless of choice. Those policies will be enabled at runtime if the option below is selected.

Please note that a security manager imposes a significant performance overhead when enabled. It is suggested the included policies be applied in production if user requirements call for a stronger measure than what is already provided by the application container's authentication and authorization mechanism.

Please see the JBoss Business Process Management Suite administrative documentation for further details and consideration.

[ ] Enable the Java security manager

Input 1 to select, 0 to deselect:

- After the Java Security Manager choice, choose an option from the prompt below:

press 1 to continue, 2 to quit, 3 to redisplay.

- Specify whether or not you are using IPv6.

IPv6 configuration

If this computer is using a pure IPv6 configuration, please check the box below. A pure IPv6 setup requires additional configuration

```
at runtime to ensure the proper bindings of the management and http
interfaces.
```

```
[ ] Enable pure IPv6 configuration
Input 1 to select, 0 to deselect:
```

After selecting or deselecting IPv6 configuration, select one of the following options:

```
press 1 to continue, 2 to quit, 3 to redisplay.
```

11. Configure the runtime environment by either choosing the default configuration or advanced options.

```
Configure runtime environment
Red Hat JBoss Business Process Management Suite can be further
customized at this time.
0 [x] Perform default configuration
1 [ ] Perform advanced configuration
```

If you select 1, "Perform advanced configuration," complete the following configurations:

- o [ ] Install password vault  
Input 1 to select, 0 to deselect:
- o [ ] Enable SSL security  
Input 1 to select, 0 to deselect:
- o [ ] Secure EAP Management Console with LDAP  
Input 1 to select, 0 to deselect:
- o [ ] Secure Business Central with LDAP  
Input 1 to select, 0 to deselect:
- o [ ] Add a security-domain  
Input 1 to select, 0 to deselect:
- o [ ] Generate JMS Client Keystores  
Input 1 to select, 0 to deselect:

12. Next, choose an option from the prompt below:

```
press 1 to continue, 2 to quit, 3 to redisplay.
```

13. The .jar file will begin to unpack and configure.

14. After a successful installation, the command-line will ask you if you would like to generate an automatic installation script and properties file.

```
Installation has completed successfully.
```

```
Application installed on /home/user/BPMSuite-6.1.0/jboss-eap-6.4
Would you like to generate an automatic installation script and
properties file?
(y/n) [n]:
```

15. If you select [ y ], provide a path for the automatic installation script:

```
Select path for the automatic installation script:
[/home/user/BPMSuite-6.1.0/jboss-eap-6.4/<auto script filename>]
```

This generated script will allow the user to run the installer in the following way for future installations:

```
java -jar jboss-bpmsuite-6.1.0.GA-installer.jar <auto script
filename>
```



#### NOTE

Running the installer in this way will result in an installation identical to the installation from which the auto script was generated. Note that sensitive values, such as passwords, will need to be provided from an external file or provided at auto installation time. The optional argument below allows the user to provide these values automatically:

```
-variablefile <variable filename>
```

Sensitive values can also be provided using the following argument:

```
-variables key1=value1, key2=value2
```

16. The command-line will provide the following message upon a successful auto script creation and/or console installation:

```
XML written successfully.
[ Console installation done ]
[BPMS_Installer]$
```

17. Start JBoss EAP by running `standalone.sh` in the `jboss-eap-6.4/bin` directory.

```
./standalone.sh
```

18. Navigate to <http://localhost:8080/business-central> in a web browser.

19. Login with the correct username/password as given to the JBoss BPM Suite user in the "Create and confirm a password for the JBoss BPM Suite user" step.

### 2.1.4. Installing Red Hat JBoss BPM Suite in the Domain Mode

To install the deployable package for JBoss EAP in the domain mode, do the following steps:

1. Download and extract the Red Hat JBoss BPM Suite 6.1.0 Deployable for EAP 6 ZIP file from [Red Hat Customer Portal](#) and copy the following directories into your local installation of EAP 6.4:

- o **bin**
- o **domain**

Skip the **standalone** directory.

2. On the command line, move to the **/domain** directory and start the domain:

In a Unix environment, run:

```
./domain.sh
```

In a Windows environment, run:

```
./domain.bat
```

3. Deploy the archive either via `${jboss-eap-home}/bin/jboss-cli.sh / ${jboss-eap-home}/bin/jboss-cli.bat`, or via management web UI (`localhost:9990/`):



#### NOTE

The web applications `business-central.war` and `dashbuilder.war` supplied in the EAP deployable binaries are directories, but for deployment into the domain, you have to use WAR archives. To create them, simply zip the content of the `business-central.war` and the `dashbuilder.war` directories.

- a. To deploy the archive via `${jboss-eap-home}/bin/jboss-cli.sh` or `${jboss-eap-home}/bin/jboss-cli.bat`, move into the `${jboss-eap-home}/bin` directory and deploy the WAR file:

In a Unix environment, run:

```
./jboss-cli.sh
```

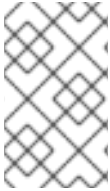
In a Windows environment, run:

```
./jboss-cli.bat
```

- b. To deploy the archive via management web UI (`localhost:9990/`):

- log in using your EAP management account
- select **Domain -> Manage Deployments -> Content Repository -> Add**
- select the web archive from the file system, upload the web archive
- select the deployment, click the **Assign** button
- select the server group



**NOTE**

In order to log in to Business Central deployed on Host Controller (HC) machines, the user created on the Domain Controller Machine has to be created on the Host Controller machines as well, by following the steps in the [Section 4.2, “Creating users”](#) section.

**Installing Multiple JBoss BPM Suite Server Instances**

In many situations, users may want to group together a set of EAP 6 nodes on the same machine and give them a meaningful name for easy maintenance. Unique values need to be incorporated for the system properties for each server instance. Listed below are the common properties that can be specified with a single JBoss BPM Suite node to change the default configuration; however, they should be specified for multiple nodes running on a single machine so every node can point to a different directory:

- `org.uberfire.nio.git.dir`
- `org.uberfire.metadata.index.dir`
- `org.uberfire.nio.git.ssh.cert.dir`

When multiple JBoss BPM Suite nodes are used on a single machine, the below properties need to be specified:

- `org.uberfire.nio.git.daemon.host` - can be left on default to bind to localhost.
- `org.uberfire.nio.git.daemon.port`
- `org.uberfire.nio.git.ssh.host` - can be left on default to bind to localhost.
- `org.uberfire.nio.git.ssh.port`

**NOTE**

Both the `org.uberfire.nio.git.daemon.port` and the `org.uberfire.nio.git.ssh.port` require different port values in order to avoid port conflicts.

Incorporate the previous properties in the `$EAP_HOME/domain/configuration/host.xml` file as illustrated in the two nodes below:

Node A:

```
<system-properties>
  <property name="org.uberfire.nio.git.dir" value="/valid/path/.." boot-
time="false"/>
  <property name="org.uberfire.metadata.index.dir"
value="/valid/path/.." boot-time="false"/>
  <property name="org.uberfire.nio.git.ssh.cert.dir"
value="/valid/path/.." boot-time="false"/>

  <property name="org.uberfire.nio.git.daemon.host" value="10.10.10.10"
boot-time="false"/>
  <property name="org.uberfire.nio.git.daemon.port" value="9417" boot-
time="false"/>
```

```
<property name="org.uberfire.nio.git.ssh.host" value="10.10.10.10"
boot-time="false"/>
  <property name="org.uberfire.nio.git.ssh.port" value="8002" boot-
time="false"/>
</system-properties>
```

**Node B:**

```
<system-properties>
  <property name="org.uberfire.nio.git.dir" value="/valid/path/.." boot-
time="false"/>
  <property name="org.uberfire.metadata.index.dir"
value="/valid/path/.." boot-time="false"/>
  <property name="org.uberfire.nio.git.ssh.cert.dir"
value="/valid/path/.." boot-time="false"/>

  <property name="org.uberfire.nio.git.daemon.host" value="10.10.10.10"
boot-time="false"/>
  <property name="org.uberfire.nio.git.daemon.port" value="9418" boot-
time="false"/>
  <property name="org.uberfire.nio.git.ssh.host" value="10.10.10.10"
boot-time="false"/>
  <property name="org.uberfire.nio.git.ssh.port" value="8003" boot-
time="false"/>
</system-properties>
```

The system properties depicted above should indicate the host, port, or location of the `.index` or `.niogit` files. These files, which should be used by a respective node, would then be grouped in a particular domain.

## 2.2. THE GENERIC DEPLOYABLE BUNDLE INSTALLATION

To install Red Hat JBoss BPM Suite on Red Hat JBoss Web Server (EWS), you need to use the generic deployable package of the product.

For installation on EWS, the generic deployable package contains additional transaction manager and security libraries that are not part of Red Hat JBoss EWS.

Note that to install the generic deployable package, you need the following ZIP files:

- **jboss-bpmsuite-VERSION-deployable-generic.zip**: contains the `business-central.war`, `dashbuilder.war` and `kie-server.war` web applications.
- **jboss-bpmsuite-VERSION-engine.zip**: supported execution engine libraries for embedding the engine into your application and other libraries needed for generic deployment.

### 2.2.1. Downloading the Generic Deployable Package

To download the generic deployable Red Hat JBoss BPM Suite package for JBoss Web Server, do the following:

1. Go to the [Red Hat Customer Portal](#) and log in.
2. Click on **Downloads**.

3. From the list of products click on **Red Hat JBoss BPM Suite**.
4. From the **Version** drop-down menu, select version 6.1 (if not already selected).
5. In the Software Downloads section that comes up, navigate to the **Red Hat JBoss BPM Suite 6.1 Deployable for All Supported Containers** row and then click **Download**.
6. Also navigate to the **Red Hat JBoss BPM Suite 6.1 Engine** files row and click **Download** to download the JBoss BPM Suite Engine files.

## 2.2.2. Installing the Generic Deployable Package

To install the generic deployable package, you need to set up the following after you have installed the underlying platform (Red Hat JBoss WS):

- Set up the database driver and the transaction manager (Bitronix) (refer to [Section 2.2.2.1, “Setting up Transaction Manager for Red Hat JBoss Web Server 2.1 \(Tomcat 7\)”](#)).
- Set up the Business Central application: set up users and roles and set up persistence (refer to [Section 2.2.2.2, “Setting up Business Central for Red Hat JBoss Web Server 2.1 \(Tomcat 7\)”](#) ).
- Set up the Dashbuilder application: set up users and roles and set up persistence (refer to [Section 2.2.2.3, “Setting up Dashbuilder for Red Hat JBoss Web Server 2.0 \(Tomcat 7\)”](#) ).

### 2.2.2.1. Setting up Transaction Manager for Red Hat JBoss Web Server 2.1 (Tomcat 7)

1. Extract the generic deployable zip package you downloaded from [Red Hat Customer Portal](#) to a temporary location. This zip package contains the following three web application archives: **business-central.war**, **dashbuilder.war** and **kie-server.war** in an exploded format. Rename these folders to remove the **.war** extension.
2. Copy these folders directly under the **\$TOMCAT\_DIR/webapps** folder.

You should end up with three folders in an exploded format:

**\$TOMCAT\_DIR/webapps/business-central**, **\$TOMCAT\_DIR/webapps/dashbuilder** and **\$TOMCAT\_DIR/webapps/kie-server**.



#### NOTE

**\$TOMCAT\_DIR** stands for the home directory where your web server is located. Replace it with the actual path to your web server home directory, for example: **/home/john/jboss-ews-2.1/tomcat7/**

3. Extract the contents of the JBoss BPM Suite Engine files archive to a temporary location from where you can copy the required libraries. This folder now contains all the core JBoss BPM Suite libraries under the extracted folder and a **lib** folder.
4. **Install the transaction manager.**

**WARNING**

Please note that the following section describes the setup of a transaction manager, Bitronix that is not officially supported by Red Hat.

Copy the following transaction manager jar libraries from the `lib` folder to `$TOMCAT_DIR/lib/` directory:

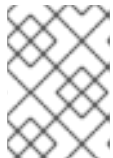
- o `btm-VERSION.jar`
- o `btm-tomcat55-lifecycle-VERSION.jar`
- o `jta-VERSION.jar`
- o `slf4j-api-VERSION.jar`
- o `slf4j-jdk14-VERSION.jar`

In addition, download the following library and copy it into the `$TOMCAT_DIR/lib/` folder as well:

- o [javax.security.jacc-api.jar](#)

**5. Install the Driver to Your Database**

Copy the jar file with the relevant database driver to `$TOMCAT_DIR/lib/`.

**NOTE**

If using the embedded H2 database, the driver is available in `business-central/WEB-INF/lib/`.

**6. Create the transaction manager configuration files in `$TOMCAT_DIR/conf/`:**

- o `btm-config.properties`

```
bitronix.tm.serverId=tomcat-btm-node0
bitronix.tm.journal.disk.logPart1Filename=${btm.root}/work/btm1.t
log
bitronix.tm.journal.disk.logPart2Filename=${btm.root}/work/btm2.t
log
bitronix.tm.resource.configuration=${btm.root}/conf/resources.pro
perties
```

- o `resources.properties` (the `resource.ds1.uniqueName` defines the datasource name used in tomcat resource definition later - make a note of this value).

Make sure to change the values in the following definitions to match your environment.

**Example 2.1. H2 datasource definition**

■

```

resource.ds1.className=bitronix.tm.resource.jdbc.lrc.LrcXADataSource
resource.ds1.uniqueName=jdbc/jbpm
resource.ds1.minPoolSize=10
resource.ds1.maxPoolSize=20
resource.ds1.driverProperties.driverClassName=org.h2.Driver
resource.ds1.driverProperties.url=jdbc:h2:file:~/jbpm
resource.ds1.driverProperties.user=sa
resource.ds1.driverProperties.password=
resource.ds1.allowLocalTransactions=true

```

### Example 2.2. MySQL 5.5 datasource definition

```

resource.ds1.className=com.mysql.jdbc.jdbc2.optional.MysqlXADataSource
resource.ds1.uniqueName=jdbc/jbpm
resource.ds1.minPoolSize=0
resource.ds1.maxPoolSize=10
resource.ds1.driverProperties.URL=jdbc:mysql://localhost:3306/sampledb
resource.ds1.driverProperties.user=dbuser
resource.ds1.driverProperties.password=dbpassword
resource.ds1.allowLocalTransactions=true

```

### Example 2.3. DB2 Type 4 datasource definition

```

resource.ds1.className=com.ibm.db2.jcc.DB2Driver
resource.ds1.uniqueName=jdbc/jbpm
resource.ds1.minPoolSize=0
resource.ds1.maxPoolSize=10
resource.ds1.driverProperties.URL=jdbc:db2://localhost:50000/sampledb
resource.ds1.driverProperties.user=dbuser
resource.ds1.driverProperties.password=dbpassword
resource.ds1.allowLocalTransactions=true

```

### Example 2.4. Oracle datasource definition

```

resource.ds1.className=oracle.jdbc.xa.client.OracleXADataSource
resource.ds1.uniqueName=jdbc/jbpm
resource.ds1.minPoolSize=0
resource.ds1.maxPoolSize=10
resource.ds1.driverProperties.URL=jdbc:oracle:thin:@//localhost:1521/bpms
resource.ds1.driverProperties.user=dbuser
resource.ds1.driverProperties.password=dbpassword
resource.ds1.allowLocalTransactions=true

```

**Example 2.5. Microsoft SQL Server datasource definition**

```
resource.ds1.className=com.microsoft.sqlserver.jdbc.SQLServerDriver
resource.ds1.uniqueName=jdbc/jbpm
resource.ds1.minPoolSize=0
resource.ds1.maxPoolSize=10
resource.ds1.driverProperties.URL=jdbc:sqlserver://localhost:1433;databaseName=bpms;
resource.ds1.driverProperties.user=dbuser
resource.ds1.driverProperties.password=dbpassword
resource.ds1.allowLocalTransactions=true
```

7. Set up the transaction manager listener in `$TOMCAT_DIR/conf/server.xml` to start and stop Bitronix on container startup and shutdown:

Add the following element as the last `<Listener>` element into the `<Server>` element:

```
<Listener
  className="bitronix.tm.integration.tomcat55.BTMLifecycleListener" />
```

8. Define the `btm.root` system property and location where bitronix config file is placed:

In `$TOMCAT_DIR/bin/`, create a readable `setenv.sh` file with the following content:

```
CATALINA_OPTS="-Xmx512M -XX:MaxPermSize=512m -
Djava.security.auth.login.config=$CATALINA_HOME/webapps/business-
central/WEB-INF/classes/login.config -Dbtm.root=$CATALINA_HOME -
Dbitronix.tm.configuration=$CATALINA_HOME/conf/btm-config.properties
-Dorg.jbpm.designer.perspective=RuleFlow -
Djbpm.tsr.jndi.lookup=java:comp/env/TransactionSynchronizationRegistry"
```

The property `org.jbpm.designer.perspective` is set to `RuleFlow` to allow the default perspective for the designer to be `RuleFlow` rather than `Full`. Grant the file execute permissions if applicable.

The `java.security.auth.login.config` property must be set in order for the ssh clone of the git repository to work.

**IMPORTANT**

On Microsoft Windows systems, replace the `$CATALINA_HOME` value in the content of the file with the equivalent environment variable name or use the absolute path and add the values in `setenv.bat` file as shown here in the following example:

```
set "CATALINA_OPTS=-Xmx512m -XX:MaxPermSize=512m -
Dbtm.root=C:\apache-tomcat -
Dbitronix.tm.configuration=C:\apache-tomcat\conf\btm-
config.properties"
```

### 2.2.2.2. Setting up Business Central for Red Hat JBoss Web Server 2.1 (Tomcat 7)

To set up Business Central, do the following:

1. Set up a Valve so that the Business Central web application can load the users set up in Tomcat:
  - a. Define users and roles in `$TOMCAT_DIR/conf/tomcat-users.xml`. Note that Business Central requires users to have the roles specified as `admin` and/or `analyst` (for more information about user and role definitions, refer to the Tomcat 7 documentation).

The program listing below shows an example of how these two roles would be added and how a user named `bpmsadmin` will be assigned these roles.



#### NOTE

Make sure that the usernames don't conflict with any known roles. For example, you should not create a user with the username `admin`.

```
<role rolename="admin"/>
<role rolename="analyst" />
<user username="bpmsadmin" password="P@ssw0rd"
roles="admin, analyst"/>
```

- b. Move (not copy) `kie-tomcat-integration-VERSION.jar` from `$TOMCAT_DIR/webapps/business-central/WEB-INF/lib/` to `$TOMCAT_DIR/lib/`.
- c. Copy `jboss-jaxb-api-VERSION.jar` from `$TOMCAT_DIR/webapps/business-central/WEB-INF/lib/` to `$TOMCAT_DIR/lib/`.
- d. In `$TOMCAT_DIR/conf/server.xml`, add the Tomcat Valve declaration in the relevant `<host>` element:

```
<Valve className="org.kie.integration.tomcat.JACCValve" />
```

2. If you are using a datasource other than the default provided by the underlying H2 database, you will need to setup persistence. If you are using the default H2 database, then you can ignore the rest of the steps in this procedure.

In this procedure, you configure a datasource with the JNDI name `jdbc/myDatasource` as defined in `uniqueName=jdbc/jbpm` in the bitronix `resources.properties` file earlier (for the MySQL option).

- a. In `business-central/META-INF/context.xml`, replace the datasource JNDI name in the `<Resource>` element. The `uniqueName` attribute refers to the `resource.ds1.uniqueName` property set in `resources.properties`:

```
<Resource name="jdbc/myDatasource" uniqueName="jdbc/jbpm"
auth="Container" removeAbandoned="true"
factory="bitronix.tm.resource.ResourceObjectFactory"
type="javax.sql.DataSource"/>
```

- b. In **business-central/WEB-INF/web.xml**, replace the datasource JNDI name in the `<res-ref-name>` element with your datasource name:

```
<resource-ref>
    <description>Console DS</description>
    <res-ref-name>jdbc/myDatasource</res-ref-name>
    <res-type>javax.sql.DataSource</res-type>
    <res-auth>Container</res-auth>
</resource-ref>
```

- c. Change **business-central/WEB-INF/classes/META-INF/persistence.xml**.

In this file, change the name of the hibernate dialect used for your database, if using a different database other than H2. The code below demonstrates the original database information for **persistence.xml**:

```
<property name="hibernate.dialect"
value="org.hibernate.dialect.H2Dialect"/>
```

This information can be updated in the following manner (as demonstrated with MySQL database below):

```
<property name="hibernate.dialect"
value="org.hibernate.dialect.MySQLDialect"/>
```



#### NOTE

The dialect for DB2 is `org.hibernate.dialect.DB2Dialect`, for DB2 on AS/400 is `org.hibernate.dialect.DB2400Dialect`, for Oracle is `org.hibernate.dialect.Oracle10gDialect` and for Microsoft SQL Server is `org.hibernate.dialect.SQLServerDialect`

- d. Change **business-central/WEB-INF/classes/META-INF/persistence.xml** file so that JBoss BPM Suite process engine can use the new database.

The code below demonstrates the original datasource information for **persistence.xml**:

```
<jta-data-source>java:comp/env/jdbc/jbpm</jta-data-source>
```

Change this value to the datasource defined earlier:

```
<jta-data-source>java:comp/env/jdbc/myDatasource</jta-data-source>
```

3. You can now start the JBoss Web Server to login to Business Central.

- a. Run **startup.sh** in the `$TOMCAT_HOME/bin` directory.

```
./startup.sh
```

- b. Navigate to <http://localhost:8080/business-central> in a web browser.



- c. Login with the username/password you defined earlier in `tomcat-users.xml` file.

### 2.2.2.3. Setting up Dashbuilder for Red Hat JBoss Web Server 2.0 (Tomcat 7)



#### NOTE

Before setting up Dashbuilder on Red Hat JBoss Web Server, you must ensure that you have correctly installed and started Business Central as described in [Section 2.2.2.2, “Setting up Business Central for Red Hat JBoss Web Server 2.1 \(Tomcat 7\)”](#). Dashbuilder requires the history log database tables to exist, which are only provided by Business Central. If these tables are not present in the database before attempting the steps below, you may get initialization errors.

To set up Dashbuilder on Red Hat JBoss Web Server, do the following:

1. Define users and roles in `$TOMCAT_DIR/conf/tomcat-users.xml`. Note that Dashbuilder requires users to have the role specified as `admin` and/or `analyst`. If you have already defined these users earlier for Business-Central, you don't need to define them again.
2. Enable single sign-on between Dashbuilder and Business Central by uncommenting the following lines in `$TOMCAT_DIR/conf/server.xml` file:

```
<Valve className="org.apache.catalina.authenticator.SingleSignOn" />
```

3. As with Business Central setup, if you are using a database other than the default and integrated H2 database, you will need to setup persistence.

In this procedure, you configure a datasource with the JNDI name `jdbc/dashbuilderDS` as defined in `uniqueName=jdbc/jbpm` in the bitronix `resources.properties` file:

- a. In `dashbuilder/META-INF/context.xml`, replace the datasource JNDI name in the `<Resource>` element. The `uniqueName` attribute refers to the `resource.ds1.uniqueName` property set in `resources.properties`:

```
<Resource name="jdbc/dashbuilderDS" uniqueName="jdbc/jbpm"
auth="Container" removeAbandoned="true"
factory="bitronix.tm.resource.ResourceObjectFactory"
type="javax.sql.DataSource"/>
```

**NOTE**

Depending upon your database, you may need to define some other properties here as well. For example, in an Oracle environment, this entry may look like the following listing.

```
<Resource name="jdbc/jbpm" uniqueName="jdbc/jbpm"
auth="Container" removeAbandoned="true"
factory="bitronix.tm.resource.ResourceObjectFactory"
type="javax.sql.DataSource" username="username"
password="password"
driverClassName="oracle.jdbc.xa.client.OracleXADataSou
rce" url="jdbc:oracle:thin:YOUR-URL:1521:YOUR-DB"
maxActive="8" />
```

- b. In **dashbuilder/WEB-INF/web.xml**, add the datasource JNDI name in the **<res-ref-name>** element with your datasource name:

```
<resource-ref>
  <description>Dashboard Builder Datasource</description>
  <res-ref-name>jdbc/dashbuilderDS</res-ref-name>
  <res-type>javax.sql.DataSource</res-type>
  <res-auth>Container</res-auth>
</resource-ref>
```

- c. In **dashbuilder/META-INF/context.xml**, define the transaction factory:

```
<Transaction
factory="bitronix.tm.BitronixUserTransactionObjectFactory"/>
```

- d. Update the datasource JNDI name in **dashbuilder/WEB-INF/etc/hibernate.cfg.xml** in the **<session-factory>** element:

```
<property
name="connection.datasource">java:/comp/env/jdbc/dashbuilderDS</p
roperty>
```

4. Restart Java Web server for these changes to take effect. Once restarted, you can navigate to Dashbuilder from within Business Central or directly via:  
**http://localhost:8080/dashbuilder.**

## CHAPTER 3. SPECIAL SETUPS

### 3.1. SETTING UP PERSISTENCE FOR BUSINESS CENTRAL

Red Hat JBoss BPM Suite is configured to use an example datasource with Java Naming and Directory Interface (JNDI) name `java:jboss/datasources/ExampleDS`. For Business Central, this example datasource is located in the file `business-central.war/WEB-INF/classes/META-INF/persistence.xml`.

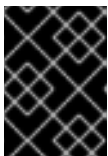
If you want to configure JBoss BPM Suite to use an external database, make the following changes. For file `business-central.war/WEB-INF/classes/META-INF/persistence.xml`:

1. Install the respective Java Database Connectivity (JDBC) driver using modular approach for easy subsequent configuration (see [EAP 6 documentation](#)).
2. Create a new datasource according to the example in [EAP 6 documentation, section 6.7.1. Example PostgreSQL Datasource](#). This is the default used H2 database specific datasource configuration:

```
<subsystem xmlns="urn:jboss:domain:datasources:1.1">
  <datasources>
    <datasource jndi-name="java:jboss/datasources/ExampleDS"
pool-name="ExampleDS" enabled="true" use-java-context="true">
      <connection-url>jdbc:h2:mem:test;DB_CLOSE_DELAY=-
1</connection-url>
      <driver>h2</driver>
      <security>
        <user-name>sa</user-name>
        <password>sa</password>
      </security>
    </datasource>
    <drivers>
      <driver name="h2" module="com.h2database.h2">
        <xa-datasource-
class>org.h2.jdbcx.JdbcDataSource</xa-datasource-class>
      </driver>
    </drivers>
  </datasources>
```

3. Use the JNDI name of the datasource to update the following entry inside the `persistence.xml` file, which is by default set to this entry.

```
<jta-data-source>java:jboss/datasources/ExampleDS</jta-data-source>
```



#### IMPORTANT

When configuring your datasource, make sure to enable JTA (typically, by adding `jta="true"` to the `datasource` tag).

4. Replace the following text with the appropriate database specific hibernate dialect name.

```
<property name="hibernate.dialect"
value="org.hibernate.dialect.H2Dialect" />
```

For example, for an Oracle Database Express Edition 11g, it would be:

```
<property name="hibernate.dialect"
value="org.hibernate.dialect.Oracle10gDialect" />
```

## NOTE

When JBoss BPM Suite uses Oracle 11 as the datasource, multiple warning (**WARN**) messages are produced in the logs, without any corresponding Business Central activity being performed. This is expected behavior. To turn off these messages, set the level of the `org.hibernate.loader` category of the logger to **ERROR** in the `standalone.xml` file:

```
<logger category="org.hibernate.loader">
  <level name="ERROR"/>
</logger>
```

## 3.2. SETTING UP PERSISTENCE FOR DASHBUILDER

As Dashbuilder is dependent on the configuration of Business Central, ensure Business Central is configured according to [Section 3.1, “Setting up Persistence for Business Central”](#). Red Hat JBoss BPM Suite 6 is configured to use a datasource with Java Naming and Directory Interface (JNDI) name `java:jboss/datasources/ExampleDS`. If you want to make the application work with a database different from H2, for example Oracle, MySQL, Postgres, or MS SQL Server, follow these steps.

## NOTE

If you want to use UTF 8 to display non-English characters, you should check your database documentation and set up the encoding at the level of database in order for Dashbuilder to work correctly. For example in MySQL, add the following to the server configuration file:

```
[mysqld]
character-set-server=utf8
collation-server=utf8_general_ci
```

## IMPORTANT

On Unix like systems override the default value of MySQL `lower_case_table_names` from `0` (case sensitive) to `1` (case insensitive). The JBoss BPM Suite KPI queries are written in lowercase but the table names are in camelCase. By changing the `lower_case_table_names` property you will prevent exceptions from occurring later on.

1. Install the database driver and create a new datasource according to the example in [EAP 6 Documentation in section 6.7.1. Example PostgreSQL Datasource](#). Use modular approach to the installation of JDBC driver for easy subsequent configuration.
2. Create an empty database.
3. Modify file `dashbuilder.war/WEB-INF/jboss-web.xml` whose default entry is:

```
<jboss-web>
```

```

<context-root>/dashbuilder</context-root>
<resource-ref>
  <res-ref-name>jdbc/dashbuilder</res-ref-name>
  <res-type>javax.sql.DataSource</res-type>
  <jndi-name>java:jboss/datasources/ExampleDS</jndi-name>
</resource-ref>

```

4. Modify also files `WEB-INF/jboss-deployment-structure.xml` from both the `business-central.war` and `dashbuilder.war` applications, and add a dependency under the dependencies section on the JDBC driver module created earlier during datasource creation. The following snippet shows a sample configuration where `jdbcDriverModuleName` is the name of the JBoss EAP 6 JDBC driver module.

```

<dependencies>
  ...
  <module name="jdbcDriverModuleName" />
  ...
</dependencies>

```

### 3.3. SPECIAL SETUP FOR IBM DB2 DATABASE

If you want to use an IBM DB2 database as the underlying data source for Business Central, you will need to increase the page size for the database. The default page size of 4 kB is not sufficient for the Dashbuilder table columns size.

When creating the database, force the page size to 16384 as in the example below:

#### Example 3.1. Adjusting page size

```
CREATE DATABASE dashb PAGESIZE 16384
```

This increase in page size for the underlying database must be performed before the BPM Suite has been run for the first time.

## CHAPTER 4. ROLES AND USERS

### 4.1. DEFINING ROLES

Before starting the server and logging onto Business Central, you will need to create some user accounts. This section describes the different user roles that are used in Red Hat JBoss BPM Suite :

- **admin:** The users with admin role are the administrators of the application. Administrators can manage users, manage the repositories (create and clone) and have full access to make the required changes in the application. Admins have access to all areas within the system.
- **developer:** A developer has access to almost all features and can manage rules, models, process flows, forms and dashboards. They can manage the asset repository, they can create, build and deploy projects and they can even use Red Hat JBoss Developer Studio to view processes. Only certain administrative functions like creating and cloning a new repository are hidden for the developer role.
- **analyst:** An analyst role has access to all high-level features to model and execute their projects. However, **Authoring** → **Administration** access is unavailable to users with the analyst role. Certain lower-level features targeted towards developers, like the **Deployment** → **Artifact Repository** view are not accessible for this role. However, the **Build & Deploy** button is available for the analyst role while using the Project Editor.
- **user:** User or a business user work on the business task lists that are used to operate a certain process. A user with this role can access the dashboard and manage processes.
- **manager:** A manager is a viewer of the system and is interested in statistics around the business processes and their performance, business indicators, and other reporting of the system. A user with this role has access to the BAM only.



#### NOTE

Enter the above mentioned roles during the user creation process.

### 4.2. CREATING USERS

To start adding new users, you will need to run the `add-user.sh` script on a Unix system or the `add-user.bat` file on a Windows system from the EAP bin directory.

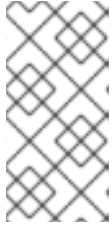
1. Run `./add-user.sh` on a Unix system or `add-user.bat` on a Windows system from the bin directory.
2. Enter `b` to select an Application User at the type of user prompt and press Enter.
3. Accept the default Realm (ApplicationRealm): by pressing Enter.
4. At the username prompt, enter a user name and confirm. For example: `helloworlduser`.



#### NOTE

Make sure that the usernames don't conflict with any known groups. For example, if there is a group called `admin`, you should not create a user with the username `admin`.

5. Create the user's password at the password prompt and reenter the password. For example: `HelloWorld@123`.

**NOTE**

The password should be at least 8 characters in length and should contain upper and lower case alphabetic characters (e.g. A-Z, a-z), at least one numerical character (e.g. 0-9) and at least one special character (e.g. ~ ! @ # \$ % ^ \* ( ) - \_ + =).

6. Enter a comma separate list of roles the user will need at the roles prompt (refer to [Section 4.1, “Defining Roles”](#)).

Business Central users need to have at least the `analyst` role, and dashbuilder users need to have the `admin` role. Roles should be entered as a comma-separated list.

7. Confirm you want to add the user.
8. Enter yes at the next prompt (this is to enable clustering in the future if required).

## CHAPTER 5. TESTING THE INSTALLATION

### 5.1. STARTING THE SERVER

If you have installed Red Hat JBoss BPM Suite using the JBoss EAP 6 bundle install, you can now start your server in one of two modes.



#### WARNING

In addition to the previous install steps, we strongly recommend to set the system property `org.kie.tx.lock.enabled` to `false` in order to prevent unresponsiveness or deadlock issues. You can either start your server with the option `-Dorg.kie.tx.lock.enabled=false` or edit the `standalone.xml` file:

```
<system-properties>
  <property name="org.kie.tx.lock.enabled" value="false"/>
  ...
</system-properties>
```

For further details, please refer to this article:  
<https://access.redhat.com/solutions/1610723>.



#### NOTE

If you installed JBoss BPM Suite using the generic deployable version on Red Hat Java Web Server, the instructions for download and install also contain the instructions for starting the server. You can ignore the following discussion.

The default startup script, `standalone.sh` that Red Hat JBoss BPM Suite ships with is optimized for performance. To run your server in the performance mode, do the following:

1. On the command line, move into the `$SERVER_HOME/bin/` directory.
2. In a Unix environment run:

```
./standalone.sh
```

In a Windows environment run:

```
./standalone.bat
```

Red Hat JBoss BPM Suite also ships with a separate script, `standalone-secure.sh` that is optimized for security. This script applies a security policy by default that protects against a known security vulnerability.



**NOTE**

It is recommended that production environments use `standalone-secure.sh` script.

**WARNING**

The use of a security manager imposes a significant performance penalty that you should be aware of. The tradeoff between security and performance must be made by taking into consideration individual circumstances. See [Section 5.2, “Java Security Manager and Performance Management”](#).

To run your server in the secure mode with this script, do the following:

1. On the command line, move into the `$SERVER_HOME/bin/` directory.
2. In a Unix environment run:

```
./standalone-secure.sh
```

In a Windows environment run:

```
./standalone-secure.bat
```

**NOTE**

If you installed JBoss BPM Suite using the installer, an option to apply the security policy is given to you at the time of install. The installer doesn't provide a separate `standalone-secure.sh` script.

**NOTE**

If you are starting the server in the domain mode, the corresponding scripts are `domain.sh` and `domain-secure.sh` respectively.

## 5.2. JAVA SECURITY MANAGER AND PERFORMANCE MANAGEMENT

As noted earlier, enabling the Java Security Manager (JSM) to sandbox the evaluation of MVEL scripts in Red Hat JBoss BPM Suite introduces a performance hit in high load environments. Environments and performance markers must be kept in mind when deploying a JBoss BPM Suite application. Use the following guidelines to deploy secure and high performance JBoss BPM Suite applications.

- In high load environments where performance is critical it is recommended to only deploy applications that have been developed on other systems and properly reviewed. It is also recommended not to create any users with Analyst role on such systems. If these safeguards are followed, it is safe to leave JSM disabled on these systems so it does not introduce any performance degradation.

- In testing and development environments without high loads, or in environments where rule and process authoring is exposed to external networks, it is recommended to have JSM enabled in order to achieve security benefits of properly sandboxed evaluation of MVEL.

Allowing users with Analyst role to log in to the Business Central console with JSM disabled is not secure and not recommended.

### 5.3. LOGGING ON TO BUSINESS CENTRAL

Log into Business Central after the server has successfully started.

1. Navigate to <http://localhost:8080/business-central> in a web browser. If the user interface has been configured to run from a domain name, substitute `localhost` for the domain name. For example <http://www.example.com:8080/business-central>.
2. Log in with the user credentials that were created during installation. For example: User = `helloworlduser` and password = `HelloWorld@123`.

## CHAPTER 6. CLUSTERING

When clustering Red Hat JBoss BPM Suite, consider which components need to be clustered. You can cluster the following:

- **GIT repository:** virtual-file-system (VFS) repository that holds the business assets so that all cluster nodes use the same repository
- **Execution Server and Web applications:** the runtime server that resides in the container (such as, Red Hat JBoss EAP) along with BRMS and BPM Suite web applications so that nodes share the same runtime data.

For instructions on clustering the application, refer to the container clustering documentation.

- **Back-end database:** database with the state data, such as, process instances, KIE sessions, history log, etc., for fail-over purposes

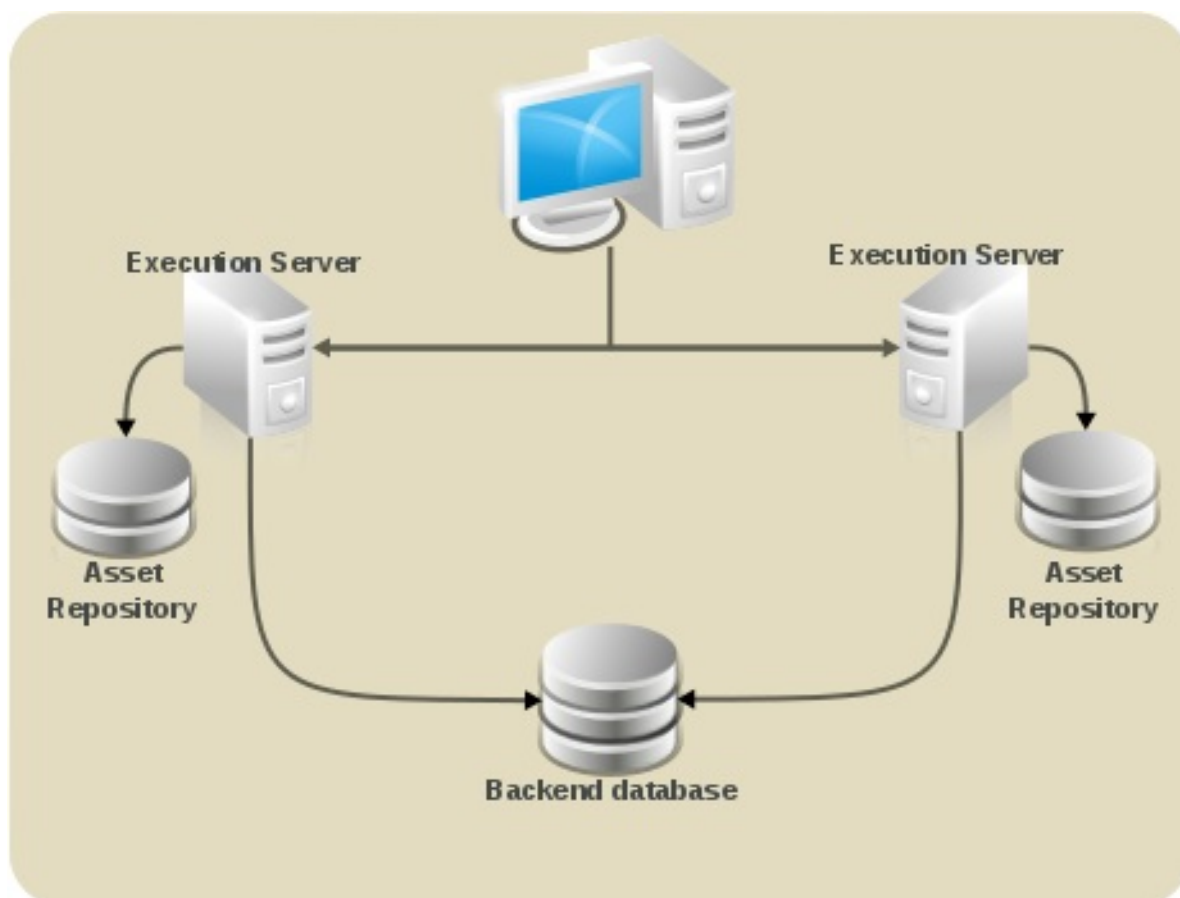


Figure 6.1. Schema of Red Hat JBoss BPM Suite system with individual system components

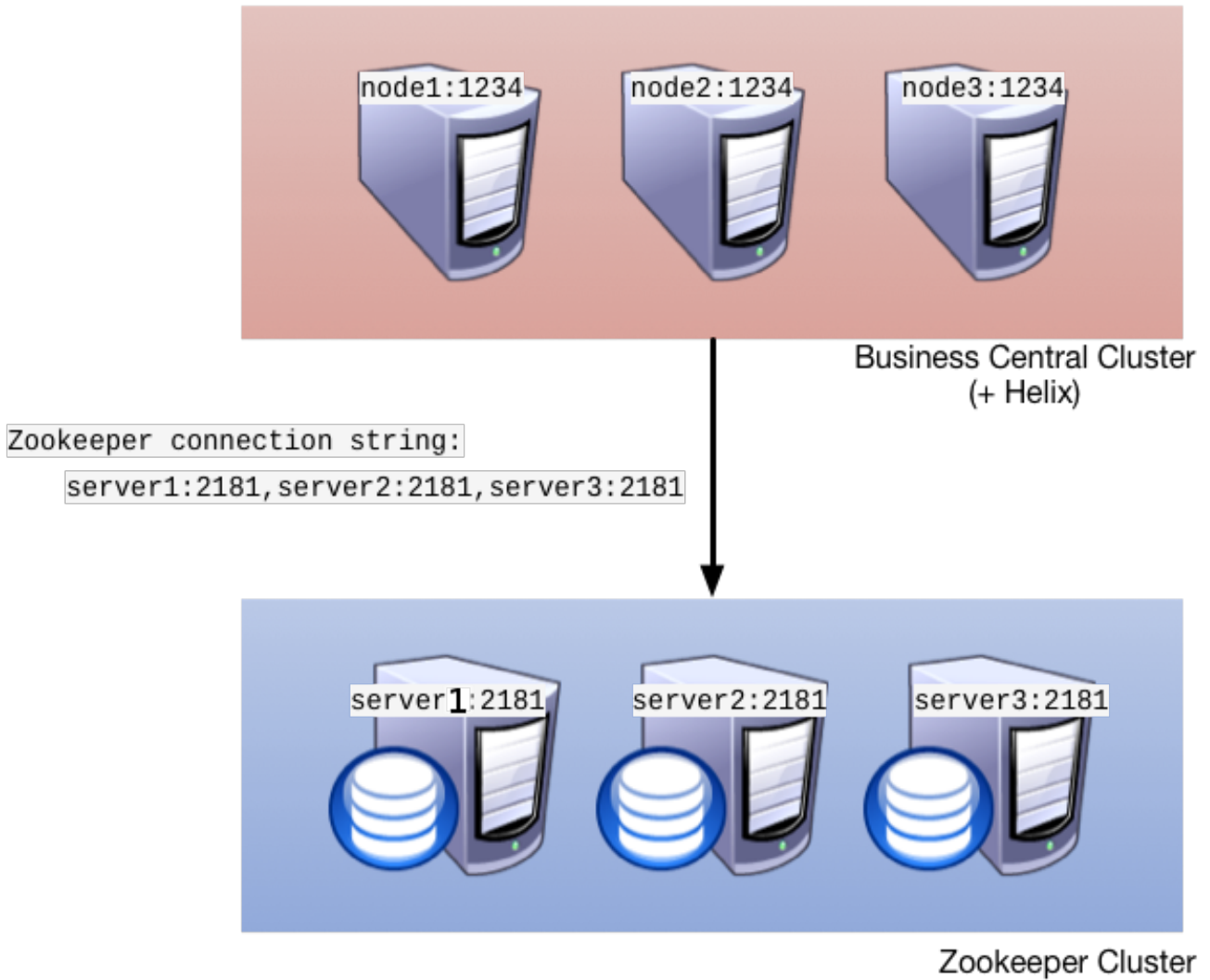
### GIT REPOSITORY CLUSTERING MECHANISM

To cluster the GIT repository the following is used:

- Apache Zookeeper brings all parts together.
- Apache Helix is the cluster management component that registers all cluster details (the cluster itself, nodes, resources).

The runtime environment, that is the Execution Server, utilizes the following to provide the clustering capabilities:

- uberfire framework which provides the backbone of the web applications



**Figure 6.2. Clustering schema with Helix and Zookeeper**

A typical clustering setup involves the following:

- Setting up the cluster itself using Zookeeper and Helix
- Setting up the back-end database with Quartz tables and configuration
- Configuring clustering on your container (this documentation provides only clustering instructions for Red Hat JBoss EAP 6 )

## CLUSTERING MAVEN REPOSITORIES

Various operations within the Business Central publish JARs to the Business Central's internal Maven Repository.

This repository exists on the application server's file-system as regular files and is *not* cluster aware. This folder is not synchronized across the various nodes in the cluster and must be synchronized using external tools like `rsync`.

An alternate to the use of an external synchronization tool is to set the system property `org.guvnor.m2repo.dir` on each cluster node to point to a SAN or NAS. In this case clustering of the Maven repository folder is not needed.

## 6.1. SETTING UP A CLUSTER

To cluster your GIT (VFS) repository in Business Central, do the following (If you don't use Business Central, you may skip this section):

1. Download the `jboss-bpmsuite-brms-VERSION-supplementary-tools.zip`, which contains Apache Zookeeper, Apache Helix, and quartz DDL scripts. After downloading, unzip the archive: the Zookeeper directory (`$ZOOKEEPER_HOME`) and the Helix directory (`$HELIX_HOME`) are created.
2. Now Configure ZooKeeper:

- a. In the ZooKeeper directory, go to `conf` directory and do the following:

```
cp zoo_sample.cfg zoo.cfg
```

- b. Open `zoo.cfg` for editing and adjust the settings including the following:

```
# the directory where the snapshot is stored.
dataDir=$ZOOKEEPER_HOME/data/
# the port at which the clients connects
clientPort=2181
server.1=server1:2888:3888
server.2=server2:2888:3888
server.3=server3:2888:3888
```

Make sure the `dataDir` location exists and is accessible.

- c. Assign a node ID to each member that will run ZooKeeper. For example, use "1", "2" and "3" respectively for node 1, node 2 and node 3 respectively. ZooKeeper should have an odd number of instances, at least 3 in order to recover from failure.

The node ID is specified in a field called `myid` under the data directory of ZooKeeper on each node. For example, on node 1, run: `$ echo "1" > /zookeeper/data/myid`

3. Set up ZooKeeper, so you can use it when creating the cluster with Helix:

- a. Go to the `$ZOOKEEPER_HOME/bin/` directory and start ZooKeeper:

```
./zkServer.sh start
```

You can check the ZooKeeper log in the `$ZOOKEEPER_HOME/bin/zookeeper.out` file. Check this log to ensure that the 'ensemble' (cluster) is formed successfully. One of the nodes should be elected as leader with the other two nodes following it.

4. Once the ZooKeeper ensemble is started, the next step is to configure and start Helix. Helix only needs to be configured once and from a single node. The configuration is then stored by the ZooKeeper ensemble and shared as appropriate.

Set up the cluster with the ZooKeeper server as the master of the configuration:

- a. Create the cluster by providing the ZooKeeper Host and port as a comma separated list:

```
$HELIX_HOME/bin/helix-admin.sh --zkSvr
ZOOKEEPER_HOST:ZOOKEEPER_PORT --addCluster CLUSTER_NAME
```

■

b. Add your nodes to the cluster:

```
$HELIX_HOME/bin/helix-admin.sh --zkSvr
ZOOKEEPER_HOST:ZOOKEEPER_PORT --addNode CLUSTER_NAME
NODE_NAMEUNIQUE_ID
```

**Example 6.1. Adding three cluster nodes**

```
./helix-admin.sh --zkSvr server1:2181,server2:2181,server3:2181
--addNode bpms-cluster nodeOne:12345
./helix-admin.sh --zkSvr server1:2181,server2:2181,server3:2181
--addNode bpms-cluster nodeTwo:12346
./helix-admin.sh --zkSvr server1:2181,server2:2181,server3:2181
--addNode bpms-cluster nodeThree:12347
```

5. Add resources to the cluster.

**Example 6.2. Adding vfs-repo as resource**

```
./helix-admin.sh --zkSvr server1:2181,server2:2181,server3:2181 --
addResource bpms-cluster vfs-repo 1 LeaderStandby AUTO_REBALANCE
```

6. Rebalance the cluster with the three nodes.

**Example 6.3. Rebalancing the bpms-cluster**

```
./helix-admin.sh --zkSvr server1:2181,server2:2181,server3:2181 --
rebalance bpms-cluster vfs-repo 3
```

In the above command, **3** stands for three zookeeper nodes.

7. Start the Helix controller in all the nodes in the cluster.

**Example 6.4. Starting the Helix controller**

```
./run-helix-controller.sh --zkSvr
server1:2181,server2:2181,server3:2181 --cluster bpms-cluster 2>&1
> /tmp/controller.log &
```

**NOTE**

Zookeeper should an odd number of instances, at least 3 in order to recover from failure. After a failure, the remaining number of nodes still need to be able to form a majority. For example a cluster of five Zookeeper nodes can withstand loss of two nodes in order to fully recover. One Zookeeper instance is still possible, replication will work, however no recover possibilities are available if it fails.

## Stopping Helix and Zookeeper

To stop Helix processes and the Zookeeper server, use the following procedure.

### Procedure 6.1. Stopping Helix and Zookeeper

1. Stop JBoss EAP server processes.
2. Stop the Helix process that has been created by `run-helix-controller.sh`, for example, `kill -15 <pid of HelixControllerMain>`.
3. Stop ZooKeeper server using the `zkServer.sh stop` command.

## 6.2. SETTING UP QUARTZ

Before you can configure the database on your application server, you need to prepare the database for Quartz to create Quartz tables, which will hold the timer data, and the Quartz definition file.

Of course, if you are not using Quartz (timers) in your business processes, or if you are not using the Execution Server at all, you can skip this section completely. Please note that if you want to replicate timers in your business process, you need to use the Quartz component.

To set this up, do the following:

1. Set up the database. Make sure to use one of the supported non-JTA data source. Note, that since Quartz need a non-JTA data source, you cannot use the Business Central data source. In the example code, PostgreSQL with the user `bpms` and password `bpms` is used. This database will need to be connected to your application server, so make sure to keep a note on the database information and credentials.
2. Create Quartz tables on your database to allow timer events synchronization. To do so, use the DDL script for your database, which is available in the extracted supplementary zip archive in `QUARTZ_HOME/docs/dbTables`.
3. Create the Quartz configuration file `quartz-definition.properties` in `$JBOSS_HOME/PROFILE/configuration/` directory and define the Quartz properties.

### Example 6.5. Quartz configuration file for a PostgreSQL database

```

#=====
=====
# Configure Main Scheduler Properties
#=====
=====

org.quartz.scheduler.instanceName = jBPMClusteredScheduler
org.quartz.scheduler.instanceId = AUTO

#=====
=====
# Configure ThreadPool
#=====
=====

org.quartz.threadPool.class = org.quartz.simpl.SimpleThreadPool
org.quartz.threadPool.threadCount = 5

```

```

org.quartz.threadPool.threadPriority = 5

#=====
# Configure JobStore
#=====

org.quartz.jobStore.misfireThreshold = 60000

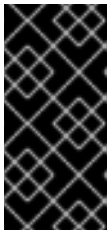
org.quartz.jobStore.class=org.quartz.impl.jdbcjobstore.JobStoreCMT
org.quartz.jobStore.driverDelegateClass=org.quartz.impl.jdbcjobsto
re.PostgreSQLDelegate
org.quartz.jobStore.useProperties=false
org.quartz.jobStore.dataSource=managedDS
org.quartz.jobStore.nonManagedTXDataSource=notManagedDS
org.quartz.jobStore.tablePrefix=QRTZ_
org.quartz.jobStore.isClustered=true
org.quartz.jobStore.clusterCheckinInterval = 20000

#=====
# Configure Datasources
#=====

org.quartz.dataSource.managedDS.jndiURL=jboss/datasources/psbpmsDS
org.quartz.dataSource.notManagedDS.jndiURL=jboss/datasources/quart
zNotManagedDS

```

Note the configured datasources that will accommodate the two Quartz schemes at the very end of the file.



### IMPORTANT

The recommended interval for cluster discovery is 20 seconds and is set in the `org.quartz.jobStore.clusterCheckinInterval` of the `quartz-definition.properties` file. Depending on your set up consider the performance impact and modify the setting as necessary.

Also note the `org.quartz.jobStore.driverDelegateClass` property that defines the DB dialect to be used when communicating with the set database (in this example, `org.quartz.impl.jdbcjobstore.PostgreSQLDelegate`. When using Oracle, use `org.quartz.impl.jdbcjobstore.oracle.OracleDelegate`).

## 6.3. CONFIGURING CLUSTERING ON RED HAT JBOSS EAP

The information provided in this section is a simple clustering recipe. For additional information on clustering refer to *Red Hat JBoss EAP documentation*

When using JBoss EAP clustering, a single JBoss EAP domain controller exists with other JBoss EAP slaves connecting to it as management users. Deployment of Business Central and dashbuilder can be done as a management user on a domain controller, and the WAR deployments will be distributed to other members of the JBoss EAP cluster.



To configure clustering on Red Hat JBoss EAP 6, do the following:

1. Install your JDBC driver as a core module: copy the driver jar to `$EAP_HOME/modules/system/layers/base/` and create a `module.xml` file in the directory.
2. Edit the `module.xml` file as of the respective module XSD.

**Example 6.6. The module.xml file content for a PostgreSQL datasource**

```
<module xmlns="urn:jboss:module:1.0" name="org.postgresql">
  <resources>
    <resource-root path="postgresql-jdbc.jar"/>
  </resources>

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

3. Configure individual server nodes in the `main-server-group` element in the `$EAP_HOME/domain/configuration/host.xml` file with properties defined in [Table 6.1](#), “Cluster node properties”:

Note that a when configuring a JBoss EAP cluster with Zookeeper, a different number of JBoss EAP nodes than Zookeeper nodes is possible (keeping in mind that Zookeeper should to have an odd number of nodes). However, having the same node count for both Zookeeper and JBoss EAP is considered best practice.

**Table 6.1. Cluster node properties**

Property name	Value	Description
<code>jboss.node.name</code>	<code>nodeOne</code>	node name unique within the cluster
<code>org.quartz.properties</code>	<code>/bpms/quartz-definition.properties</code>	absolute path to the quartz configuration file
<code>org.uberfire.cluster.autostart</code>	<code>true</code>	This value delays VFS clustering until the application is fully initialized to avoid conflicts when all cluster members create local clones.
<code>org.uberfire.cluster.id</code>	<code>bpms-cluster</code>	Helix cluster name
<code>org.uberfire.cluster.local.id</code>	<code>nodeOne_12345</code>	unique ID of the Helix cluster node  Note that <code>:</code> is replaced with <code>_</code> .

Property name	Value	Description
<code>org.uberfire.cluster.vfs.lock</code>	<code>vfs-repo</code>	name of the resource defined on the Helix cluster
<code>org.uberfire.cluster.zk</code>	<code>server1:2181</code>	Zookeeper location
<code>org.uberfire.metadata.index.dir</code>	<code>/home/jbpm/node[N]/index</code>	location where the index for search is to be created (maintained by Apache Lucene)
<code>org.uberfire.nio.git.daemon.host</code>	<code>nodeOne</code>	the name of the daemon host machine in a physical cluster.
<code>org.uberfire.nio.git.daemon.port</code>	<code>9418</code>	port used by the VFS repo to accept client connections  The port must be unique for each cluster member.
<code>org.uberfire.nio.git.dir</code>	<code>/home/jbpm/node[N]/repo</code>	GIT (VFS) repository location on node[N]
<code>org.uberfire.nio.git.ssh.host</code>	<code>nodeOne</code>	the name of the SSH host machine in a physical cluster.
<code>org.uberfire.nio.git.ssh.port</code>	<code>8003</code>	the unique port number for ssh access to the GIT repo for a cluster running on physical machines.
<code>org.uberfire.nio.git.ssh.hostport</code> and <code>org.uberfire.nio.git.daemon.hostport</code>	<code>8003</code> and <code>9418</code>	In a virtualized environment, the outside port to be used.

#### Example 6.7. Cluster nodeOne configuration

```
<system-properties>
  <property name="org.uberfire.nio.git.dir"
    value="/tmp/bpms/nodeone" boot-time="false"/>
  <property name="jboss.node.name" value="nodeOne" boot-
    time="false"/>
  <property name="org.uberfire.cluster.id" value="bpms-cluster"
    boot-time="false"/>
  <property name="org.uberfire.cluster.zk"
    value="server1:2181,server2:2181,server3:2181" boot-time="false"/>
  <property name="org.uberfire.cluster.local.id"
```

```

value="nodeOne_12345" boot-time="false"/>
  <property name="org.uberfire.cluster.vfs.lock" value="vfs-repo"
boot-time="false"/>
  <property name="org.uberfire.cluster.autostart" value="true"
boot-time="true"/>

  <property name="org.uberfire.nio.git.daemon.host"
value="nodeOne" />
  <property name="org.uberfire.nio.git.daemon.port" value="9418"
boot-time="false"/>
  <property name="org.uberfire.nio.git.daemon.hostport"
value="9418" boot-time="false"/>

  <property name="org.uberfire.nio.git.ssh.port" value="8003"
boot-time="false"/>
  <property name="org.uberfire.nio.git.ssh.hostport" value="8003"
boot-time="false"/>
  <property name="org.uberfire.nio.git.ssh.host" value="nodeOne"
/>

  <property name="org.uberfire.metadata.index.dir"
value="/tmp/jbpm/nodeone" boot-time="false"/>
  <property name="org.uberfire.nio.git.ssh.cert.dir"
value="/tmp/jbpm/nodeone" boot-time="false"/>

  <property name="org.quartz.properties"
value="/tmp/jbpm/quartz/quartz-db-postgres.properties" boot-
time="false"/>

</system-properties>

```

### Example 6.8. Cluster nodeTwo configuration

```

<system-properties>

  <property name="org.uberfire.nio.git.dir"
value="/tmp/bpms/nodetwo" boot-time="false"/>
  <property name="jboss.node.name" value="nodeTwo" boot-
time="false"/>

  <property name="org.uberfire.cluster.id" value="bpms-cluster"
boot-time="false"/>
  <property name="org.uberfire.cluster.zk"
value="server1:2181,server2:2181,server3:2181" boot-time="false"/>
  <property name="org.uberfire.cluster.local.id"
value="nodeTwo_12346" boot-time="false"/>
  <property name="org.uberfire.cluster.vfs.lock" value="vfs-repo"
boot-time="false"/>
  <property name="org.uberfire.cluster.autostart" value="true"
boot-time="true"/>

  <property name="org.uberfire.nio.git.daemon.host"
value="nodeTwo" />
  <property name="org.uberfire.nio.git.daemon.port" value="9418"

```

```
boot-time="false"/>
  <property name="org.uberfire.nio.git.daemon.hostport"
value="9418" boot-time="false"/>

  <property name="org.uberfire.nio.git.ssh.port" value="8003"
boot-time="false"/>
  <property name="org.uberfire.nio.git.ssh.hostport" value="8003"
boot-time="false"/>
  <property name="org.uberfire.nio.git.ssh.host" value="nodeTwo"
/>

  <property name="org.uberfire.metadata.index.dir"
value="/tmp/jbpm/nodetwo" boot-time="false"/>
  <property name="org.uberfire.nio.git.ssh.cert.dir"
value="/tmp/jbpm/nodetwo" boot-time="false"/>

  <property name="org.quartz.properties"
value="/tmp/jbpm/quartz/quartz-db-postgres.properties" boot-
time="false"/>

</system-properties>
```

#### Example 6.9. Cluster nodeThree configuration

```
<system-properties>

  <property name="org.uberfire.nio.git.dir"
value="/tmp/bpms/nodethree" boot-time="false"/>
  <property name="jboss.node.name" value="nodeThree" boot-
time="false"/>

  <property name="org.uberfire.cluster.id" value="bpms-cluster"
boot-time="false"/>
  <property name="org.uberfire.cluster.zk"
value="server1:2181,server2:2181,server3:2181" boot-time="false"/>
  <property name="org.uberfire.cluster.local.id"
value="nodeThree_12347" boot-time="false"/>
  <property name="org.uberfire.cluster.vfs.lock" value="vfs-repo"
boot-time="false"/>
  <property name="org.uberfire.cluster.autostart" value="true"
boot-time="true"/>

  <property name="org.uberfire.nio.git.daemon.host"
value="nodeThree" />
  <property name="org.uberfire.nio.git.daemon.port" value="9418"
boot-time="false"/>
  <property name="org.uberfire.nio.git.daemon.hostport"
value="9418" boot-time="false"/>

  <property name="org.uberfire.nio.git.ssh.port" value="8003"
boot-time="false"/>
  <property name="org.uberfire.nio.git.ssh.hostport" value="8003"
boot-time="false"/>
```

```

    <property name="org.uberfire.nio.git.ssh.host" value="nodeThree"
    />

    <property name="org.uberfire.metadata.index.dir"
    value="/tmp/jbpm/nodethree" boot-time="false"/>
    <property name="org.uberfire.nio.git.ssh.cert.dir"
    value="/tmp/jbpm/nodethree" boot-time="false"/>

    <property name="org.quartz.properties"
    value="/tmp/jbpm/quartz/quartz-db-postgres.properties" boot-
    time="false"/>

</system-properties>

```

4. Add management users as instructed in the *Administration and Configuration Guide* for Red Hat JBoss EAP and application users as instructed in *Red Hat JBoss BPM Suite Administration and Configuration Guide*.

5. Start the application server:

```
]$ $JBOSS_HOME/bin/domain.sh
```

6. Check that the nodes are available.

Deploy the Business Central application to your servers:

1. Change the predefined persistence of the application to the required data base (PostgreSQL): in `persistence.xml` change the following:
  - a. `jta-data-source` name to the source defined on the application server (`java:joboss/datasources/psbpmsDS`)
  - b. `hibernate dialect` to be match the data source dialect (`org.hibernate.dialect.PostgreSQLDialect`)
2. Log on as the management user to the server Administration console of your domain and add the new deployments using the Runtime view of the console. Once the deployment is added to the domain, assign it to the correct server group (`main-server-group`).

## NOTE

It is important users explicitly check deployment unit readiness with every cluster member.

When a deployment unit is created on a cluster node, it takes some time before it is distributed among all cluster members. Deployment status can be checked via UI and REST, however if the query goes to the node where the deployment was originally issued, the answer is **deployed**. Any request targeting this deployment unit sent to a different cluster member fails with **DeploymentNotFoundException**.

## CHAPTER 7. MAVEN REPOSITORIES

### 7.1. ABOUT MAVEN

Apache Maven is a distributed build automation tool used in Java application development to build and manage software projects. Maven uses configuration XML files called POM (Project Object Model) to define project properties and manage the build process. POM files describe the project's module and component dependencies, build order, and targets for the resulting project packaging and output. This ensures that projects are built in a correct and uniform manner.

Maven uses repositories to store Java libraries, plug-ins, and other build artifacts. Repositories can be either local or remote. A local repository is a download of artifacts from a remote repository cached on a local machine. A remote repository is any other repository accessed using common protocols, such as `http://` when located on an HTTP server, or `file://` when located on a file server. The default repository is the public remote [Maven 2 Central Repository](#).

Configuration of Maven is performed by modifying the `settings.xml` file. You can either configure global Maven settings in the `M2_HOME/conf/settings.xml` file, or user-level settings in the `USER_HOME/.m2/settings.xml` file.

For more information about Maven, see [Welcome to Apache Maven](#).

For more information about Maven repositories, see [Apache Maven Project - Introduction to Repositories](#).

For more information about Maven POM files, see the [Apache Maven Project POM Reference](#).



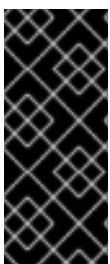
#### NOTE

Your Red Hat JBoss product has been built with maven 3.0.x Therefore, this is the recommended maven version for building your own SwitchYard applications.

### 7.2. ABOUT THE PROVIDED MAVEN REPOSITORIES

A set of repositories containing artifacts required to build applications based on Red Hat JBoss BPM Suite is provided with this release. Maven must be configured to use these repositories and the Maven Central Repository in order to provide correct build functionality.

Two interchangeable sets of repositories ensuring the same functionality are provided. The first set is available for download and storage in a local file system, the second set is hosted online for use as remote repositories. If you provided the location of Maven's `settings.xml` file during installation, Maven is already configured to use the online repositories.



#### IMPORTANT

The set of online remote repositories is a technology preview source of components. As such, it is not in scope of patching and is supported only for use in development environment. Using the set of online repositories in production environment is a potential source of security vulnerabilities and is therefore not a supported use case. For more information see <https://access.redhat.com/site/maven-repository>.

### 7.3. CONFIGURING MAVEN TO USE THE FILE SYSTEM REPOSITORIES

## Overview

In situations where you cannot use the online repositories, you will have to download and configure the required repositories locally.

### Procedure 7.1.

1. Download the following ZIP archives containing the required repositories:
  - o <https://access.redhat.com/jbossnetwork/restricted/listSoftware.html?product=bpm.suite&version=6.1.0>
2. Unzip the downloaded ZIP files into an arbitrary location in a local file system.
3. Add entries for the unzipped repositories to Maven's `settings.xml` file. The following code sample contains a profile with the repositories, configuration of authentication for access to the repositories, and an activation entry for the profile:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<settings xmlns="http://maven.apache.org/SETTINGS/1.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/xsd/settings-1.0.0.xsd">
  <localRepository/>
  <profiles>
    <!-- Profile with local repositories required by JBoss
BRMS/JBoss BPM Suite -->
    <profile>
      <id>brms-bpms-local-profile</id>
      <repositories>
        <repository>
          <id>jboss-brms-bpmsuite-repository</id>
          <name>BRMS/BPMS 6.1.0 GA Repository</name>
          <url>file://<!-- path to the repository -->/jboss-brms-
bpmsuite-6.1.0.GA-redhat-2-maven-repository/maven-repository</url>
          <layout>default</layout>
          <releases>
            <enabled>>true</enabled>
            <updatePolicy>never</updatePolicy>
          </releases>
          <snapshots>
            <enabled>>false</enabled>
            <updatePolicy>never</updatePolicy>
          </snapshots>
        </repository>
      </repositories>
      <pluginRepositories>
        <pluginRepository>
          <id>jboss-brms-bpmsuite-repository</id>
          <name>BRMS/BPMS 6.1.0 GA Repository</name>
          <url>file://<!-- path to the repository -->/jboss-brms-
bpmsuite-6.1.0.GA-redhat-2-maven-repository/maven-repository</url>
          <layout>default</layout>
          <releases>
            <enabled>>true</enabled>
            <updatePolicy>never</updatePolicy>
          </releases>
        </pluginRepository>
      </pluginRepositories>
    </profile>
  </profiles>
</settings>
```

```

        <snapshots>
            <enabled>>false</enabled>
            <updatePolicy>never</updatePolicy>
        </snapshots>
    </pluginRepository>
</pluginRepositories>
</profile>
</profiles>

    <!-- Configuring pre-emptive authentication for the repository
server -->
    <server>
        <id>brms-bpms-m2-repo</id>
        <username>admin</username>
        <password>admin</password>
        <configuration>
            <wagonProvider>httpClient</wagonProvider>
            <httpConfiguration>
                <all>
                    <usePreemptive>>true</usePreemptive>
                </all>
            </httpConfiguration>
        </configuration>
    </server>

    <!-- Alternative to enabling pre-emptive authentication -
configuring
        the Authorization HTTP header with Base64-encoded credentials
    <server>
        <id>brms-bpms-m2-repo</id>
        <configuration>
            <httpHeaders>
                <property>
                    <name>Authorization</name>
                    <value>Basic YWRtaW46YWRtaW4=</value>
                </property>
            </httpHeaders>
        </configuration>
    </server>
    -->

    <activeProfiles>
        <!-- Activation of the JBoss BRMS/JBoss BPM Suite profile -->
        <activeProfile>brms-bpms-local-profile</activeProfile>
    </activeProfiles>
</settings>

```

## Result

The Maven repositories are downloaded, unzipped in a local file system, registered in Maven's `settings.xml` file, and ready to be used when performing Maven builds.

## Troubleshooting

**Q:** Why do I still get errors when building or deploying my applications?

**A:** Issue



When you build or deploy a project, it fails with one or both of the following errors:

```
[ERROR] Failed to execute goal on project PROJECT_NAME
```

```
Could not find artifact ARTIFACT_NAME
```

#### Cause

Your cached local Maven repository might contain outdated artifacts.

#### Resolution

To resolve the issue, delete the cached local repository – the `~/ .m2/repository/` directory on Linux or the `%SystemDrive%\Users\USERNAME\.m2\repository\` directory on Windows – and run `mvn clean install -U`. This will force Maven to download correct versions of required artifacts when performing the next build.

---

**Q:** Why is JBoss Developer Studio using my old Maven configuration?

**A:** Issue

You have updated your Maven configuration, but this configuration is not reflected in JBoss Developer Studio.

#### Cause

If JBoss Developer Studio is running when you modify your Maven `settings.xml` file, this configuration will not be reflected in JBoss Developer Studio.

#### Resolution

Refresh the Maven settings in the IDE. From the menu, choose **Window** → **Preferences**. In the **Preferences Window**, expand **Maven** and choose **User Settings**. Click the **Update Settings** button to refresh the Maven user settings in JBoss Developer Studio.

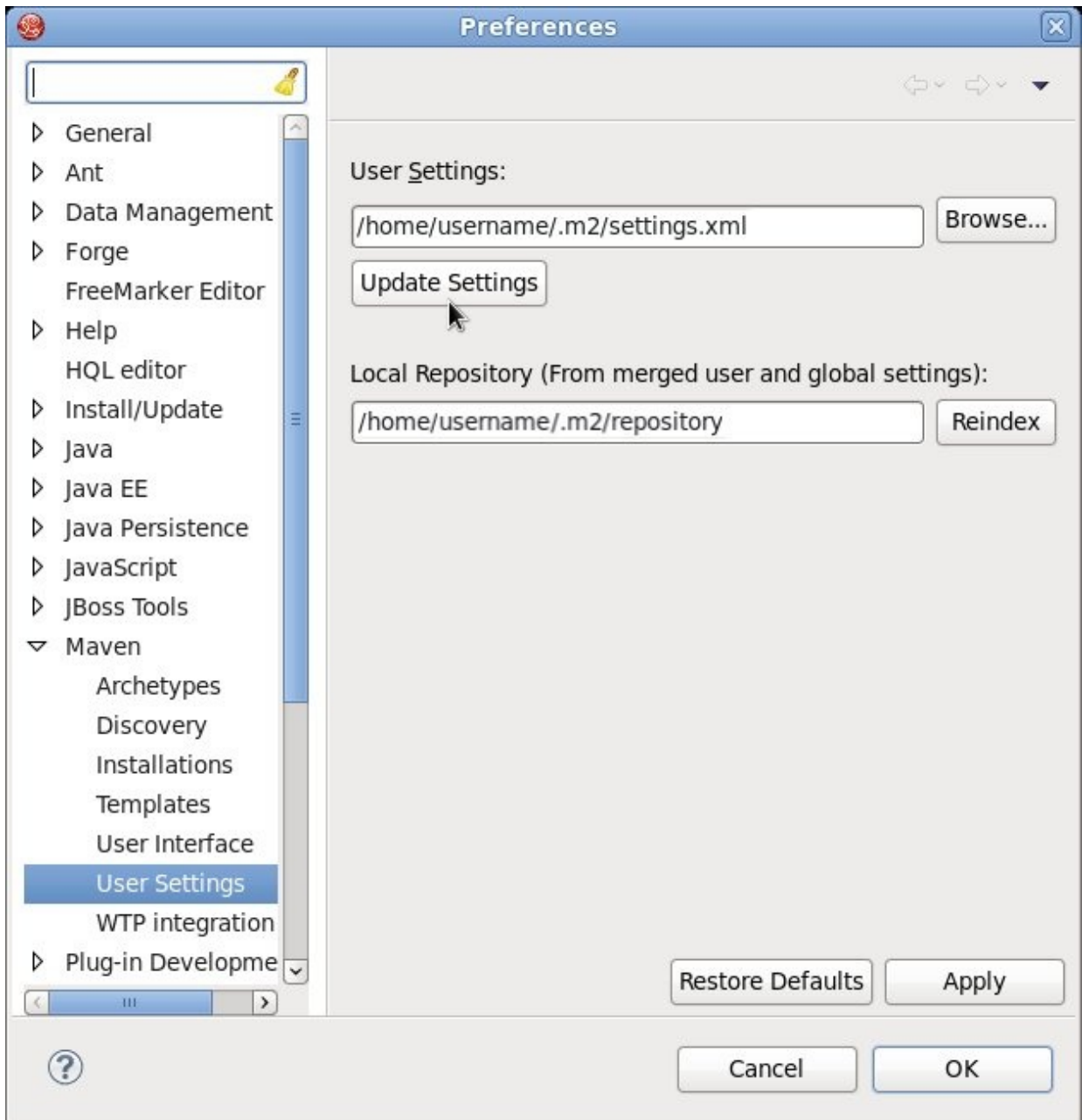


Figure 7.1. Update Maven User Settings

## 7.4. CONFIGURING MAVEN TO USE THE ONLINE REPOSITORIES

The online repositories required for Red Hat JBoss BPM Suite applications are located at <http://maven.repository.redhat.com/techpreview/all/>.

If you did not configure the Maven repository during installation, you can configure it using the following procedure. (It is also possible to do this using the project's POM file, but this is not recommended.)

### Procedure 7.2. Configuring Maven to Use the Online Repositories

1. Add entries for the online repositories and configuration of authentication for accessing them to Maven's `settings.xml` file as in the code sample below:

-

```

<settings xmlns="http://maven.apache.org/SETTINGS/1.0.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/SETTINGS/1.0.0
http://maven.apache.org/xsd/settings-1.0.0.xsd">

  <profiles>
    <!-- Profile with online repositories required by BRMS/BPMS -->
    <profile>
      <id>brms-bpms-online-profile</id>
      <repositories>
        <repository>
          <id>jboss-ga-repository</id>

          <url>http://maven.repository.redhat.com/techpreview/all</url>
          <releases>
            <enabled>>true</enabled>
          </releases>
          <snapshots>
            <enabled>>false</enabled>
          </snapshots>
        </repository>
      </repositories>
      <pluginRepositories>
        <pluginRepository>
          <id>jboss-ga-plugin-repository</id>

          <url>http://maven.repository.redhat.com/techpreview/all</url>
          <releases>
            <enabled>>true</enabled>
          </releases>
          <snapshots>
            <enabled>>false</enabled>
          </snapshots>
        </pluginRepository>
      </pluginRepositories>
    </profile>
  </profiles>

  <!-- Configuring pre-emptive authentication for the repository
server -->
  <server>
    <id>brms-bpms-m2-repo</id>
    <username>admin</username>
    <password>admin</password>
    <configuration>
      <wagonProvider>httpClient</wagonProvider>
      <httpConfiguration>
        <all>
          <usePreemptive>>true</usePreemptive>
        </all>
      </httpConfiguration>
    </configuration>
  </server>

  <!-- Alternative to enabling pre-emptive authentication -
configuring

```

```
    the Authorization HTTP header with Base64-encoded credentials
<server>
  <id>brms-bpms-m2-repo</id>
  <configuration>
    <httpHeaders>
      <property>
        <name>Authorization</name>
        <value>Basic YWRtaW46YWRtaW4=</value>
      </property>
    </httpHeaders>
  </configuration>
</server>
-->

<activeProfiles>
  <!-- Activation of the BRMS/BPMS profile -->
  <activeProfile>brms-bpms-online-profile</activeProfile>
</activeProfiles>

</settings>
```

2. If you modified the `settings.xml` file while JBoss Developer Studio was running, you must refresh Maven settings in the IDE. From the menu, choose **Window** → **Preferences**. In the **Preferences** Window, expand **Maven** and choose **User Settings**. Click the **Update Settings** button to refresh the Maven user settings in JBoss Developer Studio.

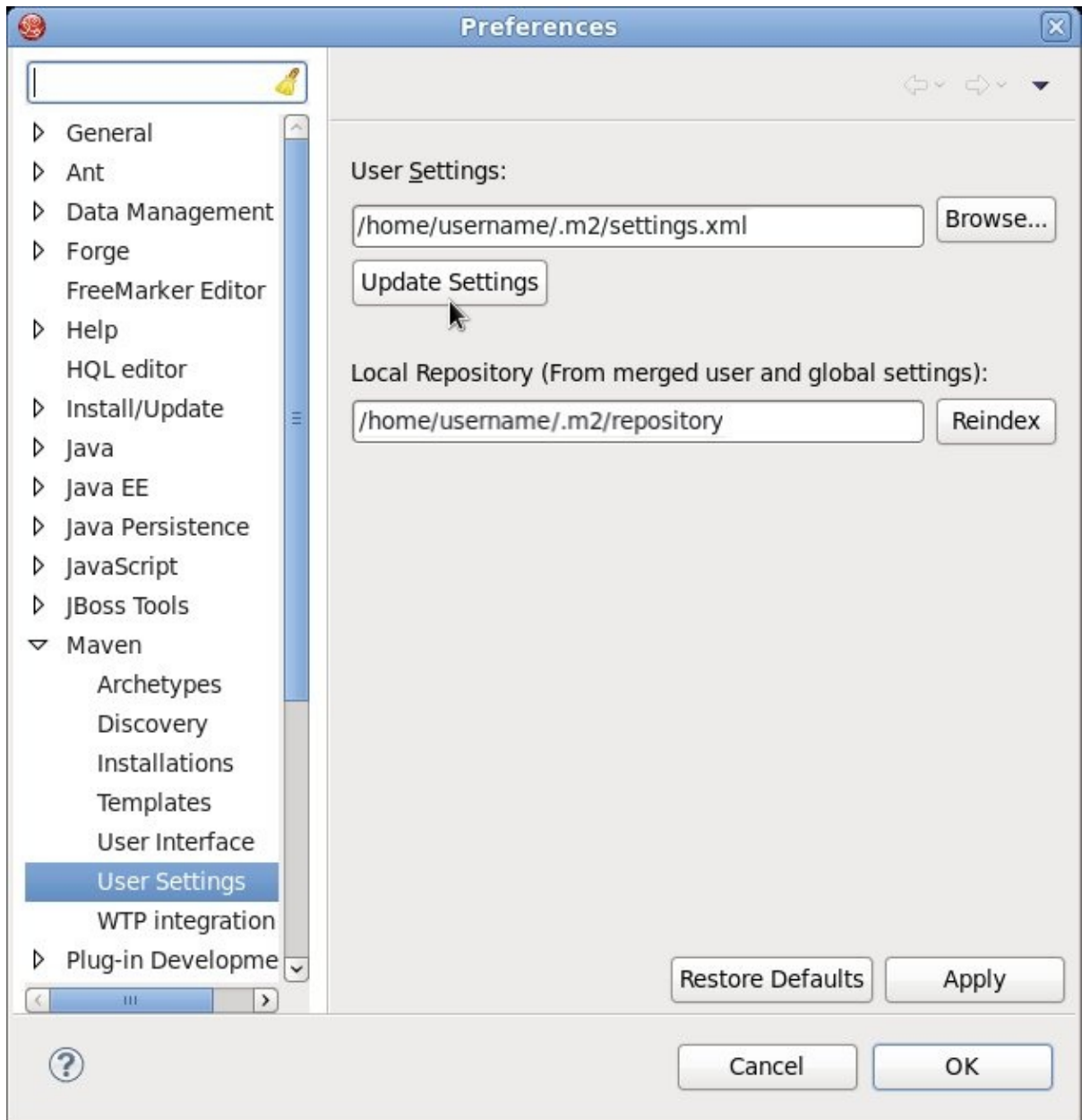


Figure 7.2. Update Maven User Settings

## Result

Maven has been configured to use the online repositories provided for your Red Hat JBoss product.

## IMPORTANT

If your cached local Maven repository contains outdated artifacts, you may encounter one of the following Maven errors when you build or deploy your project:

- Missing artifact *ARTIFACT\_NAME*
- [ERROR] Failed to execute goal on project *PROJECT\_NAME*; Could not resolve dependencies for *PROJECT\_NAME*

To resolve the issue, delete the cached local repository - the `~/ .m2/repository/` directory on Linux or the `%SystemDrive%\Users\USERNAME\.m2\repository\` directory on Windows. This will force Maven to download correct versions of required artifacts during the next build.

## 7.5. DEPENDENCY MANAGEMENT

In order to use the correct Maven dependencies in your Red Hat JBoss BPM Suite project, you must add relevant Bill Of Materials (BOM) files to the project's `pom.xml` file. Adding the BOM files ensures that the correct versions of transitive dependencies from the provided Maven repositories are included in the project.

The Maven repository in 6.1.0 is designed to be used only in combination with Maven Central and no other repositories are required.

Depending on your project requirements, declare the dependencies in your POM file in the dependencies section:

- `org.jboss.bom.brms:jboss-brms-bpmsuite-bom:VERSION`: This is the basic BOM without any Java EE6 support.
- `org.jboss.bom.brms:jboss-javaee-6.0-with-brms-bpmsuite:VERSION`: This provides support for Java EE6.

## CHAPTER 8. RED HAT JBOSS DEVELOPER STUDIO

### 8.1. RED HAT JBOSS DEVELOPER STUDIO

Red Hat JBoss Developer Studio is the JBoss Integrated Development Environment (IDE) based on Eclipse. Get the latest JBoss Developer Studio from the Red Hat customer support portal at <https://access.redhat.com>. JBoss Developer Studio provides plug-ins with tools and interfaces for Red Hat JBoss BRMS and Red Hat JBoss BPM Suite. These plugins are based on the community version of these products. So, the JBoss BRMS plug-in is called the Drools plug-in and the JBoss BPM Suite plug-in is called the jBPM plug-in.

Refer to the *Red Hat JBoss Developer Studio* documentation for installation and set-up instructions.



#### WARNING

Due to an issue in the way multi-byte rule names are handled, you must ensure that the instance of JBoss Developer Studio is started with the file encoding set to UTF-8. You can do this by editing the `$JBDS_HOME/studio/jbdevstudio.ini` file and adding the following property: `"-Dfile.encoding=UTF-8"`

### 8.2. INSTALLING THE JBOSS DEVELOPER STUDIO PLUG-INS

The Drools and jBPM plug-ins for JBoss Developer Studio are available via the update site.

#### Procedure 8.1. Install the Drools and jBPM JBoss Developer Studio Plug-in

1. Start JBoss Developer Studio.
2. Select **Help** → **Install New Software**.
3. Click **Add** to enter the **Add Repository** menu.
4. Give the software site a name next to **Name** field and add the following url in the **Location** field: <https://devstudio.jboss.com/updates/8.0/integration-stack/>
5. Click **OK**.
6. Select the **JBoss Business Process and Rule Development** feature from the available options and click **Next** and then **Next** again.
7. Read the license and accept it by selecting the appropriate radio button, and click **Finish**.
8. After installation of the plug-ins has completed, restart JBoss Developer Studio.

### 8.3. SETTING THE DROOLS RUNTIME

In order to use the Red Hat JBoss BRMS plug-in with Red Hat JBoss Developer Studio, it is necessary to set up the runtime.

A runtime is a collection of jar files that represent a specific release of the software and provides libraries needed for compilation and running of your business assets.

### Procedure 8.2. Configure JBoss BRMS Runtime

1. Extract the runtime jar files located in the `jboss-brms-VERSION-engine.zip` archive that you can download from [Red Hat Customer Portal](#).
2. From the JBoss Developer Studio menu, select **Window** and click **Preferences**.
3. Select **Drools** → **Installed Drools Runtimes**.
4. Click **Add . . .**; provide a name for the new runtime, and click **Browse** to navigate to the directory where you extracted the runtime files in step 1. Click **OK** to register the selected runtime in JBDS.
5. Mark the runtime you have created as the default Drools runtime by clicking on the check box next to it.
6. Click **OK**. If you have existing projects, a dialog box will indicate that you have to restart JBoss Developer Studio to update the Runtime.

## 8.4. CONFIGURING THE JBOSS BPM SUITE SERVER

JBoss Developer Studio can be configured to run the Red Hat JBoss BPM Suite Server.

### Procedure 8.3. Configure the Server

1. Open the jBPM view by selecting **Window** → **Open Perspective** → **Other** and select **jBPM** and click **OK**.
2. Add the server view by selecting **Window** → **Show View** → **Other...** and select **Server** → **Servers**.
3. Open the server menu by right clicking the Servers panel and select **New** → **Server**.
4. Define the server by selecting **JBoss Enterprise Middleware** → **JBoss Enterprise Application Platform 6.4+** and clicking **Next**.
5. Set the home directory by clicking the **Browse** button. Navigate to and select the installation directory for JBoss EAP 6.4 which has JBoss BPM Suite installed.
6. Provide a name for the server in the **Name** field, make sure that the configuration file is set, and click **Finish**.

## 8.5. IMPORTING PROJECTS FROM A GIT REPOSITORY INTO JBOSS DEVELOPER STUDIO

You can configure JBoss Developer Studio to connect to a central Git asset repository. The repository stores rules, models, functions and processes.

You can either clone a remote Git repository or import a local Git repository.

### Procedure 8.4. Cloning a Remote Git Repository




1. Start the Red Hat JBoss BRMS/BPM Suite server (whichever is applicable) by selecting the server from the server tab and click the start icon.
2. Simultaneously, start the Secure Shell server, if not running already, by using the following command. The command is Linux and Mac specific only. On these platforms, if sshd has already been started, this command fails. In that case, you may safely ignore this step.

```
/sbin/service sshd start
```

3. In JBoss Developer Studio, select **File** → **Import...** and navigate to the Git folder. Open the Git folder to select **Projects from Git** and click **Next**.
4. Select the repository source as **Clone URI** and click **Next**.
5. Enter the details of the Git repository in the next window and click **Next**.

**Clone Git Repository**

**Source Git Repository** 

Enter the location of the source repository.

Location

URI:  Local File...

Host:

Repository path:

Connection

Protocol: ssh ▼


Port:

Authentication

User:

Password:

Store in Secure Store

? 
< Back
Next >
Cancel
Finish

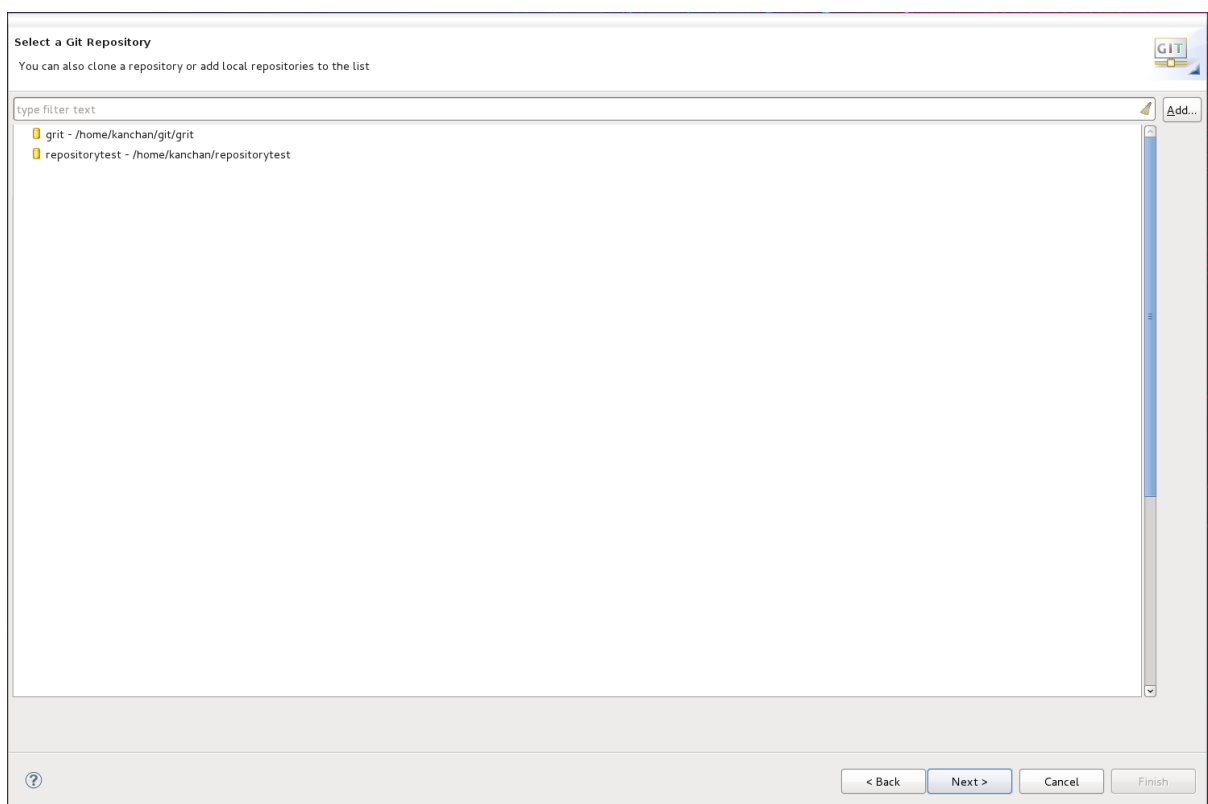
**Figure 8.1. Git Repository Details**

6. Select the branch you wish to import in the following window and click **Next**.

7. To define the local storage for this project, enter (or select) a non-empty directory, make any configuration changes and click **Next**.
8. Import the project as a general project in the following window and click **Next**. Name the project and click **Finish**.

### Procedure 8.5. Importing a Local Git Repository

1. Start the Red Hat JBoss BRMS/BPM Suite server (whichever is applicable) by selecting the server from the server tab and click the start icon.
2. In JBoss Developer Studio, select **File** → **Import...** and navigate to the Git folder. Open the Git folder to select **Projects from Git** and click **Next**.
3. Select the repository source as **Existing local repository** and click **Next**.



**Figure 8.2. Git Repository Details**

4. Select the repository that is to be configured from the list of available repositories and click **Next**.
5. In the dialog that opens, select the radio button **Import as general project** from the **Wizard for project import group** and click **Next**. Name the project and click **Finish**.

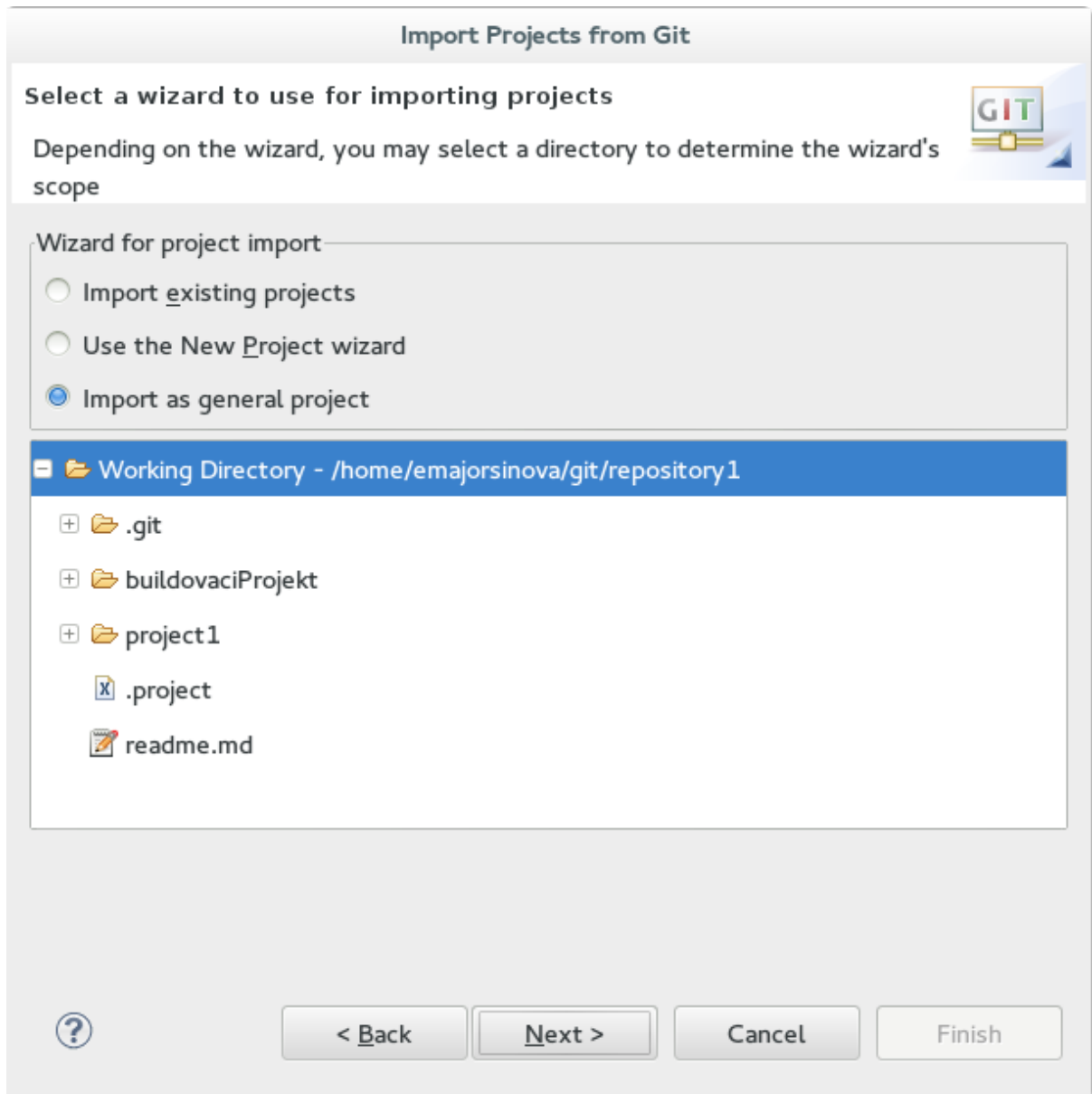


Figure 8.3. Wizard for Project Import

## CHAPTER 9. PATCHING AND UPGRADING RED HAT JBOSS BPM SUITE

### 9.1. ABOUT PATCHES AND UPGRADES

Red Hat JBoss BPM Suite patches can be either an asynchronous update, or a planned update:

- **Asynchronous updates:** Individual patches which are released outside the normal update cycle of the existing product. These may include security patches, as well as other individual patches provided by Red Hat Global Support Services (GSS) to fix specific issues.
- **Planned updates:** The cumulative patches of an existing product, which includes all previously developed updates for that version of the product.

Red Hat JBoss BPM Suite patches can be downloaded from <https://access.redhat.com/downloads/>.

The following files are included as part of a JBoss BRMS and JBoss BPM Suite patch release.

- JBoss BRMS customers - `jboss-brms-<version>-patch.zip`
- JBoss BPM Suite customers - `jboss-bpmsuite-<version>-patch.zip`
- Maven repo updates (Same for both JBoss BRMS and JBoss BPM Suite customers) - `jboss-brms-bpmsuite-<version>-incremental-maven-repository.zip`

### 9.2. APPLYING PATCHES IN RED HAT JBOSS BPM SUITE 6.1

In Red Hat JBoss BPM Suite, the client patching tool is distributed as a zip file that includes simple `.sh` and `.bat` scripts, allowing for easy and automatic application of updates to an existing JBoss BPM Suite 6.1 (or better) installation.



#### IMPORTANT

The patching tool is for use with JBoss BPM Suite 6.1 or better, and should not be used for earlier versions. For more information, see <https://access.redhat.com/articles/1455733>.

The script requires two mandatory parameters: `<path-to-distribution-root>` and `<type-of-distribution>`. For example, the following command applies the updates to the specified JBoss EAP bundle:



#### NOTE

Patch updates should not be applied while you are running an instance of Red Hat JBoss BPM Suite. Make sure that the server is shut down before running the following command.

```
$ ./apply-updates.sh ~/EAP_HOME/jboss-eap-6.4 eap6.x
```

The following distribution types are supported:

- eap6.x
- eap6.x-bc
- eap6.x-kie-server
- eap6.x-dashbuilder
- generic
- generic-bc
- generic-kie-server
- generic-dashbuilder
- was8
- was8-bc
- was8-kie-server
- was8-dashbuilder
- wls12c
- wls12c-bc
- wls12c-kie-server
- wls12c-dashbuilder
- brms-engine
- planner-engine
- supplementary-tools

The quickstarts and migration tool are also included in the patch and are available for download as a zip file.



#### NOTE

Only updates for BRMS/BPM Suite are included in the patch distribution. Patches to EAP itself must be applied using the EAP patching mechanism. See the [Red Hat JBoss EAP Installation Guide](#).

### Backup Feature

Before applying any updates, the client script takes a backup of the specified distribution. It copies the distribution file or directory into the `backup/<current - timestamp>` subdirectory. The top-level backup directory is created at the same filesystem level as the `apply-updates` script.

### Blacklist Feature

The client patching tool provides a blacklist feature that allows you to tell the script the files that must not be updated. This is a really useful feature that helps you to preserve your configuration files from being overwritten automatically by the update process. Of course, you can specify non-configuration

files as well, if required.

To specify the *blacklisted* files, open the file `blacklist.txt` present within the patch distribution. Enter the relative path to the files that must not be updated. Each file must be specified on a line by itself.

```
# lines with a '#' are comment lines, like this one
# blank lines are ignored

# we have made changes to the web.xml that must be preserved
WEB-INF/web.xml

# this file has custom modifications
styles/base.css
```

Files specified in the `blacklist.txt` file that have updated content in the patch, are not touched by the update tool. Instead, the tool copies the new, updated file in the same location and appends the `new` suffix to it. For example, after running the patch tool, both these files will exist in the `styles` folder, continuing with the `blacklist.txt` file in the example above.

```
$ ls styles
base.css base.css.new
```

It is now up to you to compare the contents of the two files and merge the changes.

What happens if there are files that are no longer being distributed but you want to preserve them? Put them into the `blacklist.txt` file as well. The patch update tool will not delete these files, and instead create an empty marker file with the suffix `removed`. It is then up to you to either keep or delete these files manually.

Continuing with the previous example, if the `base.css` file was removed and you had this file listed in the `blacklist.txt` file, then after the patch tool has run, the contents of the `styles` directory would be similar to:

```
$ ls styles
base.css base.css.removed
```

## 9.3. PATCHING OTHER PLATFORMS AND APPLICATIONS

The following commands can be used for updating other supported platforms and common applications in Red Hat JBoss BPM Suite.

### Patch EAP 6.x Business Central WAR

```
$ ./apply-updates.[sh|bat] <some-path>/jboss-eap-6.4/standalone/deployments/business-central.war eap6.x-bc
```

### Patch Generic KIE Server WAR

```
$ ./apply-updates.[sh|bat] <some-path-to-tomcat-home>/webapps/kie-server.war generic-kie-server
```

### Patch Whole WebLogic 12c Bundle

```
$ ./apply-updates.[sh|bat] <path-to-unzipped-wls12c-bundle> wls12c
```

**Patch Planner Engine Bundle**

```
$ ./apply-updates.[sh|bat] <path-to-unzipped-planner-bundle> planner-engine
```

## APPENDIX A. REVISION HISTORY

**Revision 1.0.0-44**

**Thu Dec 17 2015**

**Vidya Iyengar**

Build includes various enhancements and fixes