# Sun OpenSSO Enterprise Policy Agent 3.0 Guide for JBoss Application Server 4.x/5.x

Part No: 820–7585
November 22, 2010

101122@25097

# Sun OpenSSO Enterprise Policy Agent 3.0 Guide for JBoss Application Server 4.x/5.x

Last updated November 22, 2010

The JBoss Application Server 4.x/5.x policy agent is a Java EE agent (formerly called a J2EE agent) that functions with Oracle OpenSSO to protect resources on JBoss Application Server.

**Contents**

For general information about version 3.0 Java EE agents, including the new version 3.0 features, see the *Sun OpenSSO Enterprise Policy Agent 3.0 User's Guide for J2EE Agents*.

**Note –** Oracle also provides a version 2.2 policy agent for JBoss Application Server. However, to use the new version 3.0 policy agent features, you must deploy the JBoss Application Server 4.x/5.x agent described in this guide.

# Supported Platforms and Web Containers for the JBoss Application Server 4.x/5.x Agent

**Note –** If you plan to use web services security (WSS) and JAX-WS with the JBoss Application Server 4.x/5.x agent, you will need to download and install specific JAX-WS JAR files into the JBoss Application Server web container. See "Configuring Web Services Security for the JBoss Application Server 4.x/5.x Agent" on page 25.

## Supported Versions of JBoss Application Server

The JBoss Application Server 4.x/5.x agent is supported on these releases:

- JBoss Application Server 5.x
- JBoss Application Server 4.x

For information about JBoss Application Server, see: `http://www.jboss.org/`

## Supported Platforms for the JBoss Application Server 4.x/5.x Agent

**TABLE 1** Supported Platforms for the JBoss Application Server 4.x/5.x Agent

| Agent For | Supported Platforms |
|---|---|
| JBoss Application Server 5.x<br>JBoss Application Server 4.x | ■ Solaris OS on SPARC and x86 platforms, versions 9 and 10 (32-bit and 64-bit platforms)<br>■ Red Hat Enterprise Linux Advanced Server 4.0 and 5.0 (32-bit and 64-bit platforms)<br>■ Windows 2003 and 2008, Standard Edition and Enterprise Edition (32-bit and 64-bit platforms) |
| ■ Minor versions of the JBoss Application Server web container are supported.<br>■ Minor versions of the supported platforms, including updates, service packs, and patches, are also supported. | |

# Compatibility and Coexistence for the JBoss Application Server 4.x/5.x Agent

## Compatibility With Access Manager 7.1 and Access Manager 7 2005Q4

Sun Java System Access Manager 7.1 and Sun Java System Access Manager 7 2005Q4 are compatible with version 3.0 policy agents. However, because Access Manager does not support centralized agent configuration, a version 3.0 agent deployed with Access Manager must store its configuration data locally in the `OpenSSOAgentConfiguration.properties` and `OpenSSOAgentBootstrap.properties` files.

For both configurations, the `OpenSSOAgentBootstrap.properties` file on the server where the agent is deployed contains the information required for the agent to start and initialize itself.

## Coexistence With Version 2.2 Policy Agents

OpenSSO Enterprise supports both version 3.0 and version 2.2 agents in the same deployment. The version 2.2 agents, however, must continue to store their configuration data locally in their respective `AMAgent.properties` file. Because the version 2.2 agent configuration data is local to the agent, OpenSSO Enterprise centralized agent configuration is not supported for version 2.2 agents. To configure a version 2.2 agent, you must continue to edit the agent's `AMAgent.properties` file.

The OpenSSO Enterprise Console allows you to create and configure a version 2.2 agent profile under Access Control, *realm-name*, Agents, 2.2 Agents.

For information about version 2.2 agents, see the following documentation collection:

`http://docs.sun.com/coll/1322.1`

# Pre-Installation Tasks for the JBoss Application Server 4.x/5.x Agent

## Setting Your JAVA_HOME Environment Variable

Version 3.0 policy agents, including the agentadmin program, require JDK 1.5 or later on the server where you plan to install the agent. Before you install the JBoss Application Server 4.x/5.x agent, set your JAVA_HOME environment variable to point to the JDK installation directory.

## Downloading and Unzipping the jboss_v42_agent_3.zip Distribution File

### ▼ To Download and Unzip the jboss_v42_agent_3.zip Distribution File

1 **Login to the server where you want to install the agent.**

2 **Create a directory to unzip the jboss_v42_agent_3.zip distribution file.**

This guide uses *Agent-Home* to represent the directory where you unzip the distribution file.

3 **Download and unzip the jboss_v42_agent_3.zip distribution file from the Oracle E-Delivery Web site:**

http://edelivery.oracle.com/

The following table shows the files and directories after you unzip the agent distribution file, which are in the following directory:

*Agent-Home*/j2ee_agents/jboss_v42_agent, where *Agent-Home* is where you unzipped the agent distribution file.

For example: /agents/j2ee_agents/jboss_v42_agent

| File or Directory | Description |
|---|---|
| README.txt and license.txt | Readme and license files |

| File or Directory | Description |
|---|---|
| /bin | agentadmin and agentadmin.bat programs |
| /config | Template, properties, and XML files |
| /data | license.log file. Do not edit this file. |
| /etc | Agent application (agentapp.war) and related files. The agent application is a housekeeping application used by the agent for notifications and other functions such as cross-domain single sign-on (CDSSO). For information, see "Deploying the Agent Application" on page 20. |
| /installer-logs | Log files generated when you run the agentadmin or agentadmin.bat program:<br>■ /audit contains local audit trail for the agent instance.<br>■ /debug contains the debug files for the agent instance when the agent runs in debug mode. |
| /lib | Required JAR files |
| /locale | Required properties files |
| /sampleapp | Policy agent sample application. For information, see "Deploying the Java EE Policy Agent Sample Application" on page 24. |

## Creating a Password File

A password file is an ASCII text file with only one line specifying the password in clear text. By using a password file, you are not forced to expose a password at the command line during the agent installation. When you install the JBoss Application Server 4.x/5.x agent using the agentadmin program, you are prompted to specify paths to following password files:

- An **agent profile password file** is required for both the agentadmin default and custom installation options.
- An **agent administrator password file** is required only if you use the custom installation option and have the agentadmin program automatically create the agent profile in OpenSSO Enterprise server during the installation.

### ▼ To Create a Password File

1 **Create an ASCII text file for the agent profile. For example: /tmp/jbossagentpw**

2 **If you want the `agentadmin` program to automatically create the agent profile in OpenSSO Enterprise server during the installation, create another password file for the agent administrator. For example: `/tmp/agentadminpw`**

If you wish, you can specify `amadmin` as the agent administrator when you run the install program.

3 **Using a text editor, enter the appropriate password in clear text on the first line in each file.**

4 **Secure each password file appropriately, depending on the requirements for your deployment.**

**Next Steps** Make a note of the password file names and passwords. You will need this information when you create the agent profile and install the agent using the `agentadmin` program.

## Creating an Agent Profile

The JBoss Application Server 4.x/5.x agent uses an agent profile to communicate with OpenSSO Enterprise server. You can create an agent profile using any of these three methods:

- Allow the `agentadmin` program to create the agent profile during installation when you run the `--custom-install` option. The program prompts you for this information:
  - Agent profile name and path to the agent profile password file
  - Agent administrator name and path to the agent administrator password file
- Use the OpenSSO Enterprise Console.
- Use the `ssoadm` command-line utility with the `create-agent` subcommand. For more information about the `ssoadm` command, see the *Sun OpenSSO Enterprise 8.0 Administration Reference*.

### ▼ To Create an Agent Profile in the OpenSSO Enterprise Console

1 **Login into the OpenSSO Enterprise Administration Console as `amAdmin`.**

2 **Click Access Control, *realm-name*, Agents, and then J2EE.**

3 **Under Agent, click New.**

4 **In the Name field, enter the name for the new agent profile. For example: `JBossAgentProfile`**

5 **Enter and confirm the Password.**

**Important**: This password must be the same password that you enter in the agent profile password file that you specify when you run the `agentadmin` program to install the agent.

6 **In the Server URL field, enter the OpenSSO Enterprise server URL.**

For example: `http://opensso-host.example.com:`*port-number*`/opensso`

7 **In the Agent URL field, enter the URL for the agent application (`agentapp`).**

For example: `http://agent-host.example.com:`*port-number*`/agentapp`

The `agentapp` is a housekeeping application used by the agent for notifications and other functions such as cross domain single sign-on (CDSSO) support.

8 **Click Create.**

The console creates the agent profile and displays the J2EE Agent page again with a link to the new agent profile, `JBossAgentProfile`.

To do additional configuration for the agent profile, click this link to display the `Edit` agent page. For information about the agent configuration fields, see the Console online Help. Also, see the `readme.txt` file for information about configuring the agent profile.

If you prefer, you can also use the `ssoadm` command-line utility to edit the agent profile. For more information, see the *Sun OpenSSO Enterprise 8.0 Administration Reference*.

**Next Steps** Make a note of the values you specify for the agent profile, such as the Server URL and Agent URL. You will need this information when you install the agent using the `agentadmin` program.

## Creating an Agent Administrator

An agent administrator can manage agents in OpenSSO Enterprise, including:

- **Agent management**: Use the agent administrator to manage agents either in the OpenSSO Enterprise Console or by executing the `ssoadm` utility.
- **Agent installation**: If you install the agent using the custom installation option (`agentadmin --custom-install`) and want to have the installation program create the agent profile, specify the agent administrator (and password file) when you are prompted.

  If you prefer, you can specify `amadmin` as the agent administrator when you run the install program.

### ▼ To Create an Agent Administrator

1 **Login to OpenSSO Enterprise Administration Console.**

2 **Create a new agents administrator group:**

a. **Click Access Control,** *realm-name*, **Subjects, and then Group.**

b. **Click New.**

c. **In ID, enter the name of the group. For example: `agentadmingroup`**

d. **Click OK.**

3 **Create a new agent administrator user and add the agent administrator user to the agents administrator group:**

a. **Click Access Control,** *realm-name***, Subjects, and then User.**

b. **Click New and provide the following values:**

- **ID**: Name of the agent administrator. For example: `agentadminuser`

  This is the name you will use to login to the OpenSSO Enterprise Console .
- **First Name** (optional), **Last Name**, and **Full Name**.

  For simplicity, use the same name for each of these values that you specified for ID.
- **Password** (and confirmation)
- **User Status**: Active

c. **Click OK.**

d. **Click the new agent administrator name.**

e. **On the Edit User page, click Group.**

f. **Add the agents administrator group from Available to Selected.**

g. **Click Save.**

4 **Assign read and write access to the agents administrator group:**

a. **Click Access Control,** *realm-name***, Privileges and then on the new agents administrator group link.**

b. **Check "Read and write access to all configured Agents".**

c. **Click Save.**

**Next Steps** Login into the OpenSSO Enterprise Console as the new agent administrator. The only available top-level tab is Access Control. Under *realm-name*, you will see only the Agents tab and sub tabs.

## Configuring the JBoss Application Server 4.x/5.x Agent for Apache CXF

The JBoss Application Server 4.x/5.x agent is supported with JBoss Application Server 5.x and Apache CXF 2.2.5. However, because Apache CXF uses JAXB 2.x, you must first make the configuration changes described in this section.

For information about Apache CXF, see http://cxf.apache.org/.

### ▼ To Configure the JBoss Application Server 4.x/5.x Agent for Apache CXF

1 **After you download and unzip the JBoss Application Server 4.x/5.x agent distribution file, locate the openssoclientsdk.jar file in the following directory:**

*Agent-Home*/j2ee_agents/jboss_v42_agent/lib

*Agent-Home* is where you unzipped the agent distribution file.

2 **Extract the files from openssoclientsdk.jar. For example:**

```
cd /agents/j2ee_agents/jboss_v42_agent/lib
jar xvf openssoclientsdk.jar
```

3 **In each jaxb.properties file, set the javax.xml.bind.context.factory property to the v2 ContextFactory class:**

**Old value**: javax.xml.bind.context.factory=com.sun.xml.bind.ContextFactory_1_0_1

**New value**: javax.xml.bind.context.factory=com.sun.xml.bind.v2.ContextFactory

**Note**: The openssoclientsdk.jar contains a number of different jaxb.properties files in various subdirectories. You must edit each of these files, so consider writing a script to edit the files.

4 **Generate a revised openssoclientsdk.jar file. For example:**

```
jar uvf openssoclientsdk.jar *
```

# Installing the JBoss Application Server 4.x/5.x Agent

- "Gathering Information to Install the JBoss Application Server 4.x/5.x Agent" on page 12
- "Installing the JBoss Application Server 4.x/5.x Agent Using the agentadmin Program" on page 14
- "Considering Specific Deployment Scenarios for the JBoss Application Server 4.x/5.x Agent" on page 18

# Gathering Information to Install the JBoss Application Server 4.x/5.x Agent

The following table describes the information you will need to provide when you run the `agentadmin` program to install the JBoss Application Server 4.x/5.x agent. For some `agentadmin` prompts, you can accept the default value displayed by the program, if you prefer.

**Note**. The examples and sample runs in this guide refer to JBoss AS 4.x; however, JBoss 5.x is also supported.

TABLE 2 Information Required to Install the JBoss Application Server 4.x/5.x Agent

| Prompt | Description |
|---|---|
| Config Directory Path | Path to the configuration directory for the JBoss Application Server instance.<br>Applies to both default and custom installation options.<br>For example: `/opt/jboss-4.2.3.GA/server/default/conf` |
| Home Directory Path | Path to the JBoss Application Server home directory.<br>Applies to both default and custom installation options.<br>For example: `/opt/jboss-4.2.3.GA` |
| OpenSSO server URL | OpenSSO Enterprise server URL, including the deployment URI.<br>Applies to both default and custom installation options.<br>For example: `http://opensso-host.example.com:8080/opensso` |
| Agent URL | Agent URL, including the deployment URI for the agent application.<br>Applies to both default and custom installation options.<br>For example: `http://agent-host.example.com:8090/agentapp`<br>The agent application (`agentapp.war`) is a housekeeping application used by the agent for notifications and other functions such as cross-domain single sign-on (CDSSO). For information, see "Deploying the Agent Application" on page 20. |
| Encryption Key | Key used to encrypt the agent profile password. The encryption key should be at least 12 characters long. You can accept the default key or create a new key using the `agentadmin --getEncryptKey` command.<br>Applies only to the custom installation option. |

TABLE 2 Information Required to Install the JBoss Application Server 4.x/5.x Agent *(Continued)*

| Prompt | Description |
|---|---|
| Agent profile name | A policy agent communicates with OpenSSO Enterprise using the name and password in the agent profile.<br>Applies to both default and custom installation options.<br>For information, see "Creating an Agent Profile" on page 8. |
| Agent profile password file name | Path to the agent profile password file, which is ASCII text file with only one line specifying the agent profile password. You create the agent profile password file as a pre-installation step.<br>Applies to both default and custom installation options.<br>For information, see "Creating a Password File" on page 7. |
| Option to the create the agent profile<br>The `agentadmin` program displays the following prompt if the agent profile previously specified for the Agent Profile Name prompt does not already exist in OpenSSO Enterprise:<br>`Enter true if the Agent Profile is being created into OpenSSO by the installer. Enter false if it will be not be created by installer.` | To have the installation program create the agent profile, enter `true`. The program then prompts you for:<br>■ Agent administrator who can create, update, or delete the agent profile. For example: `agentadmin` **Important**: To use this option, the agent administrator must already exist in OpenSSO Enterprise server. For information see, "Creating an Agent Administrator" on page 9. If you prefer, you can specify `amadmin` as this user.<br>■ Path to the agent administrator password file. For information, see "Creating a Password File" on page 7.<br>Applies only to the custom installation option. |
| Option to add Java security permissions to the JBoss Application Server security policy file | Indicates whether the JBoss Application Server instance runs with the Java Security Manager enabled and you want the installer to add the Java security permissions to the security policy file.<br>■ `true` — The installer adds the Java security permissions to the JBoss Application Server security policy file. The installer first displays the `server.policy` file in the configuration directory. If the JBoss Application Server instance is using a different serve policy file, specify the path to that file.<br>■ `false` (default) — The installer does not add Java security permissions to the security policy file.<br>Applies only to the default installation option. |

# Installing the JBoss Application Server 4.x/5.x Agent Using the `agentadmin` Program

The version 3.0 `agentadmin` program includes these installation options:

- Default install (`agentadmin --install`): The program asks a limited number of questions and uses default values for the other options. Use the default install option when the default options, as shown in Table 2, meet your deployment requirements.

  or

- Custom install (`agentadmin --custom-install`): The program asks a full set of questions similar to the version 2.2 program. Use the custom install option when you want to specify values other than the default options shown in Table 2.

Before you install the JBoss Application Server 4.x/5.x agent:

- An OpenSSO Enterprise server instance must be installed and running. To check the server, specify the server URL. For example: `http://opensso-host.example.com:8080/opensso`
- A JBoss Application Server server instance must be installed and configured on the machine where you plan to install the agent. For documentation, see `http://www.jboss.org/`.
- You must have downloaded and unzipped the distribution file, as described in "Downloading and Unzipping the `jboss_v42_agent_3.zip` Distribution File" on page 6.

## ▼ To Install the JBoss Application Server 4.x/5.x Agent Using the `agentadmin` Program

1. **Log into the host server where you want to install the agent.**

   **Important**: To install the agent, you must have write permission to the JBoss Application Server instance files and directories.

2. **If the JBoss Application Server instance is running, shut it down.**

3. **Change to the following directory:**

   *PolicyAgent-base*`/bin`

4. **On Solaris and Linux systems, set the permissions for the `agentadmin` program as follows, if needed:**

   ```
   # chmod 755 agentadmin
   ```

5. **Start the agent installation:**

   Default install: `# ./agentadmin --install`

   or

Custom install: `# ./agentadmin --custom-install`

On Windows systems, run the `agentadmin.bat` program.

6 **Enter information as requested by the `agentadmin` program, or accept the default values displayed by the program.**

After you have made your choices, the `agentadmin` program displays a summary of your responses. For example, for a custom installation:

```
-----------------------------------------------
SUMMARY OF YOUR RESPONSES
-----------------------------------------------

JBoss Server Config Directory : /opt/jboss-4.2.3.GA/server/default/conf
JBoss Server Home Directory : /opt/jboss-4.2.3.GA
OpenSSO server URL : http://opensso-host.example.com:8080/opensso
Agent URL : http://agent-host.example.com:8090/agentapp
Agent Profile name : JBossAgentProfile
Agent Profile Password file name : /tmp/jbossagentpw
Agent permissions gets added to java permissions policy file : false
```

7 **Verify your choices and either continue with the installation (selection 1, the default) , or make any necessary changes.**

If you continue, the program installs the agent and displays a summary of the installation. For example, for a custom installation:

```
SUMMARY OF AGENT INSTALLATION
-----------------------------
Agent instance name: Agent_001
Agent Bootstrap file location:
/agents/j2ee_agents/jboss_v42_agent/Agent_001/config/OpenSSOAgentBootstrap.properties
Agent Configuration file location
/agents/j2ee_agents/jboss_v42_agent/Agent_001/config/OpenSSOAgentConfiguration.properties
Agent Audit directory location:
/agents/j2ee_agents/jboss_v42_agent/Agent_001/logs/audit
Agent Debug directory location:
/agents/j2ee_agents/jboss_v42_agent/Agent_001/logs/debug
Install log file location:
/agents/j2ee_agents/jboss_v42_agent/installer-logs/audit/install.log
```

8 **After the installation finishes successfully, if you wish, check the installation logs in the following directory:**

```
installer-logs/audit
```

9 **Start the JBoss Application Server instance.**

## Example 1 Sample `agentadmin` Program Installation for the JBoss Application Server 4.x/5.x Agent

```
************************************************************************
Welcome to the OpenSSO Policy Agent for JBoss Server 4.x.
************************************************************************
```

```
Enter the complete path to the directory which is used by JBoss Server to store
its configuration Files. This directory uniquely identifies the JBoss
Server instance that is secured by this Agent.
[ ? : Help, ! : Exit ]
Enter the JBoss Server Config Directory Path
[/opt/jboss-4.2.3/server/default/conf]: /opt/jboss-4.2.3.GA/server/default/conf

Enter the complete path to the directory where JBoss Server home directory is
stored
[ ? : Help, < : Back, ! : Exit ]
Enter the JBoss Server Home Directory Path
[/opt/jboss-4.2.3.GA]: /opt/jboss-4.2.3.GA

Enter the URL where the OpenSSO server is running. Please include the
deployment URI also as shown below:
(http://opensso.sample.com:58080/opensso)
[ ? : Help, < : Back, ! : Exit ]
OpenSSO server URL: http://opensso-host.example.com:8080/opensso

Enter the Agent URL. Please include the deployment URI also as shown below:
(http://agent1.sample.com:1234/agentapp)
[ ? : Help, < : Back, ! : Exit ]
Agent URL: http://agent-host.example.com:8090/agentapp

Enter the Agent profile name
[ ? : Help, < : Back, ! : Exit ]
Enter the Agent Profile name: JBossAgentProfile

Enter the path to a file that contains the password to be used for identifying
the Agent.
[ ? : Help, < : Back, ! : Exit ]
Enter the path to the password file: /tmp/jbossagentpw

Indicate the specified server instance runs with Java security manager
permissions.
[ ? : Help, < : Back, ! : Exit ]
Specify whether the chosen server instance runs with Java security manager
permissions. [false]: false

-----------------------------------------------
SUMMARY OF YOUR RESPONSES
-----------------------------------------------
JBoss Server Config Directory :
/opt/jboss-4.2.3.GA/server/default/conf
JBoss Server Home Directory : /opt/jboss-4.2.3.GA
OpenSSO server URL : http://opensso-host.example.com:8080/opensso
Agent URL : http://agent-host.example.com:8090/agentapp
Agent Profile name : JBossAgentProfile
Agent Profile Password file name : /tmp/jbossagentpw
Agent permissions gets added to java permissions policy file : false
Verify your settings above and decide from the choices below.
1. Continue with Installation
2. Back to the last interaction
3. Start Over
4. Exit
Please make your selection [1]: 1
Creating directory layout and configuring Agent file for Agent_001 instance ...DONE.
Reading data from file
/tmp/jbossagentpw and
```

```
encrypting it ...DONE.
Generating audit log file name ...DONE.
Creating tag swapped OpenSSOAgentBootstrap.properties file for instance
Agent_001 ...DONE.
Creating a backup for file
/opt/jboss-4.2.3.GA/server/default/conf/jboss-service.xml
...DONE.
Adding Agent parameters to
/opt/jboss-4.2.3.GA/server/default/conf/jboss-service.xml
file ...DONE.
Creating a backup for file null ...DONE.
Adding Agent parameters to null file ...DONE.
Adding Agent parameters to am-login-config.xml file ...DONE.
Adding Agent parameters to
/opt/jboss-4.2.3.GA/bin/setAgentClasspathdefault.sh
file ...DONE.
Adding Agent parameters to agentapp.war file ...DONE.

SUMMARY OF AGENT INSTALLATION
-----------------------------
Agent instance name: Agent_001
Agent Bootstrap file location:
/agents/j2ee_agents/jboss_v42_agent/Agent_001/config/OpenSSOAgentBootstrap.properties
Agent Configuration file location
/agents/j2ee_agents/jboss_v42_agent/Agent_001/config/OpenSSOAgentConfiguration.properties
Agent Audit directory location:
/agents/j2ee_agents/jboss_v42_agent/Agent_001/logs/audit
Agent Debug directory location:
/agents/j2ee_agents/jboss_v42_agent/Agent_001/logs/debug

Install log file location:
/agents/j2ee_agents/jboss_v42_agent/installer-logs/audit/install.log

Thank you for using OpenSSO Policy Agent 3.0.
```

## After You Finish the Install

### Agent Instance Directory

The installation program creates the following directory for each agent instance:

*PolicyAgent-base*/`Agent_`*nnn*

- *PolicyAgent-base* is *Agent-Home*/`j2ee_agents/jboss_v42_agent`, where *Agent-Home* is where you unzipped the agent distribution file.

  For example: `/agents/j2ee_agents/jboss_v42_agent`
- *nnn* identifies the agent instance as `Agent_001`, `Agent_002`, and so on for each additional agent instance.

Each agent instance directory contains the following subdirectories:

- `/config` contains the configuration files for the agent instance, including `OpenSSOAgentBootstrap.properties` and `OpenSSOAgentConfiguration.properties`.
- `/installer-logs` contains the following subdirectories

- `/audit` contains local audit trail for the agent instance.
- `/debug` contains the debug files for the agent instance when the agent runs in debug mode.

# Considering Specific Deployment Scenarios for the JBoss Application Server 4.x/5.x Agent

- "Installing the JBoss Application Server 4.x/5.x Agent on Multiple JBoss Application Server Instances" on page 18
- "Installing the JBoss Application Server 4.x/5.x Agent on the OpenSSO Enterprise Host Machine" on page 18

## Installing the JBoss Application Server 4.x/5.x Agent on Multiple JBoss Application Server Instances

You can install the JBoss Application Server 4.x/5.x agent on multiple JBoss Application Server instances on the same host machine. However, you must run the `agentadmin` program for each JBoss Application Server instance. During each installation, specify the unique server configuration directory and instance name, so the agent can differentiate the different instances.

## Installing the JBoss Application Server 4.x/5.x Agent on the OpenSSO Enterprise Host Machine

You can install the JBoss Application Server 4.x/5.x agent on a different web container instance on the same host machine where OpenSSO Enterprise server is installed, as long as the web container is supported for both the JBoss Application Server 4.x/5.x agent and OpenSSO Enterprise server.

# Required Post-Installation Tasks for the JBoss Application Server 4.x/5.x Agent

- "Setting the `JBOSS_CLASSPATH` Variable for the JBoss Application Server Instance" on page 19
- "Deploying the Agent Application" on page 20
- "Installing the Agent Filter for an Application Protected by the JBoss Application Server 4.x/5.x Agent" on page 20

# Setting the JBOSS_CLASSPATH Variable for the JBoss Application Server Instance

You must set the JBOSS_CLASSPATH variable for the JBoss Application Server 4.x/5.x configuration and locale directories.

To set the JBOSS_CLASSPATH variable, modify the JBoss startup script, depending on your platform:

- Solaris and Linux systems: JBOSS_HOME/bin/run.sh
- Windows: JBOSS_HOME\bin\run.bat

## ▼ To Set the JBOSS_CLASSPATH Variable on Solaris and Linux Systems

1 **In the JBOSS_HOME/bin/run.sh script, find the following lines:**

```
if [ "x$JBOSS_CLASSPATH" = "x" ]; then
  JBOSS_CLASSPATH="$JBOSS_BOOT_CLASSPATH:$JAVAC_JAR"
  else
  JBOSS_CLASSPATH="$JBOSS_CLASSPATH:$JBOSS_BOOT_CLASSPATH:$JAVAC_JAR"
fi
```

2 **After the lines you found in Step 1, add the following new lines:**

```
CONFIG=$2
if [ "x$1"="x" ] && [ "x$CONFIG" = "x" ]; then CONFIG=default; fi
if [ -r "setAgentClasspath$CONFIG.sh" ]; then
          . /opt/jboss-4.2.3.GA/bin/setAgentClasspath$CONFIG.sh
fi
```

**Note**: The previous command is for JBoss Application Server 4.2.3.GA. If you are using a different version, specify the appropriate JBoss Application Server directory.

3 **Save the change.**

## ▼ To Set the JBOSS_CLASSPATH Variable on Windows Systems

1 **In the JBOSS_HOME\bin\run.bat script, find the following lines:**

```
if "%JBOSS_CLASSPATH%" == "" (
           set JBOSS_CLASSPATH=%JAVAC_JAR%;%RUNJAR%
) ELSE (
           set JBOSS_CLASSPATH=%JBOSS_CLASSPATH%;%JAVAC_JAR%;%RUNJAR%
)
```

2 **After the lines you found in Step 1, add the following new lines:**

```
set CONFIG=%2%
if "x%CONFIG%" == "x" (
      set CONFIG=default
)
```

```
if exist setAgentClasspath%CONFIG%.bat (
      call c:\jboss-4.2.3.GA\bin\setAgentClasspath%CONFIG%.bat
)
```

**Note**: The previous command is for JBoss Application Server 4.2.3.GA. If you are using a different version, specify the appropriate JBoss Application Server directory.

3 **Save the change.**

## Deploying the Agent Application

The agent application (`agentapp.war`) is a housekeeping application used by the agent for notifications and other functions such as cross domain single sign-on (CDSSO) support.

### ▼ To Deploy the Agent Application

1 **The agent application (`agentapp.war`) is bundled with the jboss_v42_agent_3.zip distribution file and is available as follows after you unzip the file:**

*PolicyAgent-base*`/etc/agentapp.war`

2 **Deploy `agentapp.war` on the JBoss Application Server instance using the JBoss administration console or deployment command.**

**Important**: You must use the same deployment URI that you specified for the "Agent URL" prompt during the agent installation. For example, if you accepted the default value (`/agentapp`) as the deployment URI for the agent application, use this same URI to deploy `agentapp.war`.

## Installing the Agent Filter for an Application Protected by the JBoss Application Server 4.x/5.x Agent

You install the agent filter by modifying the deployment descriptor of each application that you want to protect with the JBoss Application Server 4.x/5.x agent.

### ▼ To Install the Agent Filter for an Application Protected by the JBoss Application Server 4.x/5.x Agent

1 **Ensure that the application you want to protect is not currently deployed on JBoss Application Server.**

If the application is deployed, undeploy it before continuing.

2 **Backup the application's `web.xml` file before you modify the deployment descriptor.**

The backup copy can be useful if you need to uninstall the agent later.

3 **Edit the deployment descriptors in the application's `web.xml` file as follows:**

a. **Set the `<DOCTYPE>` element as shown in the following example:**

```
<!DOCTYPE web-app version="2.4"
xmlns="http://java.sun.com/xml/ns/j2ee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd">
```

**Note**: JBoss Application Server supports the Java Servlet specification version 2.4. Version 2.4 is fully backward compatible with version 2.3. Therefore, all existing servlets should work without modification or recompilation.

b. **Add the `<filter>` elements to the deployment descriptor.**

Specify the agent filter as the first `<filter>` element and the agent filter mapping as the first `<filter-mapping>` element. For example:

```
<web-app>
...
    <filter>
        <filter-name>Agent</filter-name>
        <filter-class>com.sun.identity.agents.filter.AmAgentFilter</filter-class>
    </filter>
    <filter-mapping>
        <filter-name>Agent</filter-name>
        <url-pattern>/*</url-pattern>
        <dispatcher>REQUEST</dispatcher>
        <dispatcher>INCLUDE</dispatcher>
        <dispatcher>FORWARD</dispatcher>
        <dispatcher>ERROR</dispatcher>
    </filter-mapping>
...
</web-app>
```

4 **In the applications's `jboss-web.xml` and `jboss.xml` files, specify the value of the `security-domain` element as `AMRealm`. For example:**

```
<security-domain>java:/jaas/AMRealm</security-domain>
```

5 **Restart the JBoss Application Server instance.**

6 **Deploy (or redeploy) the application on the JBoss Application Server web container.**

The agent filter is then added for the application.

**Next Steps** You can also protect an application with Java EE declarative security. To learn more about protecting your application with Java EE declarative security, consider "Deploying the Java EE Policy Agent Sample Application" on page 24.

# Optional Post-Installation Tasks for the JBoss Application Server 4.x/5.x Agent

## Changing the Password for an Agent Profile

After you install the agent, you can change the agent profile password, if required for your deployment.

### ▼ To Change the Password for an Agent Profile

1 **On the OpenSSO Enterprise server:**

a. **Login into the OpenSSO Administration Console.**

b. **Click Access Control,** *realm-name*, **Agents, J2EE, and then the name of the agent profile you want to update.**

The Console displays the Edit page for the agent profile.

c. **Enter and confirm the new unencrypted password.**

d. **Click Save.**

2 **On the server where the JBoss Application Server 4.x/5.x agent is installed:**

a. **In the agent profile password file, replace the old password with the new unencrypted password.**

b. **Change to the** *PolicyAgent-base*`/bin` **directory.**

c. **Encrypt the new password using the `agentadmin --encrypt` command following this syntax.**

`agentadmin --encrypt` *agent-instance password-file*

For example:

```
# ./agentadmin --encrypt Agent_001 jbossagentpw
```

The agentadmin --encrypt command returns the new encrypted password. For example:

```
ASEWEJIowNBJHTv1UGD324kmT==
```

d. **In the *agent-instance*/config/OpenSSOAgentBootstrap.properties file, set the following property to the new encrypted password from the previous step. For example:**

```
com.iplanet.am.service.secret=ASEWEJIowNBJHTv1UGD324kmT==
```

e. **Restart the JBoss Application Server instance that is being protected by the policy agent.**

## Creating the Necessary URL Policies

If the JBoss Application Server 4.x/5.x agent is configured to operate in the URL_POLICY or ALL filter mode, you must create the appropriate URL policies. For instance, if the agent is available on port 8080 using the HTTP protocol, you must create at minimum, a policy to allow access to the following resource:

```
http://myhost.mydomain.com:8080/agentsample
```

where agentsample is the context URI for the sample application.

If no policies are defined and the agent is configured to operate in the URL_POLICY or ALL filter mode, then no user is allowed access to the resources protected by the JBoss Application Server 4.x/5.x agent.

For more information, see:

- Agent sample application readme.txt file in the /sampleapp directory
- *Sun OpenSSO Enterprise 8.0 Administration Guide* to create these policies using the OpenSSO Enterprise Console or command-line utilities

## Enabling Programmatic Web Logins for the JBoss Application Server 4.x/5.x Agent

JBoss Application Server 4.2.3 and later supports programmatic web logins, which allows the agent to programmatically authenticate against the JBoss Application Server web container. This feature uses the org.jboss.web.tomcat.security.login.WebAuthentication class. For more information, see http://community.jboss.org/wiki/WebAuthentication.

The JBoss Application Server 4.x/5.x agent uses the com.sun.identity.agents.config.jboss.webauth.available property to enable or disable this feature. The default is false.

For the JBoss Application Server 4.x/5.x agent, set this property depending on the agent configuration:

- If the agent configuration is local, set the property in the agent's `OpenSSOAgentConfiguration.properties` file.
- If the agent configuration is centralized, set the property in the OpenSSO Enterprise Administration Console, as follows.

### ▼ To Enable Programmatic Web Logins for the JBoss Application Server 4.x/5.x Agent in the Console

1. **Log in to the OpenSSO Enterprise Administration Console.**

2. **Click Access Control,** *realm-name*, **Agents, J2EE, and then the name of the JBoss Application Server 4.x/5.xJBoss Application Server 4.x/5.x agent.**

3. **Enable programmatic web logins, depending on your version of OpenSSO Enterprise:**

   - For OpenSSO Enterprise 8.0 RTM, under the JBoss Application Server 4.x/5.x agent profile name, click Advanced, and then Custom Properties. Add the following property and click Save.

     ```
     com.sun.identity.agents.config.jboss.webauth.available=true
     ```

   - In later builds of OpenSSO Enterprise 8.0, under the JBoss Application Server 4.x/5.x agent profile name, click Advanced and then JBoss Application Server. Check Enabled for Web Authentication Available and then click Save.

The `com.sun.identity.agents.config.jboss.webauth.available` property is hot-swappable, so you do not need to restart the OpenSSO Enterprise web container for the value to take effect.

## Deploying the Java EE Policy Agent Sample Application

Deploying the policy agent sample application is optional. However. after you install the JBoss Application Server 4.x/5.x agent, consider deploying the sample application to help you better understand the key features, functions, and configuration options of Java EE agents, including:

- Single sign-on (SSO)
- Web-tier declarative security
- Programmatic security
- URL policy evaluation
- Session, policy, and profile attribute fetch

The sample application can be especially useful if you are writing a custom agent application.

After you install the JBoss Application Server 4.x/5.x agent, the sample application is available as:

*PolicyAgent-base*/sampleapp/dist/agentsample.ear

For information about compiling, deploying, and running the sample application, see the readme.txt file in the /sampleapp directory.

# Configuring Web Services Security for the JBoss Application Server 4.x/5.x Agent

The JBoss Application Server 4.x/5.x agent supports Web Services Security (WSS) for web service providers. A web service provider (WSP) deployed on JBoss Application Server protected by the agent can have additional security provided by the agent. For example, you can configure the JBoss Application Server 4.x/5.x agent and OpenSSO Enterprise server to support various WSS profiles, including Username token, X509 token, and SAML2 token.

**Note –** During testing of the agent with JAX-WS web services, it was observed the JBoss Application Server has an implementation of JAX-WS that is not compatible with the com.sun.identity.wss.security.handler.SecureSOAPMessage implementation. To use WSS, you must integrate compatible JAX-WS JAR files into your deployment, or you will get a org.jboss.ws.core.soap.SOAPPartImpl.normalize() Not Implemented exception.

**About the Examples**. The examples in this section use /opt as the download and installation directory. However, if you prefer, you can use a different directory. These examples are also intended for a Solaris or Linux system. If you are running on another platform such as Windows, you will need to make changes for the paths and filenames.

## ▼ To Download Compatible JAX-WS JAR Files

You must first download and install the JAX-WS JAR files from the JAX-WS Reference Implementation (RI) project.

1 **Download and unzip JBoss Application Server 4.x/5.x in the /opt directory.**

2 **Download jaxws-ri.zip from the following site: https://jax-ws.dev.java.net/**

3 **Unzip jaxws-ri.zip in /opt.**

4 **On Solaris and Linux systems, set the JAX-WS RI shell scripts to be executable. For example:**

```
cd /opt/jaxws-ri/bin
chmod +x *.sh
```

## Configuring the StockService and StandAloneStockClient Samples

This section describes how to configure the StockService sample as the WSP and the StandAloneStockClient as the WSC. Use these samples as models to configure your own WSS applications.

- "To Configure the StockService Sample" on page 26
- "To Install the JBoss Application Server 4.x/5.x Agent and Setup the WSP and WSC" on page 29
- "To Configure the StandAloneStockClient Sample" on page 29

### ▼ To Configure the StockService Sample

1 **Create the wsp directory under /opt.**

2 **Download openssowssproviders.zip from the WSS Agent link on https://opensso.dev.java.net/public/use/index.html.**

3 **Unzip openssowssproviders.zip in /opt/wsp/.**

4 **Create the jboss directory under /opt/wsp/samples for the JBoss Application Server files. For example:**

```
cd /opt/wsp/samples
mkdir jboss
```

5 **Copy the GlassFish sample files to the new jboss directory:**

```
cp -r /opt/wsp/samples/glassfish/* /opt/wsp/samples/jboss/
```

6 **Rename glassfish.properties for JBoss Application Server:**

```
cd /opt/wsp/samples/jboss/
mv glassfish.properties jboss.properties
```

7 **In /opt/wsp/samples/jboss/jboss.properties, remove the GlassFish properties and add the following:**

```
wsp.home=/opt/wsp
jaxws.home=/opt/jaxws-ri
jaxws.lib.dir=/opt/jaxws-ri/lib
```

8 **Edit /opt/wsp/samples/jboss/StockService/build.xml, as shown in the next example.**

---

**Tip –** To create a new JBoss Application Server build.xml file, just copy the following XML statements.

---

```
<?xml version="1.0" encoding="UTF-8"?>
<project name="StockQuoteService" default="all" basedir=".">
```

```
  <description>Builds, tests, and runs the project stockclient.</description>
  <property file="../jboss.properties"/>
  <condition property="wsimport-script-suffix" value=".bat">
    <os family="windows"/>
  </condition>
  <condition property="wsimport-script-suffix" value=".sh">
    <not>
      <os family="windows"/>
    </not>
  </condition>
  <path id="build.class.path">
  <pathelement location="build/classes"/>
    <fileset dir="${jaxws.lib.dir}">
      <include name="**/*.jar"/>
    </fileset>
  </path>
  <target name="-pre-compile">
    <mkdir dir="build/classes"/>
    <mkdir dir="web/WEB-INF/classes"/>
    <exec executable="${jaxws.home}/bin/wsimport${wsimport-script-suffix}">
    <arg line="-verbose -d build/classes web/WEB-INF/wsdl/StockService/stockservice.wsdl"/>
    </exec>
    <copy file="src/java/handlers.xml" todir="web/WEB-INF/classes"/>
  </target>
  <target name="compile" depends="-pre-compile">
    <javac fork="true" destdir="build/classes" srcdir="src/java">
      <classpath refid="build.class.path" />
    </javac>
  </target>
  <target name ="war" depends="compile">
      <mkdir dir="dist"/>
    <copy todir="web/WEB-INF/classes">
      <fileset dir="build/classes" />
    </copy>
    <copy todir="web/WEB-INF/lib">
      <fileset dir="${jaxws.lib.dir}"/>
    </copy>
    <war destfile="dist/StockService.war" webxml="web/WEB-INF/web.xml">
      <zipfileset dir="web" />
    </war>
  </target>
  <target name="all">
    <antcall target="war" />
  </target>
</project>
```

9. **In the following file, change any references to `localhost` and port `8080`, depending on your deployment:**

   ```
   /opt/wsp/samples/jboss/StockService/web/WEB-INF/wsdl/StockService/stockservice.wsdl
   ```

10. **Remove `/opt/wsp/samples/jboss/StockService/web/WEB-INF/sun-web.xml`. For example:**

    ```
    cd /opt/wsp/samples/jboss/StockService/web/WEB-INF
    rm sun-web.xml
    ```

11. **In the same directory, create `sun-jaxws.xml` with the following content:**

    ```
    <?xml version="1.0" encoding="UTF-8"?>
    <endpoints
    ```

```
  xmlns=’http://java.sun.com/xml/ns/jax-ws/ri/runtime’
  version=’2.0’>
  <endpoint
    name=’StockService’
    implementation=’com.samples.StockService’
    url-pattern=’/StockService’ />
</endpoints>
```

12 **In the same directory, in `web.xml`, add the agent `<filter>`, `<filter-mapping>`, `<listener>`, `<servlet>`, and `<servlet-mapping>` entries, as follows:**

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

  <filter>
    <filter-name>Agent</filter-name>
    <filter-class> com.sun.identity.agents.filter.AmAgentFilter </filter-class>
  </filter>
  <filter-mapping>
    <filter-name>Agent</filter-name>
    <url-pattern>/*</url-pattern>
    <dispatcher>REQUEST</dispatcher>
   <dispatcher>INCLUDE</dispatcher>
    <dispatcher>FORWARD</dispatcher>
    <dispatcher>ERROR</dispatcher>
  </filter-mapping>

  <session-config>
    <session-timeout>
      30
    </session-timeout>
  </session-config>
  <welcome-file-list>
    <welcome-file>
      index.jsp
    </welcome-file>
  </welcome-file-list>

  <listener>
    <listener-class>
      com.sun.xml.ws.transport.http.servlet.WSServletContextListener
    </listener-class>
  </listener>

  <servlet>
    <description>JAX-WS endpoint</description>
    <display-name>The JAX-WS servlet</display-name>
    <servlet-name>jaxws</servlet-name>
    <servlet-class>com.sun.xml.ws.transport.http.servlet.WSServlet</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>jaxws</servlet-name>
    <url-pattern>/StockService</url-pattern>
  </servlet-mapping>
</web-app>
```

13 **Build the `StockService` WAR file. For example, using `ant`:**

```
cd /opt/wsp/samples/jboss/StockService
/share/builds/components/ant/1.6.5/bin/ant -f build.xml
```

## ▼ To Install the JBoss Application Server 4.x/5.x Agent and Setup the WSP and WSC

1 **Install and configure the JBoss Application Server 4.x/5.x agent, as described in this guide.**

As noted, the examples in this guide use `/opt` as the installation directory.

2 **Follow the general steps to configure the web service provider (WSP) and web service client (WSC) in "Web Services Security Support for J2EE Agents in Policy Agent 3.0" in *Sun OpenSSO Enterprise Policy Agent 3.0 User's Guide for J2EE Agents*.**

3 **Configure and deploy your WSP application.**

If you are deploying new web services that uses JAX-WS, this guide uses the `StockService` and `StandAloneStockClient` samples as examples to follow for your web services.

If your application is already deployed and using WSS with JAX-WS, you might need only to add the agent filter in the `web.xml` file.

4 **Start the JBoss Application Server web container.**

## ▼ To Configure the `StandAloneStockClient` Sample

1 **Change to the `StandAloneStockClient` directory:**

```
cd /opt/wsp/samples/jboss/StandAloneStockClient
```

2 **Edit the `src/com/samples/SecuringWS.java` file as follows:.**

- Change any references to `localhost` and `8080`, depending on your deployment
- Change the `providerName`string to "`wsc`".

3 **In the `/opt/wss/samples/jboss/StandAloneStockClient` directory, modify `build.xml` for JBoss Application Server rather than GlassFish:**

```
<?xml version="1.0" encoding="UTF-8"?>
<project name="StandAloneStockClient" default="default" basedir=".">
  <description>Builds, tests, and runs the project stockclient.</description>
  <property file="../jboss.properties"/>
  <property name="is.java-client.module" value="true"/>
  <target name="default" depends="run"/>
  <target name="build" depends="clean">
    <mkdir dir="build/classes"/>
    <javac srcdir="src"
    destdir="build/classes"
    classpath="xyz.jar"
    debug="on">
    <classpath>
      <pathelement location="${wsp.home}/lib/openssowssproviders.jar"/>
      <pathelement location="${wsp.home}/lib/webservices-rt.jar"/>
      <pathelement location="${wsp.home}/lib/openssoclientsdk.jar"/>
      <pathelement location="${wsp.home}/lib/xalan.jar"/>
      <pathelement location="${wsp.home}/lib/xercesImpl.jar"/>
      <pathelement location="${wsp.home}/lib/j2ee.jar"/>
      <pathelement location="${wsp.home}/lib"/>
      <pathelement path="build/classes"/>
    </classpath>
    </javac>
</target>

<target name="run" depends="build">
    <echo>java.home=${java.home}</echo>
  <java classname="com.samples.SecuringWS" fork="true">
    <classpath>
      <pathelement location="${wsp.home}/lib/openssowssproviders.jar"/>
      <pathelement location="${wsp.home}/lib/ldapjdk.jar"/>
      <pathelement location="${wsp.home}/lib/webservices-rt.jar"/>
      <pathelement location="${wsp.home}/lib/openssoclientsdk.jar"/>
      <pathelement location="${wsp.home}/lib/xalan.jar"/>
      <pathelement location="${wsp.home}/lib/xercesImpl.jar"/>
      <pathelement location="${wsp.home}/lib/j2ee.jar"/>
      <pathelement location="${wsp.home}/lib"/>
      <pathelement path="build/classes"/>
    </classpath>
    </java>
  </target>

  <target name="clean">
    <delete dir="dist"/>
    <delete dir="build"/>
  </target>
</project>
```

**4 Modify `/opt/wsp/lib/AMConfig.properties` depending on your setup, so that the `StandAloneStockClient` sample sends a secure web service request:**

```
com.iplanet.services.debug.level=error
com.iplanet.services.debug.directory=/tmp/wss
com.iplanet.am.naming.url=http://opensso-host:port/opensso/namingservice
com.sun.identity.agents.app.username=amadmin
com.iplanet.am.service.password=amadmin-password
com.iplanet.am.service.secret=
am.encryption.pwd=
com.sun.identity.client.encryptionKey=
com.iplanet.am.server.protocol=http
com.iplanet.am.server.host=opensso-host
com.iplanet.am.server.port=port
com.iplanet.am.services.deploymentDescriptor=/opensso
com.iplanet.am.cookie.name=iPlanetDirectoryPro
com.sun.identity.saml.xmlsig.keystore=/opt/wsp/resources/keystore.jks
com.sun.identity.saml.xmlsig.storepass=/opt/wsp/resources/.storepass
com.sun.identity.saml.xmlsig.keypass=/opt/wsp/resources/.keypass
com.sun.identity.saml.xmlsig.certalias=cert-alias
com.sun.identity.loginurl=http://your-opensso-hostname:port/opensso/UI/Login
com.sun.identity.liberty.authnsvc.url=http://opensso-host:port/opensso/Liberty/authnsvc
```

**5 Execute the `StandAloneStockClient`. For example:**

```
/share/builds/components/ant/1.6.5/bin/ant -f build.xml.
```

You should see the requests and responses. Also, check the JBoss Application Server agent debug file.

# Managing the JBoss Application Server 4.x/5.x Agent

OpenSSO Enterprise stores version 3.0 policy agent configuration data (as well as server configuration data) in a centralized repository. To manage this configuration data, use these options:

- OpenSSO Enterprise Administration Console

  You can manage both version 3.0 Java EE and web agents from the OpenSSO Enterprise Console. Tasks that you can perform include creating, deleting, updating, listing, and displaying agent configurations. Using the Console, you can set properties for an agent that you previously set by editing the agent's `AMAgent.properties` file.

  For more information, refer to the Administration Console online Help.
- `ssoadm` command-line utility

  The `ssoadm` utility is the command-line interface to OpenSSO Enterprise server and is available after you install the tools and utilities in the `ssoAdminTools.zip` file. The `ssoadm` utility includes subcommands to manage policy agents, including:

  - Creating, deleting, updating, listing, and displaying agent configurations
  - Creating deleting, listing, and displaying agent groups
  - Adding and removing an agent to and from a group

  For information about the `ssoadm` utility, including the syntax for each subcommand, see the *Sun OpenSSO Enterprise 8.0 Administration Reference*.

## Managing a Version 3.0 Agent With a Local Configuration

In some scenarios, you might need to deploy a version 3.0 agent using a local configuration. For example, if you deploy the agent with Access Manager 7.1 or Access Manager 7 2005Q4, which do not support centralized agent configuration, local configuration is used by default.

In this scenario, you must manage the version 3.0 agent by editing properties in the agent's local `OpenSSOAgentConfiguration.properties` file (in the same manner that you edit the `AMAgent.properties` file for version 2.2 agents).

**Caution –** A version 3.0 agent also stores configuration information in the local `OpenSSOAgentBootstrap.properties` file. The agent uses information in the bootstrap file to start and initialize itself and to communicate with OpenSSO Enterprise server. In most cases, you won't need to edit the bootstrap file; however, if you do edit the file, be very careful, or the agent might not function properly.

# Uninstalling the JBoss Application Server 4.x/5.x Agent

- "Preparing to Uninstall the JBoss Application Server 4.x/5.x Agent" on page 32
- "Uninstalling the JBoss Application Server 4.x/5.x Agent Using the `agentadmin` Program" on page 32

## Preparing to Uninstall the JBoss Application Server 4.x/5.x Agent

### ▼ To Prepare to Uninstall JBoss Application Server 4.x/5.x Agent

1. **Undeploy any applications protected by the JBoss Application Server 4.x/5.x agent.**

2. **Restore the deployment descriptors of these applications to their original deployment descriptors. (Backup files are useful here if you have them.)**

3. **Conditionally, if you are permanently removing the JBoss Application Server 4.x/5.x agent, undeploy the agent application.**

   However, if you plan to re-install this agent , you don't need to undeploy the agent application.

4. **Ensure that the JBoss Application Server instance is stopped.**

## Uninstalling the JBoss Application Server 4.x/5.x Agent Using the `agentadmin` Program

### ▼ To Uninstall the JBoss Application Server 4.x/5.x Agent

1. **Change to the following directory:**

   *PolicyAgent-base*`/bin`

**2 Issue one of the following commands:**

```
# ./agentadmin --uninstall
```

or

```
# ./agentadmin --uninstallAll
```

The `--uninstall` option removes only one instance of the agent, while the `--uninstallAll` option prompts you to remove all configured instances of the agent.

**3 The `uninstall` program prompts you for the JBoss Server configuration directory and home directory.**

Enter your responses or accept the default values.

**4 The `uninstall` program displays your responses and then asks if you want to continue:**

To continue with the uninstallation, select 1 (the default).

### Example 2 Uninstallation Sample for the JBoss Application Server 4.x/5.x Agent

```
************************************************************************
Welcome to the OpenSSO Policy Agent for JBoss Server 4.x.
************************************************************************

Enter the complete path to the directory which is used by JBoss Server
to store its configuration Files. This directory uniquely identifies the
JBoss Server instance that is secured by this Agent.
[ ? : Help, ! : Exit ]
Enter the JBoss Server Config Directory Path
[/opt/jboss-4.2.3/server/default/conf]:
/opt/jboss-4.2.3.GA/server/default/conf

Enter the complete path to the directory where JBoss Server home
directory is stored
[ ? : Help, < : Back, ! : Exit ]
Enter the JBoss Server Home Directory Path
[/opt/jboss-4.2.3.GA]: /opt/jboss-4.2.3.GA


-----------------------------------------------
SUMMARY OF YOUR RESPONSES
-----------------------------------------------
JBoss Server Config Directory : /opt/jboss-4.2.3.GA/server/default/conf
JBoss Server Home Directory : /opt/jboss-4.2.3.GA

Verify your settings above and decide from the choices below.
1. Continue with Uninstallation
2. Back to the last interaction
3. Start Over
4. Exit
Please make your selection [1]: 1

Removing Agent parameters from
/opt/jboss-4.2.3.GA/server/default/conf/jboss-service.xml
```

```
file ...DONE.

Removing Agent parameters from null file ...DONE.

Removing Agent parameters from am-login-config.xml file ...DONE.

Removing Agent parameters from
/opt/jboss-4.2.3.GA/bin/setAgentClasspathdefault.sh
file ...DONE.

Removing Agent parameters from agentapp.war file ...DONE.

Deleting the config directory
/agents/j2ee_agents/jboss_v42_agent/Agent_001/config ...DONE.


Uninstall log file location:
/agents/j2ee_agents/jboss_v42_agent/installer-logs/audit/uninstall.log

Thank you for using OpenSSO Policy Agent 3.0.
```

### After You Finish the Uninstall

- The /config directory is removed from the agent instance directory, but the /installer-logs directory still exists.
- The uninstall program creates an uninstall log file in the *PolicyAgent-base*/installer-logs/audit directory.
- The agent instance directory is not automatically removed. For example, if you uninstall the agent for Agent_001, a subsequent agent installation creates the Agent_002 instance directory. To remove an agent instance directory, you must manually remove the directory.

## Migrating a Version 2.2 Policy Agent

The version 3.0 agentadmin program includes the new --migrate option to migrate a version 2.2 agent to version 3.0. After you migrate a version 2.2 agent, the agent can use the new version 3.0 agent features.

The migration process migrates the agent's binary files, updates the agent's deployment container configuration, and converts the agent's AMAgent.properties file to the new version 3.0 OpenSSOAgentBootstrap.properties and OpenSSOAgentConfiguration.properties files.

Migrating a version 2.2 agent involves these general steps:

1. On the server where the version 2.2 agent is installed, run the version 3.0 agentadmin program with the --migrate option.

To get the version 3.0 `agentadmin` program, you must download the version 3.0 agent that corresponds to the version 2.2 agent you are migrating. For example, if you are migrating the version 2.2 JBoss Application Server agent, download the version 3.0 JBoss Application Server 4.x/5.x agent.

2. On the OpenSSO Enterprise server, run the `ssoadm` utility to create the new version 3.0 agent configuration in the centralized agent configuration repository.

   Therefore, the `ssoadm` utility must be installed from the `ssoAdminTools.zip` file on the OpenSSO Enterprise server. For information, see "Installing the OpenSSO Enterprise Utilities and Scripts" in the *Sun OpenSSO Enterprise 8.0 Installation and Configuration Guide*.

The `agentadmin` program creates a new deployment directory for the migrated agent, starting with Agent_001. The program does not modify the version 2.2 agent deployment directory files, in case you need these files after you migrate.

The following procedure, the migrated version 3.0 agent instance uses a new agent profile name, which is JBossv3Agent in the examples. The old version 2.2 and new version 3.0 agent profile passwords are the same. If you need to change the password for the new version 3.0 agent profile, see "Changing the Password for an Agent Profile" on page 22.

## ▼ To Migrate a Version 2.2 Agent:

1. **Login to the server where the version 2.2 agent is installed.**

   To migrate the agent, you must have write permission to the version 2.2 agent's deployment container files and directories.

2. **Stop the JBoss Application Server instance for the version 2.2 agent.**

3. **Create a directory to download and unzip the version 3.0 agent. For example: `v30agent`**

4. **Download and unzip the version 3.0 agent that corresponds to the version 2.2 agent you are migrating.**

   The version 3.0 agents are available from the OpenSSO project site: `https://opensso.dev.java.net/public/use/index.html`

5. **Change to the version 3.0 agent's `/bin` directory.**

   For example, if you downloaded and unzipped the version 3.0 JBoss Application Server 4.x/5.x agent in the `v30agent` directory:

   ```
   cd /v30agent/j2ee_agents/jboss_v42_agent/bin
   ```

6 **On Solaris and Linux systems, set the permissions for the `agentadmin` program as follows, if needed:**

```
# chmod 755 agentadmin
```

7 **Run the version 3.0 `agentadmin` program with the `--migrate` option. For example:**

```
./agentadmin --migrate
```

8 **When the `agentadmin` program prompts you, enter the path to the version 2.2 agent's deployment directory. For example:**

```
...
Enter the migrated agent's deployment directory:
/opt/j2ee_agents/jboss_v42_agent
...
```

In this example, `/opt` is the directory where you downloaded and upzipped the version 2.2 agent.

The `agentadmin` program migrates the version 2.2 agent.

9 **After the `agentadmin` program finishes, set the following properties:**

a. **In `Agent_`*nnn*`/config/OpenSSOAgentBootstrap.properties`, change:**

`com.sun.identity.agents.config.username =` *new-v3.0-agent-profile-name*

For example:

```
com.sun.identity.agents.config.username = JBossv3Agent
```

10 **Copy the `Agent_`*nnn*`/config/OpenSSOAgentConfiguration.properties` file to the `/bin` directory where `ssoadm` is installed on the OpenSSO Enterprise server.**

11 **In `OpenSSOAgentConfiguration.properties`, add the un-encrypted version 2.2 agent profile password at the end of the file, as follows:**

`userpassword=`*v2.2–agent-profile-password*

12 **On OpenSSO Enterprise server, create a password file for the OpenSSO Enterprise administrator (`amadmin`).**

This password file is an ASCII text file with only one line specifying the `amadmin` password in plain text. For example: `amadminpw`

13 **On OpenSSO Enterprise server, run `ssoadm` to create a new agent configuration in the OpenSSO Enterprise centralized agent configuration repository. For example:**

```
cd tools_zip_root/opensso/bin
./ssoadm create-agent -e / -b JBossv3Agent -t J2EEAgent -u amadmin
-f amadminpw -D ./OpenSSOAgentConfiguration.properties
```

In this example:

- *tools_zip_root* is the directory where you unzipped `ssoAdminTools.zip`.
- `-e /` specifies the specifies the root realm for the agent configuration.
- `-b JBossv3Agent` specifies the version 3.0 agent configuration name.
- `-t J2EEAgent` specifies the agent type for Java EE agents.
- `-u amadmin` species the OpenSSO Enterprise administrator
- `-f amadminpw` specifies the path to the administrator password file.
- `-D ./OpenSSOAgentConfiguration.properties` specifies the agent configuration file

**Caution**: After you run `ssoadm`, you might want to delete `OpenSSOAgentConfiguration.properties` from the `/bin` directory. This file contains sensitive information, including as the agent profile password, and the original file is maintained on the server where the agent is installed.

14 **Restart the JBoss Application Server instance for the migrated agent.**

**Next Steps** After you migrate the agent, you can manage the new 3.0 agent configuration using the OpenSSO Enterprise Administration Console or the `ssoadm` utility, as described in "Managing the JBoss Application Server 4.x/5.x Agent" on page 31.

# Related Information

- "Additional Resources" on page 37
- "Oracle's Accessibility Program" on page 38
- "Related Third-Party Web Sites" on page 38
- "How to Report Problems and Provide Feedback" on page 38
- "Oracle Welcomes Your Comments" on page 39

## Additional Resources

You can find additional useful information and resources at the following locations:

- Oracle Advanced Customer Services: `http://www.oracle.com/us/support/systems/advanced-customer-services/index.html`
- Sun Software Product Map: `http://www.oracle.com/us/sun/sun-products-map-075562.html`
- Sun Support Resources: `http://sunsolve.sun.com/`
- Oracle Technology Network: `http://www.oracle.com/technetwork/index.html`
- Sun Developer Services: `http://developers.sun.com/services/`

## Oracle's Accessibility Program

For information about Oracle's commitment to accessibility, see the following site:

`http://www.oracle.com/us/corporate/accessibility/index.html`

## Related Third-Party Web Sites

Third-party URLs are referenced in this document and provide additional, related information.

---

**Note –** Oracle is not responsible for the availability of third-party Web sites mentioned in this document. Oracle does not endorse and is not responsible or liable for any content, advertising, products, or other materials that are available on or through such sites or resources. Oracle will not be responsible or liable for any actual or alleged damage or loss caused by or in connection with the use of or reliance on any such content, goods, or services that are available on or through such sites or resources.

---

## How to Report Problems and Provide Feedback

If you have questions or issues, contact Oracle as follows:

- Sun Support Resources (SunSolve) services at `http://sunsolve.sun.com/`.

  This site has links to the Knowledge Base, Online Support Center, and ProductTracker, as well as to maintenance programs and support contact numbers.
- The telephone dispatch number associated with your maintenance contract

So that we can best assist you in resolving problems, please have the following information available when you contact Oracle:

- Description of the problem, including when the problem occurs and its impact on your operation
- Machine type, operating system version, web container and version, JDK version, and OpenSSO Enterprise version, including any patches or other software that might be affecting the problem
- Steps to reproduce the problem
- Any error logs or core dumps

## Oracle Welcomes Your Comments

Oracle is interested in improving its documentation and welcomes your comments and suggestions. To share your comments, go to `http://docs.sun.com/` and click `Feedback`. In the online form, provide the full document title and part number. The part number is a 7-digit or 9-digit number that can be found on the title page or in the document's URL. For example, the title of this guide is *Oracle OpenSSO Policy Agent 3.0 Guide for JBoss Application Server 4.x/5.x*, and the part number is 820-7585-12.

# Revision History

| Part Number | Date | Description |
|---|---|---|
| 820-7585–12 | November 22, 2010 | ■ Added "Configuring the JBoss Application Server 4.x/5.x Agent for Apache CXF" on page 11.<br>■ Revised outdated URLs. |
| 820-7585–11 | June 8, 2009 | Updated for JBoss Application Server 5.x. |
| 820-7585-10 | April 22, 2009 | Initial release. |