

Guide for System Center Management Pack for JBoss

Microsoft Corporation

Published: October 28, 2011

Send feedback about this document to [mpgfeed@microsoft.com](mailto:mpgfeed@microsoft.com). Please include the management pack guide name with your feedback.

The Operations Manager team encourages you to provide feedback on the management pack by providing a review on the management pack’s page in the [Management Pack Catalog](http://go.microsoft.com/fwlink/?LinkID=82105) (http://go.microsoft.com/fwlink/?LinkID=82105)

Copyright

This document is provided "as-is". Information and views expressed in this document, including URL and other Internet Web site references, may change without notice. You bear the risk of using it.

Some examples depicted herein are provided for illustration only and are fictitious.  No real association or connection is intended or should be inferred.

This document does not provide you with any legal rights to any intellectual property in any Microsoft product. You may copy and use this document for your internal, reference purposes. You may modify this document for your internal, reference purposes.

© 2011 Microsoft Corporation. All rights reserved.

Microsoft, Active Directory, Windows, and Windows Server are trademarks of the Microsoft group of companies.

All other trademarks are property of their respective owners.

Contents

[Guide for System Center Management Pack for JBoss 5](#_Toc318791910)

[Guide History 5](#_Toc318791911)

[Supported Configurations 5](#_Toc318791912)

[Files Described by this Guide 6](#_Toc318791913)

[Management Pack Purpose 6](#_Toc318791914)

[Monitoring Scenarios 6](#_Toc318791915)

[Levels of Monitoring 7](#_Toc318791916)

[Monitoring Scenarios 8](#_Toc318791917)

[Custom Application Monitoring 10](#_Toc318791918)

[How Health Rolls Up 11](#_Toc318791919)

[Configuring the JBoss Management Pack 11](#_Toc318791920)

[Import the Management Packs 11](#_Toc318791921)

[Security Configuration 12](#_Toc318791922)

[Deploy BeanSpy 13](#_Toc318791923)

[Verify BeanSpy Deployment 14](#_Toc318791924)

[Additional BeanSpy Configurations 15](#_Toc318791925)

[Enable Deep Monitoring 15](#_Toc318791926)

[Enable Performance Threshold Monitors 15](#_Toc318791927)

[Best Practice: Create a Management Pack for Customizations 16](#_Toc318791928)

[Links 17](#_Toc318791929)

[Appendix A: Management Pack Contents 18](#_Toc318791930)

[Discoveries 18](#_Toc318791931)

[Monitors 18](#_Toc318791932)

[Views 19](#_Toc318791933)

[Rules 19](#_Toc318791934)

[Appendix B: BeanSpy Configurations 20](#_Toc318791935)

[Security Configurations 20](#_Toc318791936)

[Users and Roles 21](#_Toc318791937)

[Java Policy Settings 22](#_Toc318791938)

[Enable Detailed Logging 25](#_Toc318791939)

[Configuration Parameters 28](#_Toc318791940)

[ABS\_MAX\_XML\_SIZE configuration file setting 28](#_Toc318791941)

[Sample BeanSpy Query Results 28](#_Toc318791942)

[Appendix C: Creating and Importing Certificates 30](#_Toc318791943)

[Create a Test Certificate 30](#_Toc318791944)

[Import a Certificate 31](#_Toc318791945)

# Guide for System Center Management Pack for JBoss

This guide is written based on the 7.5.10380 version of the Management Pack for JBoss.

### Guide History

|  |  |
| --- | --- |
| Release Date | Changes |
| November 19, 2010 | Original preview release of this guide. |
| July 15, 2011 | Updated beta release of this guide. |
| October 28, 2011 | Updated RC release of this guide. |
| October 28th, 2014 | Updated version support of this guide |

### Supported Configurations

The Management Pack for JBoss supports monitoring the JBoss application server versions running on the operating systems as shown in the following table.

|  |  |  |
| --- | --- | --- |
| JBoss Versions | Windows Operating Systems | UNIX and Linux Operating Systems |
| JBoss 4.2 with Java 5 or Java 6  JBoss 5.1 with Java 5 or Java 6  JBoss 6.0 with Java 5 or Java 6  JBoss 7.0\* with Java 7  Wildfly (JBoss 8.0\*) with Java 7 | Windows Server 2003 SP2  Windows Server 2003 R2 SP2  Windows Server 2008 SP2 and above  Windows Server 2008 R2 and above  Windows Server 2012 and above | CentOS:   * 5(x86/x64) * 6(x86/x64) * 7(x64)   Debian Linux:   * 5(x86/x64) * 6(x86/x64) * 7(x86/x64)   Oracle Linux:   * 5(x86/x64) * 6(x86/x64) * 7(x64)   Red Hat Enterprise Linux:  **** 4(x86/x64)  **** 5(x86/x64)  **** 6(x86/x64)   * 7(x64)   SLES:  **** 9(x86)  **** 10 sp1(x86/x64)  **** 11(x86/x64)  Ubuntu Linux Server:   * 10.04(x86/x64) * 12.04(x86/x64) * 14.04(x86/x64) |

\*Standalone and Domain mode command Line options for to change port offset are not supported on these platforms.

\*Domain mode command line options to use a different host.xml is currently not supported

\*Standalone mode command line options to use the non-default standalone.xml is currently not supported

### Files Described by this Guide

The Management Pack for JBoss pertains to the following files:

 Microsoft.JEE.JBoss.4.mp

 Microsoft.JEE.JBoss.5.mp

 Microsoft.JEE.JBoss.6.mp

* Microsoft.JEE.JBoss.7.mp
* Microsoft.JEE.Wildfly.8.mp

 Microsoft.JEE.JBoss.Library.mp

 Microsoft.JEE.Templates.Library.mpb

 Microsoft.JEE.Library.mpb

# Management Pack Purpose

The System Center Management Pack for JBoss allows an IT administrator to monitor the health of JEE application server instances in Operations Manager. In addition, it provides the option to deploy BeanSpy, an open source technology from Microsoft, to provide deeper monitoring that includes memory usage.

In this section:

 [Monitoring Scenarios](#zf07b3e51f1d04041990f8a0422d8172c)

 [How Health Rolls Up](#z5229b8e190574fa09f9c95134dde5d3a)

For details on the discoveries, rules, monitors, and views contained in this management pack, see [Appendix A: Management Pack Contents](#z5d8fd41f807e47ac9d0258aabb0ffb96).

# Monitoring Scenarios

After the management packs for the JEE application servers are imported, the instances of JBoss application servers will be automatically discovered. The discovery interval is set to 4 hours by default so discovery can take up to that length of time.

On JBoss, an application server must be running for Operations Manager to discover it for the first time. After an instance is discovered, the configuration will be removed only when the application server is uninstalled.

You can monitor instances of the JBoss Application Server by doing the following:

1. In the Operations console, click Monitoring.

2. Expand Application Monitoring, expand Java Monitoring, expand JBoss Application Servers, and select the monitoring folder of interest.

## Levels of Monitoring

The Management Pack for JBoss provides two levels of capabilities for monitoring application server instances:

 Basic Monitoring

You can automatically discover instances of an application server that are running on a managed computer, and then to monitor the basic health of those instances.

 Deep Monitoring

The Management Pack for JBoss utilizes extended capabilities when BeanSpy is installed on the managed computer. BeanSpy is an open source technology from Microsoft which relies on Java Management Extension (JMX) to enable the management pack to get detailed information from the application server instances that include the following:

 Applications deployed in the application server.

 Number of garbage collections per second.

 Time spent in garbage collection.

 JVM memory usage and capacity.

 Number of class loaded in the JVM.

 Number of active threads.

With these additional details, the IT administrator can manage the memory allocated to the JEE application servers and ensure resources are being efficiently used.

After BeanSpy is installed, the Microsoft JEE Application Server management packs can enumerate the individual Java applications loaded in the application server. This enables the IT administrator to select which applications are important to monitor. The monitored Java applications report health status, which enable the IT administrator to determine if the application is running based on the application server.

Java applications running in a JEE application server also have a mechanism for providing application-specific management information. This mechanism is called “MBeans”, and is part of the JMX standard. The application writer must choose to create custom MBeans and populate them with relevant statistics as the application runs, somewhat similar to performance counters in a Windows application.

MBeans provide appropriate domain-specific knowledge that can be the best way to understand the behavior of an application. BeanSpy retrieves information from the MBeans, and IT administrators can use a template to easily create Operations Manager rules that monitor and provide alerts on the values from the MBeans.

For installation, configuration, and other details about the BeanSpy, see [Appendix B: BeanSpy Configurations](#z419fcdba084042b5b6e47a16af78dd79).

## Monitoring Scenarios

The following table lists the monitoring scenarios provided by this management pack.

|  |  |  |  |
| --- | --- | --- | --- |
| Monitoring scenario | Monitoring Folder | Description | Associated monitors |
| Application Server Availability | Configurations | Determines whether or not the process for an application server instance is running. The Health Explorer of an application server includes the availability monitor for the application server process.  If an application server process is not running, Operations Manager shows the application server as critical, otherwise healthy. | Process availability health unit monitor for JBoss application server. |
| Application Availability | Applications | A roll up the application availability health to the monitored application server.  These applications are EAR and WAR files that are deployed JBoss application servers.  On JBoss, the application state is not monitored because applications can only be discovered when they are running. If an application is stopped, or deployed but not started, it will disappear from the JBoss application state view. | Application availability health rollup |
| Deep availability health | Deep monitored configurations | Determines whether or the application server is responding to HTTP queries. | Deep availability health unit monitor of application server |
| JMX Store health |  | The configuration health monitor for the JMX store connection in a JBoss Web application server configuration.  Operations Manager returns either a warning if the store is not healthy, otherwise success. | JMX Store configuration health monitor |
| Performance Counters | Performance | Click the checkbox next to a performance counter you are interested in, and you should be able to view the performance graph for this counter. Note that different counters in the same view may need to be scaled to appear proportionally on the same graph.  Note that performance data is collected over time. If you just started monitoring an application server, you will not be able to immediately see performance graphs in the performance view. Allow the application server run for an hour or more, and you should be able to see the graphs. |  |
| Custom Application Availability Monitoring |  | You can use the "JEE Application Availability Monitor (3 state)" and "JEE Application Performance Monitor" management pack templates to monitor custom application management information exposed through MBeans. For more information, see [Custom Application Monitoring](#z2) in this topic. | Custom Availability and Performance Monitors |

## Custom Application Monitoring

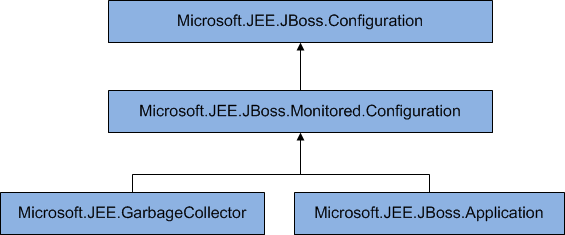
This "JEE Application Availability Monitor (3 state)" and "JEE Application Performance Monitor" management pack templates enable you to monitor information exposed through MBeans. To get the best user experience, the Operations Manager console must have HTTP or HTTPS access to the application server that has the targeted MBeans. The following procedure describes how to use the template to create a custom application monitoring scenario.

To create a custom availability monitor

|  |
| --- |
| 1. In the Operations Manager console, click Authoring.  2. Click Add Monitoring Wizard, and select JEE Application Availability Monitor (3 state) or the JEE Application Performance Monitor for the monitoring type.  3. Follow the instructions in the wizard to create a custom MBean based 3 state availability monitor or to create the performance collection rule.  The newly created monitor will appear in the Health Explorer of the application specified during monitor creation in the template wizard. |

# How Health Rolls Up

The following diagram shows how the health states of components roll up in this management pack.



# Configuring the JBoss Management Pack

This section provides guidance on configuring and tuning this management pack.

 [Import the Management Packs](#z3)

 [Security Configuration](#z4)

 [Deploy BeanSpy](#z5)

 [Verify BeanSpy Deployment](#z6)

 [Additional BeanSpy Configurations](#z7)

 [Enable Deep Monitoring](#z8)

 [Enable Performance Threshold Monitors](#z9)

 [Best Practice: Create a Management Pack for Customizations](#z10)

## Import the Management Packs

The management packs are composed of libraries and of objects that are specific to the version of the JBoss application server. Import the following library management packs:

 Microsoft.JEE.JBoss.Library.mp

 Microsoft.JEE.Templates.Library.mpb

 Microsoft.JEE.Library.mpb

Next, import the management packs required for the versions of the application servers that you are monitoring:

 Microsoft.JEE.JBoss.4.mp

 Microsoft.JEE.JBoss.5.mp

 Microsoft.JEE.JBoss.6.mp

* Microsoft.JEE.JBoss.7.mp
* Microsoft.JEE.Wildfly.8.mp

For information on how to import a management pack, or any type of management pack, see [How to Import an Operations Manager Management Pack](http://go.microsoft.com/fwlink/?LinkId=219431) in the Operations Manager Operations Guide.

## Security Configuration

If your application server requires authentication, you must create a Run As account for JEE monitoring. This management pack contains the JEE monitoring account Run as profile that must be associated with a Run as account for JEE monitoring that you create.

To create a Run As account

|  |
| --- |
| 1. Log on to the Operations console with an account that is a member of the Operations Manager Administrators role.  2. In the Operations console, click Administration.  3. In the Administration workspace, right-click Accounts, and then click Create Run As Account.  4. In the Create Run As Account Wizard, on the Introduction page click Next.  5. On the General Properties page, do the following:   Select Basic Authentication or the appropriate value in the Run As Account type list.   Type a display name in the Display Name text box.   Optionally, type a description in the Description box.   Click Next.  6. On the Credentials page, type a user name, and its password, and then select the domain for the account that you want to make a member of this Run As account.  If you installed the version of BeanSpy that does not require authentication, the account name and password can be any string.  7. Click Next.  8. On the Distribution Security page, the More secure option is recommended.  9. Click Create.  10. On the Run As Account Creation Progress page, click Close. |

To associate a Run As account to a Run As profile

|  |
| --- |
| 1. In the Operations console, click Administration.  2. In the Administration workspace, under Run As Configuration, click Profiles.  3. In the results pane, double-click the JEE Monitoring Account. The Run As Profile Wizard opens.  4. In the left pane, click Run As Accounts.  5. On the Run As Accounts page, click Add.  6. In the Add a Run As Account window, in the Run As account field, select the Run As Account that you just created.  7. Select All targeted objects or A selected class, group, or object. If you select A selected class, group, or object, click Select, and then locate and select the class, group, or object that you want the Run As account to be used for.  8. Click OK to close the Add a Run As Account window.  9. On the Run As Accounts page, click Save. |

## Deploy BeanSpy

BeanSpy is contained the Microsoft.JEE.Library.mpb, and is installed into in a folder determined by Operations Manager during installation.

Note

To deploy BeanSpy to a UNIX or Linux computer, you must first run the following procedure that copies the files to a Windows computer and then you must use a deployment method of your choosing to deploy the files to the UNIX or Linux computer.

To copy BeanSpy files to an application server

|  |
| --- |
| 1. In the Operations console, click Monitoring.  2. In the Monitoring workspace, under JEE Application Servers, click JBoss application servers.  3. In the Tasks pane, click Copy BeanSpy files.  The following BeanSpy files are copied to the computer running the selected JEE Application Server, under the folder %windir%\temp:   BeanSpy.EAR   BeanSpy.WAR   BeanSpy.Http.NoAuth.EAR   BeanSpy.Http.NoAuth.WAR  4. Deploy BeanSpy depending on your choice of authentication:   If you are using HTTPS with authentication, deploy BeanSpy.EAR.   If you are using HTTP without authentication, then rename BeanSpy.Http.NoAuth.Ear to BeanSpy.ear and deploy.   If the JBoss application server does not support EAR, then deploy BeanSpy.WAR.  These files are same for all the JEE Application Servers. So you can run the “Copy BeanSpy Files” task once, retrieve the files, and deploy them to all your application servers using the deployment method of your choice. |

After you install BeanSpy, you can determine if it is responding so that you can further monitor the application server. BeanSpy provides a better indication of the application server health than process monitoring because it verifies that the application server is responding to HTTP requests.

## Verify BeanSpy Deployment

Ensure your application server can be queried using FQDN (Fully Qualified Domain Name) such as host1.contoso.com. For example, on JBoss, this means that the JBoss application server has to be started with a “-b” option, otherwise, it can only be queried using localhost.

Verify BeanSpy is correctly installed by submitting the following BeanSpy query in your browser with your fully qualified domain name and selected port for either HTTP or HTTPS:

http://<FQDN>:<port>/BeanSpy/Stats/Info

http://<FQDN>:<port>/BeanSpy/MBeans?JMXQuery=<JMXQuery>

The following is a sample query for JBoss 4, 5, 6. Adjust the host name and port as required.

https://host1.contoso.com:8080/BeanSpy/MBeans?JMXQuery=JBoss.management.local:j2eeType=J2EEApplication,name=BeanSpy.ear,\*

If you use SSL, verify that the certificate is set up correctly as described in the previous steps. The browser should not warn about an untrusted certificate if the certificate is configured correctly.

If authentication is required, make sure the basic authentication account is configured correctly. The browser should prompt you for user name and password.

See [Configuration Parameters](#z11) in Appendix B for parameters that provide options and capabilities for using BeanSpy.

If the query is successful, there should be a XML representation of the MBeans that matched the given query. A snapshot of a sample resultant XML for each type of the application servers is provided in [Sample BeanSpy Query Results](#z12). If the query was not successful, check the following common causes for failures:

 BeanSpy is not deployed.

 BeanSpy is not started/enabled.

 A firewall is blocking the port.

 Invalid BeanSpy syntax.

 The Application Server is only listening on the localhost, not the FQDN.

## Additional BeanSpy Configurations

See [Appendix B: BeanSpy Configurations](#z419fcdba084042b5b6e47a16af78dd79) for the following configurations and information:

1. HTTP and HTPS authentication.

2. Authenticate users for a monitoring role.

3. Required Java policy settings if the Java Security Manager is enabled.

4. Enable detailed log messages.

5. Include parameters in BeanSpy queries to control the attribute depth, count, size, and time.

6. Sample BeanSpy query results.

## Enable Deep Monitoring

Deep monitoring provides extended monitoring capabilities beyond the health of application servers, such as garbage collection and memory usage statistics.

To Enable Deep Monitoring

|  |
| --- |
| 1. In the Operations console, click Monitoring.  2. In the Monitoring pane, select a JEE Application Server instance that you want to enable deep monitoring.  3. In the Tasks pane, click Enable deep monitoring using HTTP or Enable deep monitoring using HTTPS.  4. In the Enable Deep Monitoring window, click Run.  After the task completes (which can take few minutes, the JEE application server instance should appear under the Deep Monitored Configurations folder. |

## Enable Performance Threshold Monitors

The monitors for the performance counters on each application server are disabled by default because the thresholds for these monitors vary from one customer environment to another. There are three performance monitors for each application server that you can enable:

The following table lists the performance threshold monitors that are initially disabled because they may not be suitable for your environment. Before you enable a performance threshold monitor, you should baseline the relevant performance counters, and then apply the appropriate overrides to define and enable a suitable threshold for your environment.

|  |  |  |
| --- | --- | --- |
| Performance Monitor | Description | Default Value |
| Garbage Collection Rate of a Java EE Application Server | Monitors the rate at which garbage collections are happening on the JVM associated with the Java EE Application Server. | 5 collections per sampling interval. |
| Garbage Collection Time of a Java EE Application Server | Monitors the time that the garbage collector takes to perform garbage collections on the JVM associated with the application server. | 5000 milliseconds per sampling interval. |
| Performance monitor for the Percentage of Virtual Machine Memory Used on a Java EE Application Server | Monitors the percentage of used heap memory compared to maximum heap memory on an application server. | 90% |

The garbage collection monitors (2 and 3) are for each garbage collector. You can have multiple sets of garbage collection monitors.

To enable and configure performance counters

|  |
| --- |
| 1. In the Operations console, click Monitoring.  2. Expand JEE Application Servers, expand JBoss application servers, and select the Performance folder.  3. Right-click one of the performance counters to be configured, and select Monitor Properties.  4. On the Monitor Properties dialog, on the Overrides tab click Override. If you choose the memory monitor, you can either override the monitor for this application server or for all deep monitored application servers. If you choose a garbage collection monitor, you can either override the monitor for this garbage collector or for all garbage collectors in all application servers. You can also create groups for greater control in your configuration as you can with any other monitor in Operations Manager.  5. In the Override Properties dialog, enable the monitor and configure its threshold (and other properties as necessary) and apply your changes.  6. Refresh Health Explorer, it may take a few minutes before you can see that the performance counter monitor is now enabled. |

## Best Practice: Create a Management Pack for Customizations

By default, Operations Manager saves all customizations such as overrides to the Default Management Pack. As a best practice, you should instead create a separate management pack for each sealed management pack you want to customize.

When you create a management pack for the purpose of storing customized settings for a sealed management pack, it is helpful to base the name of the new management pack on the name of the management pack that it is customizing, such as “Biztalk Server 2006 Customizations”.

Creating a new management pack for storing customizations of each sealed management pack makes it easier to export the customizations from a test environment to a production environment. It also makes it easier to delete a management pack, because you must delete any dependencies before you can delete a management pack. If customizations for all management packs are saved in the Default Management Pack and you need to delete a single management pack, you must first delete the Default Management Pack, thus deleting customizations to other management packs as well.

# Links

The following links connect you to information on common tasks associated with management packs:

 [Administering the Management Pack Life Cycle](http://go.microsoft.com/fwlink/?LinkId=211463) (http://go.microsoft.com/fwlink/?LinkId=211463)

 [How to Import a Management Pack](http://go.microsoft.com/fwlink/?LinkID=142351) (http://go.microsoft.com/fwlink/?LinkID=142351)

 [How to Monitor Using Overrides](http://go.microsoft.com/fwlink/?LinkID=117777) (http://go.microsoft.com/fwlink/?LinkID=117777)

 [How to Create a Run As Account](http://go.microsoft.com/fwlink/?LinkID=165410) (http://go.microsoft.com/fwlink/?LinkID=165410)

 [How to Modify an Existing Run As Profile](http://go.microsoft.com/fwlink/?LinkID=165412) (http://go.microsoft.com/fwlink/?LinkID=165412)

 [How to Export Management Pack Customizations](http://go.microsoft.com/fwlink/?LinkId=209940) (http://go.microsoft.com/fwlink/?LinkId=209940)

 [How to Remove a Management Pack](http://go.microsoft.com/fwlink/?LinkId=209941) (http://go.microsoft.com/fwlink/?LinkId=209941)

For questions about Operations Manager and management packs, visit the [System Center Operations Manager community forum](http://go.microsoft.com/fwlink/?LinkID=179635) (http://go.microsoft.com/fwlink/?LinkID=179635).

A useful resource is the [System Center Operations Manager Unleashed blog](http://opsmgrunleashed.wordpress.com/) (http://opsmgrunleashed.wordpress.com/), which contains “By Example” posts for specific management packs.

More blogs on Operations Manager:

 [System Center Operations Manager](http://blogs.technet.com/momteam/default.aspx) (http://blogs.technet.com/momteam/default.aspx)

 [The Manageability Team Blog](http://blogs.technet.com/smsandmom/default.aspx) (http://blogs.technet.com/smsandmom/default.aspx)

 [Kevin Holman's OpsMgr Blog](http://blogs.technet.com/kevinholman/default.aspx) (http://blogs.technet.com/kevinholman/default.aspx)

 [Thoughts on OpsMgr](http://thoughtsonopsmgr.blogspot.com/) (http://thoughtsonopsmgr.blogspot.com/)

 [Raphael Burri’s blog](http://rburri.wordpress.com/) (http://rburri.wordpress.com/)

 [BWren's Management Space](http://blogs.technet.com/brianwren/default.aspx) (http://blogs.technet.com/brianwren/default.aspx)

 [The Operations Manager Support Team Blog](http://blogs.technet.com/operationsmgr/) (http://blogs.technet.com/operationsmgr/)

 [Operations Manager](http://ops-mgr.spaces.live.com) (http://ops-mgr.spaces.live.com)

 [Ops Mgr ++](http://blogs.msdn.com/boris_yanushpolsky/default.aspx) (http://blogs.msdn.com/boris\_yanushpolsky/default.aspx)

 [Notes on System Center Operations Manager](http://blogs.msdn.com/mariussutara/default.aspx) (http://blogs.msdn.com/mariussutara/default.aspx)

Important

All information and content on non-Microsoft sites is provided by the owner or the users of the Web site. Microsoft makes no warranties, express, implied, or statutory, as to the information at this Web site.

For information about monitoring UNIX and Linux computers, see [Accessing UNIX and Linux Computers in Operations Manager for System Center 2012](http://go.microsoft.com/fwlink/p/?LinkID=223881).

# Appendix A: Management Pack Contents

The Management Pack for JBoss provides the object types described in the following lists. All objects are supported by the 4.x, 5.x, 6.x, 7.x, and 8.x (Wildfly) versions of the JBoss Application Server.

## Discoveries

The following discoveries are provided for JBoss Application Server:

 Windows configurations

 Monitored Windows configurations

 UNIX and Linux configurations

 Monitored UNIX and Linux configurations

 A UNIX or Linux computer contains application server configuration.

 Attributes for a monitored JBoss application server configuration.

 Applications

JBoss 4,5,6 : BeanSpy query: jboss.management.local:j2eeType=J2EEApplication,\*

JBoss 7, Widlfly 8: BeanSpy query: jboss.as:deployment=\*.ear

 Applications for standalone web modules

JBoss 4,5,6: BeanSpy query: jboss.management.local:J2EEApplication=null,j2eeType=WebModule,\*

JBoss 7, Wildfly 8: BeanSpy query: jboss.as:deployment=\*.war

 Garbage collection

## Monitors

The following monitor is provided for Windows, UNIX, and Linux configurations:

 Process availability health unit monitor for JBoss application server

The following monitors are provided for monitored Windows, UNIX, and Linux configurations:

 JMX Store configuration health monitor

 Deep availability health unit monitor of application server

 Percentage VM memory utilized performance monitor

The following monitors are provided for garbage collection (not enabled by default):

 Garbage collection rate performance monitor

 Garbage collection time performance monitor

## Views

The following views are provided for applications:

 Application Name, J2EE Type, Object Name, Path

The following views are provided for Windows, UNIX, and Linux configuration discoveries:

 Configurations:

State, Host Name, Disk Path, HTTP Port, HTTPS Port, Version, Path

 Deep Monitored Configurations:

State, Host Name, Disk Path, HTTP Port, HTTPS Port, Version, Path, Port, Protocol

 Class loader

 Heap memory

 Garbage collector

 Threads

 JIT Compiler

## Rules

The following rules are provided to collect performance information for Windows, UNIX, and Linux configuration discoveries:

 JVM loaded class count

 JVM total loaded class count change rate

 JVM total unloaded class count change rate

 JVM peak thread count

 JVM current running thread count

 JVM total started thread count change rate

 JVM JIT compiler time change rate

 JVM initial heap memory allocated

 JVM heap memory used

 JVM maximum heap memory committed

 JVM maximum heap memory

 JVM percent heap memory used

 JVM object pending finalization (garbage collection)

# Appendix B: BeanSpy Configurations

Note

BeanSpy, an open source technology from Microsoft, is an HTTP-based JMX connector and a servlet to be installed on the application server on which you want to enable deep monitoring.

This topic contains the following sections:

 [Security Configurations](#z13)

 [Users and Roles](#z14)

 [Java Policy Settings](#z15)

 [Enable Verbose Logging](#z16)

 [Configurable Parameters](#z11)

 [Sample BeanSpy Query Results](#z12)

For information deploying BeanSpy, see [Configuring the JBoss Management Pack](#zfaccbedc14814cedb078c4341a1feab0).

## Security Configurations

BeanSpy files are digitally signed. To change the configuration parameters in the files, unzip the BeanSpy.EAR or BeanSpy.WAR files, remove the signature metadata files (manifest.mf, msftsig.rsa, msftsig.sf) and then repackage them for your deployment.

BeanSpy can be accessed through the HTTP and SSL (HTTPS) protocols, either with or without basic authentication. The following configurations are supported, listed here in the order of most secure to least secure: You can access BeanSpy through the HTTP and SSL (HTTPS) protocols. The following configurations are supported, listed here in the order of most secure to least secure:

 SSL with basic authentication (most secure)

 SSL without basic authentication

 HTTP with basic authentication

 HTTP without basic authentication (least secure)

Based on your organization’s security policies, determine whether you should configure your application server to communicate with the Operations Manager agent to use HTTP or SSL, with or without authentication.

See the procedure in [Deploy BeanSpy](#z5) for information on which files to deploy.

Caution

Using HTTP without authentication is strongly discouraged because the user name and password can be intercepted from the plain text in the HTTP protocol.

If you decide to use BeanSpy with authentication, do the following:

 If your application server is configured to use SSL, you should already have the certificate set up for your application server regardless whether or not you want to use Operations Manager to monitor it. However, to have Operations Manager monitor your application server using SSL, the CN of the certificate must be the FQDN of the computer instead of localhost or host name. In a test environment, you can use a self-signed certificate for your application server. Ensure the certificate used by the application server for SSL is trusted by the Operations Manager agent computer. See [Appendix C: Creating and Importing Certificates](#z63654f35d7d74cedb7dd020dea88192f) for how to create a test certificate for your application server and import a certificate to a computer’s trusted certificate store.

 Configure the basic authentication account for BeanSpy. The HTTPS version of BeanSpy by default requires a role called ’monitoring‘. Create a user for your application server that maps to this role in the same way you manage other users and roles in your application server. See [Users and Roles](#z14) for an example of how to create users and roles for application servers.

## Users and Roles

The BeanSpy servlet uses standard JEE application server authentication mechanisms. When authentication is required, users belonging to the "monitoring" role will be able to query BeanSpy while users belonging to the "invoke" role will be able to invoke methods on MBeans. The following procedure provides an example on how to create a user associated with a role.

JBoss 4, 5, and 6: To create a user and associate with a role

|  |
| --- |
| 1. Add an authentication domain in the ${jboss.dir/server/${server.name}/conf/login-config.xml} file, as shown in the following example:  <application-policy name=”opsmgr”>      <authentication>        <login-module code="org.jboss.security.auth.spi.UsersRolesLoginModule"          flag="required">          <module-option name="usersProperties">props/opsmgr-users.properties</module-option>          <module-option name="rolesProperties">props/opsmgr-roles.properties</module-option>        </login-module>      </authentication>    </application-policy>  2. Create opsmgr-users.properties files in the ${jboss.dir}/server/${server.name}/conf/props directory. The opsmgr-users.properties file is a text file where each entry is a pair of a username and a password separated by an equals sign, such as opsmgrmonitor=secret.  3. Create opsmgr-roles.properties files in the ${jboss.dir}/server/${server.name}/conf/props directory. The opsmgr-roles.properties file is a simple text file where each entry is a pair of username and rolename separated by an equals sign, such as: opsmgrmonitor=monitoring.  The role name "monitoring" is required by the BeanSpy servlet to query MBeans, while the role name “invoke” is required to invoke methods on MBeans. The role names must be exactly same in this opsmgr-roles.properties file. |

JBoss 7, Wildfly 8: To create a user and associate with a role

1. Create opsmgr-users.properties files in the ${jboss.dir}/standalone/configuration directory for a standalone server or in the ${jboss.dir}/domain/configuration for a domain server. The opsmgr-users.properties file is a text file where each entry is a pair of a username and a password separated by an equals sign, such as opsmgrmonitor=secret.
2. 3. Create opsmgr-roles.properties files in the ${jboss.dir}/standalone/configuration directory for a standalone server or in the ${jboss.dir}/domain/configuration directory for a domain server The opsmgr-roles.properties file is a simple text file where each entry is a pair of username and rolename separated by an equals sign, such as: opsmgrmonitor=monitoring.

The role name "monitoring" is required by the BeanSpy servlet to query MBeans, while the role name “invoke” is required to invoke methods on MBeans. The role names must be exactly same in this opsmgr-roles.properties file.

1. For standalone mode servers add the following opsmgr security domain using jboss-cli.sh or jboss-cli.bat located in ${jboss.home.dir}/bin/
   1. Connect to management port running by typing connect <hostname>:<port> inside jboss-cli
   2. Run command in jboss-cli: /subsystem=security/security-domain=opsmgr:add(cache-type=default)
   3. Run command in jboss-cli: ./subsystem=security/security-domain=opsmgr/authentication=classic:add(login-modules=[{"code"=>"org.jboss.security.auth.spi.UsersRolesLoginModule","flag"=>"required","module-options"=>[("usersProperties"=>" file:${jboss.server.config.dir}/opsmgr-users.properties "),("rolesProperties"=>" file:${jboss.server.config.dir}/opsmgr-roles.properties ")]}])
2. For domain mode servers add the following opsmgr security domain using jboss-cli.sh or jboss-cli located in ${jboss.home.dir}/bin/
   1. Connect to management port running by typing connect <hostname>:<port> inside jboss-cli

Run command in jboss-cli: /profile=full/subsystem=security/security-domain=opsmgr:add(cache-type=default)

* 1. Run command in jboss-cli for full profile (change profile as needed) : /profile=full/subsystem=security/security-domain=opsmgr:add(cache-type=default)
  2. Run command in jboss-cli for full profile (change profile as needed) : /profile=full/subsystem=security/security-domain=opsmgr/authentication=classic:add(login-modules=[{"code"=>"org.jboss.security.auth.spi.UsersRolesLoginModule","flag"=>"required","module-options"=>[("usersProperties"=>" file:${jboss.domain.config.dir}/opsmgr-users.properties "),("rolesProperties"=>" file:${jboss.domain.config.dir}/opsmgr-roles.properties ")]}])

## Java Policy Settings

You only need to configure policy settings if you are running application servers with Java Security Manager enabled.

To configure policy settings for JBoss 4.2

|  |
| --- |
| 1. JBoss 4.2 does not ship with a default policy files, so if you enable the Java Security Manager, you must add a policy file and configure it correctly. The following code is a sample policy file that demonstrates the policies needed for BeanSpy to function properly. You can append this file to the existing policy file being used with your application server.  grant codeBase "file:${jboss.server.home.dir}/tmp/deploy/-" {    permission java.util.logging.LoggingPermission       "control";    permission javax.management.MBeanServerPermission    "findMBeanServer";    permission javax.management.MBeanServerPermission    "createMBeanServer";    permission java.lang.management.ManagementPermission "monitor";    permission javax.management.MBeanPermission          "\*", "queryMBeans";    permission javax.management.MBeanPermission          "\*", "getMBeanInfo";    permission javax.management.MBeanPermission          "\*", "invoke";    permission javax.management.MBeanPermission          "\*", "getAttribute";    permission javax.management.MBeanPermission          "\*", "getDomains";    permission javax.management.MBeanPermission          "\*", "getClassLoaderRepository";      permission java.lang.RuntimePermission "accessClassInPackage.sun.util.logging.resources";    permission java.lang.RuntimePermission "accessDeclaredMembers";    permission java.lang.RuntimePermission "accessClassInPackage.org.apache.catalina.connector";    permission java.lang.RuntimePermission "accessClassInPackage.org.apache.coyote";    permission java.lang.RuntimePermission "accessClassInPackage.org.apache.coyote.http11";    permission java.lang.RuntimePermission "accessClassInPackage.org.apache.catalina.loader";    permission java.lang.RuntimePermission "accessClassInPackage.org.apache.catalina.core";    permission java.lang.RuntimePermission "accessClassInPackage.org.apache.catalina.core";    permission java.lang.RuntimePermission "accessClassInPackage.org.apache.catalina.realm";    permission java.lang.RuntimePermission "accessClassInPackage.org.apache.catalina.session";    permission java.lang.RuntimePermission "defineClassInPackage.java.security";    permission java.lang.RuntimePermission "defineClassInPackage.java.lang";    permission java.lang.RuntimePermission "getClassLoader";    permission java.lang.RuntimePermission "defineClassInPackage.java.io";    permission java.lang.RuntimePermission "defineClassInPackage.org.apache.catalina.loader";    permission java.lang.RuntimePermission "modifyThreadGroup";    permission java.net.SocketPermission "\*", "resolve";      permission java.lang.RuntimePermission "defineClassInPackage.java.util";    permission java.lang.RuntimePermission "defineClassInPackage.java.lang.reflect";    permission java.lang.RuntimePermission "defineClassInPackage.java.net";      permission java.io.FilePermission "${jboss.server.home.dir}${file.separator}lib${file.separator}\*", "read";    permission java.io.FilePermission "${jboss.server.home.dir}${file.separator}tmp${file.separator}-", "read";    permission java.io.FilePermission "${jboss.server.home.dir}${file.separator}tmp", "read";    permission java.io.FilePermission "${jboss.server.home.dir}${file.separator}deploy${file.separator}-", "read";    permission java.io.FilePermission "${jboss.server.home.dir}${file.separator}conf${file.separator}-", "read";    permission java.io.FilePermission "${jboss.server.home.dir}${file.separator}conf", "read";    permission java.io.FilePermission "${jboss.home.dir}${file.separator}lib${file.separator}-", "read";    permission java.io.FilePermission "${jboss.server.home.dir}", "read";    permission java.io.FilePermission "${jboss.server.home.dir}${file.separator}..", "read";    permission java.io.FilePermission "${jboss.home.dir}", "read";    permission java.io.FilePermission "${jboss.home.dir}${file.separator}..", "read";  }; |
|  |

To configure policy settings for JBoss 5.1

|  |  |  |
| --- | --- | --- |
| 1. JBoss 5.1 does supply a default policy file. However, this file allows all applications unlimited security privileges. The following code is a sample policy file that demonstrates the policies needed for BeanSpy to function properly. You can append this file to the existing policy file being used with your application server.  grant codeBase "file:${jboss.server.home.dir}/deploy/BeanSpy.ear" {    permission java.util.logging.LoggingPermission       "control";      permission javax.management.MBeanServerPermission    "findMBeanServer";    permission javax.management.MBeanServerPermission    "createMBeanServer";      permission javax.management.MBeanPermission          "\*", "queryMBeans";    permission javax.management.MBeanPermission          "\*", "getMBeanInfo";    permission javax.management.MBeanPermission          "\*", "getAttribute";        permission java.lang.RuntimePermission "getClassLoader";    permission java.lang.RuntimePermission "accessDeclaredMembers";    permission java.lang.RuntimePermission "accessClassInPackage.org.apache.catalina.connector";    permission java.lang.RuntimePermission "accessClassInPackage.org.apache.coyote";    permission java.lang.RuntimePermission "accessClassInPackage.org.apache.coyote.http11";    };   |  | | --- | |  |   To configure policy settings for JBoss 7, and Wildfly 8   |  | | --- | | 1. JBoss 7, and Widlfly 8 do not supply a default policy file. However, this file allows all applications unlimited security privileges. The following code is a sample policy file that demonstrates the policies needed for BeanSpy to function properly. You can append this file to the existing policy file being used with your application server.  **Important:** The file line should be modified to where the deployment is present in the file structure for the application server.  grant codeBase "file:${jboss.server.home.dir}/deployments/BeanSpy.ear" {    permission java.util.logging.LoggingPermission       "control";      permission javax.management.MBeanServerPermission    "findMBeanServer";    permission javax.management.MBeanServerPermission    "createMBeanServer";      permission javax.management.MBeanPermission          "\*", "queryMBeans";    permission javax.management.MBeanPermission          "\*", "getMBeanInfo";    permission javax.management.MBeanPermission          "\*", "getAttribute";        permission java.lang.RuntimePermission "getClassLoader";    permission java.lang.RuntimePermission "accessDeclaredMembers";    permission java.lang.RuntimePermission "accessClassInPackage.org.apache.catalina.connector";    permission java.lang.RuntimePermission "accessClassInPackage.org.apache.coyote";    permission java.lang.RuntimePermission "accessClassInPackage.org.apache.coyote.http11";    }; | |

## Enable Detailed Logging

By default, BeanSpy generates log messages only at deployment time to indicate whether it is able to connect to known JMX Stores. The following example shows the log output for a BeanSpy deployed on JBoss as it successfully connects to the JBoss and JDK JMX Stores.

13:01:55,814 INFO [EARDeployer] Init J2EE application: file:/C:/AppSrv/JBoss/jboss-4.2.1.GA/jboss-4.2.1.GA/server/default/deploy/BeanSpy.ear

13:01:56,041 INFO [TomcatDeployer] deploy, ctxPath=/BeanSpy, warUrl=.../tmp/deploy/tmp8870186149094117928BeanSpy.ear-contents/BeanSpy-exp.war/

13:01:56,337 ERROR [STDERR] Oct 13, 2010 1:01:56 PM com.microsoft.scx.log.MsLogger info

INFO: contextInitialized: connecting to JMX Stores

13:01:56,350 ERROR [STDERR] Oct 13, 2010 1:01:56 PM com.microsoft.scx.log.MsLogger info

INFO: Added JMX Store adapter com.microsoft.scx.jmx.JBossJMXAbstraction

13:01:57,765 ERROR [STDERR] Oct 13, 2010 1:01:57 PM com.microsoft.scx.log.MsLogger warning

WARNING: Failed to verify connection to com.microsoft.scx.jmx.TomcatJMXAbstraction JMX Store

13:01:57,770 ERROR [STDERR] Oct 13, 2010 1:01:57 PM com.microsoft.scx.log.MsLogger warning

WARNING: Failed to connect to com.microsoft.scx.jmx.WebSphereJMXAbstraction JMXStore

13:01:57,787 ERROR [STDERR] Oct 13, 2010 1:01:57 PM com.microsoft.scx.log.MsLogger info

INFO: Added JMX Store adapter com.microsoft.scx.jmx.JdkJMXAbstraction

13:01:57,839 INFO [EARDeployer] Started J2EE application: file:/C:/AppSrv/JBoss/jboss-4.2.1.GA/jboss-4.2.1.GA/server/default/deploy/BeanSpy.ear

To enable in JBoss 4.2

|  |
| --- |
| 1. Make a copy of the JDK log properties file: <JAVA\_HOME>\ \logging jre\lib.properties  2. Add the following line under JAVA\_OPTS:  -Djava.util.logging.config.file=<Path to copied file>\logging.properties  3. By default, nothing is logged to file and INFO and higher is logged to console. To change logging level, add a line for BeanSpy:  com.interopbridges.scx.level = FINEST  4. Adjust the console log level:  java.util.logging.ConsoleHandler.level = FINEST  5. Optionally, to log to file, add a file handler to the list of handlers: Optionally, to log to file, add a file handler to the list of handlers:  handlers = java.util.logging.ConsoleHandler, java.util.logging.FileHandler  6. Configure the file handler:  java.util.logging.FileHandler.pattern = <Full path and name of log file>  java.util.logging.FileHandler.limit = 50000  java.util.logging.FileHandler.count = 1  java.util.logging.FileHandler.formatter = java.util.logging.SimpleFormatter |

To enable in JBoss 5.1

|  |  |  |
| --- | --- | --- |
| 1. In the jboss-log4j.xml file, located in the <JBoss Home>\server\<Configuration>\conf\ directory, add a category element for BeanSpy with the desired log level (INFO, FINE, FINER, FINEST):  <category name="com.interopbridges.scx">  <priority value="FINEST"/>  </category>  2. Optionally, the logs can also be written to the console by changing the Threshold value for the console:  <appender name="CONSOLE" class="org.apache.log4j.ConsoleAppender">  <errorHandler class="org.jboss.logging.util.OnlyOnceErrorHandler"/>  <param name="Target" value="System.out"/>  <param name="Threshold" value="FINEST"/>    <layout class="org.apache.log4j.PatternLayout">  <!-- The default pattern: Date Priority [Category] Message\n -->  <param name="ConversionPattern" value="%d{ABSOLUTE} %-5p [%c{1}] %m%n"/>  </layout>  </appender>  To enable in JBoss 7, and Wildfly 8   |  | | --- | | In the standalone.xml file or domain.xml file, located in the server configuration directory: standalone: <JBoss Home>\standalone\configuration\.  domain: <JBoss Home>\domain\configuration\.    add a new logger with category attribute for BeanSpy with the desired log level (INFO, FINE, FINER, FINEST):  <logger category name="com.interopbridges.scx">  <priority value="FINEST"/>  </category> | |  | |

## Configuration Parameters

You can include configuration parameters in a BeanSpy query to control the attribute depth, count, size, and time. For example:

http://localhost:8080/BeanSpy/MBeans/JMXQuery=\*:\*&MaxSize=100&MaxDepth=10&MaxCount=100

The following table lists the configuration parameters that you can include in a query.

|  |  |  |
| --- | --- | --- |
| Parameter | Description | Default Value |
| MaxDepth | The maximum (or deepest) level of an XML structure for which to return MBean attributes. | 0 – returns all the elements of all the MBeans that satisfy the query. |
| MaxCount | The maximum number of items that will be processed for an MBean. | 5000 |
| MaxSize | The maximum size (in bytes) of the returned XML. The actual returned size, however, may be larger than the specified value because of processing primitive types and closing XML tags.  This value is overridden by the ABS\_MAX\_XML\_SIZE setting as described later in this section. | 2 MB |
| MaxTime | Limits the length of time (in seconds) that a method call can take to complete execution. When the limit is exceeded, the request will return an error to the caller stating that a timeout has occurred. | (none) |

### ABS\_MAX\_XML\_SIZE configuration file setting

You can specify that the maximum returned output size be limited to specified value regardless of the value specified by the MaxSize configuration parameter in a query. The ABS\_MAX\_XML\_SIZE setting in the resources.configuration.config file overrides the MaxSize parameter setting. The default value is 4MB.

## Sample BeanSpy Query Results

The following XML result is from a JBoss 5.1 application server and is truncated because of its size.

Query:

http://host1.contoso.com:8080/BeanSpy/MBeans?JMXQuery=JBoss.management.local:j2eeType=J2EEApplication,name=BeanSpy.ear,\*

Result:

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

- <MBeans version="7.1.1008.0">

- <MBean Name="org.jboss.management.j2ee.J2EEApplication" objectName="jboss.management.local:J2EEServer=Local,j2eeType=J2EEApplication,name=BeanSpy.ear">

- <Properties>

<Property Name="deploymentDescriptor" type="java.lang.String"><?xml version="1.0" encoding="UTF-8"?> <application 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/application\_1\_4.xsd" version="1.4"> <display-name>BeanSpy</display-name> <module> <web> <web-uri>BeanSpy.war</web-uri> <context-root>/BeanSpy</context-root> </web> </module> <security-role> <role-name>monitoring</role-name> </security-role> </application></Property>

<Property Name="eventProvider" type="java.lang.Boolean">false</Property>

<Property Name="statisticsProvider" type="java.lang.Boolean">false</Property>

<objectName type="java.lang.String">jboss.management.local:J2EEServer=Local,j2eeType=J2EEApplication,name=BeanSpy.ear</objectName>

- <objectNameElements type="objectName">

<Domain>jboss.management.local</Domain>

<J2EEServer>Local</J2EEServer>

<j2eeType>J2EEApplication</j2eeType>

<name>BeanSpy.ear</name>

</objectNameElements>

<Property Name="parent" type="java.lang.String">jboss.management.local:j2eeType=J2EEServer,name=Local</Property>

<Property Name="stateManageable" type="java.lang.Boolean">false</Property>

<Property Name="server" type="java.lang.String">unknown server name</Property>

- <Property Name="modules" type="[Ljava.lang.String;">

<Property Name="modules" index="0">jboss.management.local:J2EEApplication=BeanSpy.ear,J2EEServer=Local,j2eeType=WebModule,name=BeanSpy.war</Property>

</Property>

</Properties>

</MBean>

</MBeans>

# Appendix C: Creating and Importing Certificates

This appendix describes how to create a test certificate and import it into the trusted certificate store.

## Create a Test Certificate

If your application server is configured to use secure sockets layer (SSL), you should already have certificate configured for your application server whether or not you want to use Operations Manager to monitor the application server. Operation Manager requires that you specify the Fully Qualified Domain Name (FQDN), instead of the host name or localhost, for the CN field of the application server certificate. This is the only requirement for the application server to be monitored by Operations Manager using SSL. The following procedures are samples of how you can set up a test certificate for the JBoss application server:

To generate a key

|  |
| --- |
| 1. Run the following command at the command prompt:  $JAVA\_HOME$\bin\keytool -genkey -alias jboss -keyalg RSA -keystore jboss-x.x.x\server\<server\_of\_interest>\conf\my.keystore  2. Enter and re-enter secret as the keystore password.  3. Enter the FQDN of the application server for the first and last name, for example: host1.contoso.com  4. Enter values for the following prompts:   Organizational Unit   Name of organization   City or locality   Sate or province   Two-letter county code  5. Press Y to confirm the responses.  6. Enter and re-enter the password for JBoss, which must be the same as the keystore key (secret). |

To set up JBoss to use SSL with the generated key

|  |
| --- |
| 1. Configure the Connector element in the server.xml file, in the jboss-x.x.x\server\<server\_of\_interest>\deploy\jbossweb.sar directory, as shown here:  <!-- SSL/TLS Connector configuration using the admin devl guide keystore -->  <Connector port="8443" address="${jboss.bind.address}"  maxThreads="100" strategy="ms" maxHttpHeaderSize="8192"  emptySessionPath="true"  scheme="https" secure="true" clientAuth="false"  keystoreFile="${jboss.server.home.dir}/conf/my.keystore"  keystorePass="secret" sslProtocol = "TLS" /> |

To set up JBoss 7 Standalone to use SSL with the generated key



1. For standalone servers add an HTTPs listener using jboss-cli.sh or jboss-cli.bat connected to the management port located in ${jboss.home.dir}/bin/

/subsystem=web/connector=HTTPS:add(socket-binding=https,scheme=https,protocol=HTTP/1.1,secure=true)

1. After a successful configured HTTPs listener run the follow jboss-cli command to associate a keystore. *Change settings in following command as needed*

/subsystem=web/connector=HTTPS/ssl=configuration:add(name=https,certificate-key-file="${jboss.server.config.dir}/keystore.jks",password=secret, key-alias=jboss, cipher-suite=RSA)

To set up JBoss 7 Domain mode to use SSL with the generated key



1. For domain servers add an HTTPs listener using jboss-cli.sh or jboss-cli.bat connected to the management port located in ${jboss.home.dir}/bin/ . *Change the* *profile used as needed*

/profile=full/subsystem=web/connector=HTTPS:add(socket-binding=https,scheme=https,protocol=HTTP/1.1,secure=true)

1. After a successful configured HTTs listener added for a domain server run the following jboss-cli command to associate a keystore. *Change settings in the following command as needed*

/profile=full/subsystem=web/connector=HTTPS/ssl=configuration:add(name=https,certificate-key-file="${jboss.domain.config.dir}/keystore.jks",password=secret, key-alias=jboss, cipher-suite=RSA)

To export the certificate

|  |
| --- |
| 1. Export the certificate from the generated keystore by running command:  $JAVA\_HOME$\bin\keytool -export -alias jboss -keystore jboss-x.x.x\server\<server\_of\_interest>\conf\my.keystore -file .\jboss.cer  You can now import the certificate. Start JBoss by running jboss-x.x.x\bin\run -b 0.0.0.0 at the command prompt. Note that if you do not specify a binding address you will only be able to access JBoss using the localhost alias. Operations Manager queries JBoss using the FQDN. |

## Import a Certificate

The Operations Manager agent runs on the local Windows computer where the monitored JEE application server is running. JEE application servers running on UNIX and Linux are monitored by the Operations Manager management server. In order for the Operations Manager agent to communicate with the JEE application server using SSL, the agent must be able to trust the application server’s certificate. As long as the application server’s certificate is imported into the agent computer’s trusted certificate store, Operations Manager can monitor the application server using SSL.

To import a certificate into a computer’s trusted certificate store

|  |
| --- |
| 1. Start the Microsoft Management Console by running mmc.exe at the command prompt or the Run box.  2. On the File menu, click Add/Remove Snap-in, select Certificates, and click Add.  3. Select Computer account.  4. Click Next. Select Local computer.  5. On the Trusted Root Certification Authorities, select Certificates, right-click and select All tasks and Import.  6. Browse for the certificate file and click Next.  7. Select Place all certificates in the following store and select the Trusted Root Certification Authorities store.  8. Click Next and Finish. |