

Guide for System Center Monitoring Pack for WebLogic

Microsoft Corporation

Published: October 9, 2013

Send feedback about this document to [mpgfeed@microsoft.com](mailto:mpgfeed@microsoft.com). Please include the monitoring 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 monitoring 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.

© 2013 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 Monitoring Pack for WebLogic 5](#_Toc365536491)

[Guide History 5](#_Toc365536492)

[Supported Configurations 5](#_Toc365536493)

[Files Described by this Guide 6](#_Toc365536494)

[Monitoring Pack Purpose 6](#_Toc365536495)

[Monitoring Scenarios 7](#_Toc365536496)

[Levels of Monitoring 7](#_Toc365536497)

[Monitoring Scenarios 8](#_Toc365536498)

[Monitoring WebLogic Admin Servers and Managed Servers 10](#_Toc365536499)

[Custom Application Monitoring 10](#_Toc365536500)

[How Health Rolls Up 11](#_Toc365536501)

[Configuring the WebLogic Pack 11](#_Toc365536502)

[Import the Monitoring Packs 12](#_Toc365536503)

[Security Configuration 12](#_Toc365536504)

[Deploy BeanSpy 13](#_Toc365536505)

[Verify BeanSpy Deployment 14](#_Toc365536506)

[Additional BeanSpy Configurations 15](#_Toc365536507)

[Enable Deep Monitoring 15](#_Toc365536508)

[Enable Performance Threshold Monitors 15](#_Toc365536509)

[Best Practice: Create a Monitoring Pack for Customizations 17](#_Toc365536510)

[Links 17](#_Toc365536511)

[Appendix A: Monitoring Pack Contents 18](#_Toc365536512)

[Discoveries 18](#_Toc365536513)

[Monitors 19](#_Toc365536514)

[Views 19](#_Toc365536515)

[Rules 20](#_Toc365536516)

[Appendix B: BeanSpy Configurations 20](#_Toc365536517)

[Security Configurations 20](#_Toc365536518)

[Users and Roles 21](#_Toc365536519)

[Java Policy Settings 22](#_Toc365536520)

[Configuration Parameters 30](#_Toc365536521)

[ABS\_MAX\_XML\_SIZE configuration file setting 31](#_Toc365536522)

[Sample BeanSpy Query Result 31](#_Toc365536523)

[Appendix C: Creating and Importing Certificates 34](#_Toc365536524)

[Create a Test Certificate 34](#_Toc365536525)

[Import a Certificate 37](#_Toc365536526)

# Guide for System Center Monitoring Pack for WebLogic

This guide was written based on the 7.3.2135.0 version of the Monitoring Pack for WebLogic.

## 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. |
| June 28, 2013 | Added WebLogic 12cR1 to this guide |

## Supported Configurations

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

|  |  |  |
| --- | --- | --- |
| WebLogic Versions | Windows Operating Systems | UNIX and Linux Operating Systems |
| WebLogic 10g, Rel3  WebLogic 11g Rel1  WebLogic 12c Rel1 | Windows Server 2003 SP2  Windows Server 2003 R2 SP2  Windows Server 2008 SP2  Windows Server 2008 R2  Windows Server 2012 SP1 Windows Server 2012 R2 | CentOS:   * 5(x86/x64) * 6(x86/x64)   Debian Linux:   * 5(x86/x64) * 6(x86/x64) * 7(x86/x64)   Oracle Linux:   * 5(x86/x64) * 6(x86/x64)   Red Hat Enterprise Linux:  **** 4(x86/x64)  **** 5(x86/x64)  **** 6(x86/x64  Solaris:  **** 9 (SPARC)  **** 10(SPARC/x86)   * 11 (SPARC/x86) - Exception: does not support WebLogic 10g   SLES:  **** 9(x86)  **** 10 sp1(x86/x64)  **** 11(x86/x64)  Ubuntu Linux Server:   * 10.04(x86/x64) * 12.04(x86/x64) |

## Files Described by this Guide

The Monitoring Pack for WebLogic pertains to the following files:

 Microsoft.JEE.WebLogic.10gR3.mp

 Microsoft.JEE.WebLogic.11gR1.mp

* Microsoft.JEE.WebLogic.12cR1.mp

 Microsoft.JEE.WebLogic.Library.mp

 Microsoft.JEE.Templates.Library.mpb

 Microsoft.JEE.Library.mpb

# Monitoring Pack Purpose

The System Center Monitoring Pack for WebLogic 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](#zae5daf2d74f0437793f50c4412d77d71)

 [How Health Rolls Up](#zabecf3a42efa42b18eda24ed92bad99a)

For details on the discoveries, rules, monitors, and views contained in this monitoring pack, see [Appendix A: Monitoring Pack Contents](#zade704badef74b349e642f12b5e1bffd).

# Monitoring Scenarios

After the monitoring packs for the JEE application servers are imported, the instances of WebLogic 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 WebLogic, all application server instances are discovered whether they are running or not. WebLogic sample domains are not automatically discovered.

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

1. In the Operations console, click Monitoring.

2. Expand Application Monitoring, then Java Monitoring, then WebLogic Application Servers, and select the monitoring folder of interest.

## Levels of Monitoring

The Monitoring Pack for WebLogic 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 Monitoring Pack for WebLogic 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 monitoring 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 monitoring 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, so the IT administrator can determine if the application is running, as seen by 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](#z9a7de6d2e08341b0832736abc44744ca).

## Monitoring Scenarios

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

|  |  |  |  |
| --- | --- | --- | --- |
| Monitoring scenario | Monitoring Folder | Description | Associated monitors |
| Application Server Availability | Servers | 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.  By default, application server availability process monitor for WebLogic is not enabled. | Process availability health unit monitor for WebLogic 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 WebLogic application servers. | Application availability health rollup |
| Deep availability health | Deep monitored servers | 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 WebLogic 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 Monitoring" and "JEE Application Performance Monitoring" monitoring 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 |

### Monitoring WebLogic Admin Servers and Managed Servers

Monitoring WebLogic admin servers are not supported because a WebLogic admin server process cannot be uniquely identified using the command line options for that process.

Monitoring WebLogic managed servers is supported only if the process was started with the WebLogic node manager, otherwise the process cannot be uniquely identified using the command line options for that process. Therefore, the process monitor for WebLogic managed servers is disabled by default. If the WebLogic managed server was started with WebLogic node manager, then you can set the override to enable the monitor.

## Custom Application Monitoring

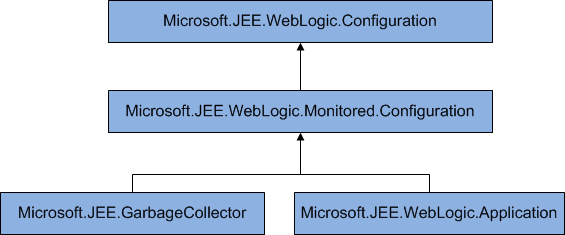
This "JEE Application Availability Monitoring" and "JEE Application Performance Monitoring" monitoring pack templates enable you to monitor information exposed through MBeans. To get the best user experience, the Operations Manager console must have HTTP or HTTPS access to the application server that 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 Monitoring or the JEE Application Performance Monitoring 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 objects roll up in this monitoring management pack.



# Configuring the WebLogic Pack

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

 [Import the Monitoring 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 Monitoring Pack for Customizations](#z10)

## Import the Monitoring Packs

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

 Microsoft.JEE.WebLogic.Library.mp

 Microsoft.JEE.Templates.Library.mpb

 Microsoft.JEE.Library.mpb

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

 Microsoft.JEE.WebLogic.10gR3.mp

 Microsoft.JEE.WebLogic.11gR1.mp

* Microsoft.JEE.WebLogic.12cR1.mp

For information on how to import a monitoring 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 monitoring 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 in the Microsoft.JEE.Library.mpb, and is installed into 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 to an application server

|  |
| --- |
| 1. In the Operations console, click Monitoring.  2. In the Monitoring workspace, under Application Monitoring, Java Monitoring, and WebLogic Application Servers, click the Application Servers for which you want to install BeanSpy.  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 WebLogic 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.

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 WebLogic. Adjust the host name and port as required.

https://host1.contoso.com:7001/BeanSpy/MBeans?JMXQuery=com.bea:Type=ApplicationRuntime,Name=BeanSpy,\*

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 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. If the query was not successful, check the following common causes for failures:

 BeanSpy is not deployed.

 BeanSpy is not started.

 A firewall is blocking the port.

 Invalid BeanSpy query syntax.

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

## Additional BeanSpy Configurations

See Appendix B: BeanSpy 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 for which you enabled deep monitoring 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 Application Monitoring, then Java Monitoring, then WebLogic Application Servers, and select the Performance folder.  3. Right-click one of the performance counters to be configured, and select Show or edit rule 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 Monitoring Pack for Customizations

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

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

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

The Monitoring Pack for WebLogic provides the object types described in the following lists. All objects are supported by the 10gR3, 11gR1, and 12cR1 versions of the WebLogic Application Server.

## Discoveries

The following discoveries are provided for WebLogic Application Server:

 Windows installations

 Domains

 Windows managed servers

 Admin servers

 Windows servers

 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 WebLogic application server configuration.

 Applications

BeanSpy query: com.bea:Type=AppDeployment,\*

 Garbage collection

## Monitors

The following monitors are provided for applications:

 Application active version state availability

 Health state availability

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

 Process availability health unit monitor for WebLogic 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:

 State, Application Name, Object Name, Path

The following views are provided for Windows installations:

 Install Path, Version, Status, Path

The following views are provided for domains:

 Name, Version, Path on Disk, Path

The following views are provided monitored attributes:

 Id, HostName, OSName, OSVersion, OSArchitecture, JVMName, JVMVersion, JVMVendorName, JVMBuildVersion, JVMClassPath, JVMLibraryPath, JVMStartupOption, JVMJavaInstallDirectory, JVMHeapSize, JEEServerName,

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

 Servers:

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

 Deep Monitored Servers:

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

 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

 Users and Roles

 Java Policy Settings

 Enable Verbose Logging

 Configurable Parameters

 Sample BeanSpy Query Results

For information about deploying BeanSpy, see [Configuring the WebLogic Pack](#z5e203e615843481fae75af6198f5a9f5).

## 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](#z5c36e89ff0d34eb480c3008e5bd5d5ed) 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 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.

To create a user and associate with a role

|  |
| --- |
| 1. In the WebLogic console, click the Security Realms in the left-hand Domain Structure panel. This should display a panel that has a list of realms. The default realm is named myrealm. This is the realm that the BeanSpy uses by default.  2. Click the link for myrealm, then Users and Groups, and then New.  3. Fill in the requisite fields of user information and click OK when finished to return to the list of user names.  4. Click the newly created user name, and select the Groups tab.  5. Under the Parent Groups section, in the Available list of users, select Monitors and click the right arrow to move the newly created user name into the Chosen column and click Save. Monitors role is required to query MBeans, while Operators role is required to invoke methods on MBeans. |

## 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 WebLogic Managed Server

|  |
| --- |
| 1. The following 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:${user.dir}/servers/${weblogic.Name}/tmp/\_WL\_user/BeanSpy/-" {      permission java.lang.management.ManagementPermission "monitor";      permission javax.management.MBeanServerPermission    "createMBeanServer";      permission javax.management.MBeanPermission "\*", "getAttribute";      permission javax.management.MBeanPermission "\*", "getMBeanInfo";      permission javax.management.MBeanPermission "\*", "queryMBeans";        permission java.lang.RuntimePermission "accessDeclaredMembers";      permission java.lang.RuntimePermission "getProtectionDomain";      permission java.lang.RuntimePermission "getClassLoader";      permission java.util.PropertyPermission "\*", "read,write";        permission java.lang.RuntimePermission "accessClassInPackage.sun.misc";      permission java.lang.RuntimePermission "accessClassInPackage.sun.util";      permission java.lang.RuntimePermission "accessClassInPackage.sun.util.logging.resources";      permission java.lang.RuntimePermission "accessClassInPackage.sun.util.calendar";      permission java.lang.RuntimePermission "accessClassInPackage.sun.management";      permission java.lang.RuntimePermission "accessClassInPackage.sun.net.www.content.content";      permission java.lang.RuntimePermission "setContextClassLoader";        permission java.lang.reflect.ReflectPermission "suppressAccessChecks";        permission java.security.SecurityPermission "getPolicy";        permission java.io.FilePermission "${java.ext.dirs}/\*", "read";      permission java.io.FilePermission "${java.home}/../lib/-", "read";      permission java.io.FilePermission "${java.home}/lib/-", "read";      permission java.io.FilePermission "${java.home}/classes", "read";      permission java.io.FilePermission "${bea.home}/wlserver\_10.3/server/lib/-", "read";      permission java.io.FilePermission "${bea.home}/wlserver\_10.3/server/\*", "read";      permission java.io.FilePermission "${bea.home}/wlserver\_10.3/server", "read";      permission java.io.FilePermission "${bea.home}/utils/-", "read";      permission java.io.FilePermission "${bea.home}/wlserver\_10.3/common/-", "read";      permission java.io.FilePermission "${bea.home}/modules/-", "read";      permission java.io.FilePermission "${patch.path}/../-", "read";        permission java.lang.RuntimePermission "accessClassInPackage.sun.reflect.generics.reflectiveObjects";      permission java.lang.RuntimePermission "accessClassInPackage.sun.net.www.content.text";      permission java.io.FilePermission "${bea.home}/user\_projects/-", "read";  }; |

To configure policy settings for WebLogic 10g Admin Server

|  |
| --- |
| 1. The following is a sample policy file that demonstrates the policies needed for the BeanSpy to function properly. You can append this file to the existing policy file being used with your application server.  grant codebase "file:${oim.domain}/Servers/AdminServer/tmp/\_WL\_user/BeanSpy/-" {      permission java.lang.management.ManagementPermission "monitor";      permission javax.management.MBeanServerPermission    "createMBeanServer";      permission javax.management.MBeanPermission "\*", "getAttribute";      permission javax.management.MBeanPermission "\*", "getMBeanInfo";      permission javax.management.MBeanPermission "\*", "queryMBeans";      permission javax.management.MBeanPermission "\*", "registerMBean";        permission java.lang.RuntimePermission "accessDeclaredMembers";      permission java.lang.RuntimePermission "getProtectionDomain";      permission java.lang.RuntimePermission "getClassLoader";      permission java.util.PropertyPermission "\*", "read,write";      permission java.lang.RuntimePermission "accessClassInPackage.sun.misc";      permission java.lang.RuntimePermission "accessClassInPackage.sun.util";      permission java.lang.RuntimePermission "accessClassInPackage.sun.util.logging.resources";      permission java.lang.RuntimePermission "accessClassInPackage.sun.util.calendar";      permission java.lang.RuntimePermission "accessClassInPackage.sun.management";      permission java.lang.RuntimePermission "accessClassInPackage.sun.net.www.content.content";      permission java.lang.RuntimePermission "accessClassInPackage.sun.net.www.content.text";      permission java.lang.RuntimePermission "setContextClassLoader";        permission java.lang.reflect.ReflectPermission "suppressAccessChecks";      permission java.net.SocketPermission "\*", "resolve";        permission java.security.SecurityPermission "getPolicy";        permission java.io.FilePermission "${java.ext.dirs}${/}\*", "read";      permission java.io.FilePermission "${java.home}${/}..${/}lib${/}-", "read";      permission java.io.FilePermission "${java.home}${/}lib${/}-", "read";      permission java.io.FilePermission "${java.home}${/}classes", "read";      permission java.io.FilePermission "${bea.home}${/}wlserver\_10.3${/}server${/}lib${/}-", "read";      permission java.io.FilePermission "${bea.home}${/}wlserver\_10.3${/}server${/}\*", "read";      permission java.io.FilePermission "${bea.home}${/}wlserver\_10.3${/}server", "read";      permission java.io.FilePermission "${bea.home}${/}utils${/}-", "read";      permission java.io.FilePermission "${bea.home}${/}wlserver\_10.3${/}common${/}-", "read";      permission java.io.FilePermission "${bea.home}${/}modules${/}-", "read";      permission java.io.FilePermission "${patch.path}${/}..${/}-", "read";      permission java.io.FilePermission "${oim.domain}${/}Servers${/}AdminServer${/}tmp${/}\_WL\_user${/}BeanSpy${/}-", "read";      permission java.io.FilePermission "${oim.domain}${/}", "read";  }; |

To configure policy settings for WebLogic 11g Admin Server

|  |  |
| --- | --- |
| 1. The following is a sample policy file that demonstrates the policies needed for the BeanSpy to function properly. You can append this file to the existing policy file being used with your application server.  grant codebase "file:${oim.domain}/Servers/AdminServer/tmp/\_WL\_user/BeanSpy/-" {      permission java.lang.management.ManagementPermission "monitor";      permission javax.management.MBeanServerPermission    "createMBeanServer";      permission javax.management.MBeanPermission "\*", "getAttribute";      permission javax.management.MBeanPermission "\*", "getMBeanInfo";      permission javax.management.MBeanPermission "\*", "queryMBeans";        permission java.lang.RuntimePermission "accessDeclaredMembers";      permission java.lang.RuntimePermission "getProtectionDomain";      permission java.lang.RuntimePermission "getClassLoader";      permission java.util.PropertyPermission "\*", "read,write";      permission java.lang.RuntimePermission "accessClassInPackage.sun.misc";      permission java.lang.RuntimePermission "accessClassInPackage.sun.util";      permission java.lang.RuntimePermission "accessClassInPackage.sun.util.logging.resources";      permission java.lang.RuntimePermission "accessClassInPackage.sun.util.calendar";      permission java.lang.RuntimePermission "accessClassInPackage.sun.management";      permission java.lang.RuntimePermission "accessClassInPackage.sun.net.www.content.content";      permission java.lang.RuntimePermission "setContextClassLoader";        permission java.lang.reflect.ReflectPermission "suppressAccessChecks";        permission java.security.SecurityPermission "getPolicy";        permission java.io.FilePermission "${java.ext.dirs}/\*", "read";      permission java.io.FilePermission "${java.home}/../lib/-", "read";      permission java.io.FilePermission "${java.home}/lib/-", "read";      permission java.io.FilePermission "${java.home}/classes", "read";      permission java.io.FilePermission "${bea.home}/wlserver\_10.3/server/lib/-", "read";      permission java.io.FilePermission "${bea.home}/wlserver\_10.3/server/\*", "read";      permission java.io.FilePermission "${bea.home}/wlserver\_10.3/server", "read";      permission java.io.FilePermission "${bea.home}/utils/-", "read";      permission java.io.FilePermission "${bea.home}/wlserver\_10.3/common/-", "read";      permission java.io.FilePermission "${bea.home}/modules/-", "read";      permission java.io.FilePermission "${patch.path}/../-", "read";  };   * To configure policy settings for WebLogic 12c Admin Server  |  | | --- | | 1. The following is a sample policy file that demonstrates the policies needed for the BeanSpy to function properly. You can append this file to the existing policy file being used with your application server.  grant codebase "file:${oim.domain}/Servers/AdminServer/tmp/\_WL\_user/BeanSpy/-" {      permission java.lang.management.ManagementPermission "monitor";      permission javax.management.MBeanServerPermission    "createMBeanServer";      permission javax.management.MBeanPermission "\*", "getAttribute";      permission javax.management.MBeanPermission "\*", "getMBeanInfo";      permission javax.management.MBeanPermission "\*", "queryMBeans";        permission java.lang.RuntimePermission "accessDeclaredMembers";      permission java.lang.RuntimePermission "getProtectionDomain";      permission java.lang.RuntimePermission "getClassLoader";      permission java.util.PropertyPermission "\*", "read,write";      permission java.lang.RuntimePermission "accessClassInPackage.sun.misc";      permission java.lang.RuntimePermission "accessClassInPackage.sun.util";      permission java.lang.RuntimePermission "accessClassInPackage.sun.util.logging.resources";      permission java.lang.RuntimePermission "accessClassInPackage.sun.util.calendar";      permission java.lang.RuntimePermission "accessClassInPackage.sun.management";      permission java.lang.RuntimePermission "accessClassInPackage.sun.net.www.content.content";      permission java.lang.RuntimePermission "setContextClassLoader";        permission java.lang.reflect.ReflectPermission "suppressAccessChecks";        permission java.security.SecurityPermission "getPolicy";        permission java.io.FilePermission "${java.ext.dirs}/\*", "read";      permission java.io.FilePermission "${java.home}/../lib/-", "read";      permission java.io.FilePermission "${java.home}/lib/-", "read";      permission java.io.FilePermission "${java.home}/classes", "read";      permission java.io.FilePermission "${bea.home}/wlserver\_12.1/server/lib/-", "read";      permission java.io.FilePermission "${bea.home}/wlserver\_12.1/server/\*", "read";      permission java.io.FilePermission "${bea.home}/wlserver\_12.1/server", "read";      permission java.io.FilePermission "${bea.home}/utils/-", "read";      permission java.io.FilePermission "${bea.home}/wlserver\_12.1/common/-", "read";      permission java.io.FilePermission "${bea.home}/modules/-", "read";      permission java.io.FilePermission "${patch.path}/../-", "read";  }; | |

## 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 Result

The following XML result is from a WebLogic 12c application server and is truncated because of its size.

Query:

http://host1.contoso.com:7001/BeanSpy/MBeans?JMXQuery=com.bea:Type=ApplicationRuntime,Name=BeanSpy,\*

Result:

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

-<MBeans version="7.4.4526.0">

-<MBean ObjectName="com.bea:Name=BeanSpy,ServerRuntime=AdminServer, Type=ApplicationRuntime" Name="weblogic.application.internal.ApplicationRuntimeMBeanImpl">

-<Properties>

<Property Name="WseeRuntimes" type="[Ljavax.management.ObjectName;"/>

<Property Name="ApplicationName" type="java.lang.String">BeanSpy</Property>

-<Property Name="Parent" type="javax.management.ObjectName">

<Property Name="canonicalKeyPropertyListString" type="java.lang.String">Name=AdminServer,Type=ServerRuntime</Property>

<Property Name="canonicalName" type="java.lang.String">com.bea:Name=AdminServer,Type=ServerRuntime</Property>

<Property Name="domain" type="java.lang.String">com.bea</Property>

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

<Property Name="keyPropertyList" type="java.util.Hashtable">{Name=AdminServer, Type=ServerRuntime}</Property>

<Property Name="keyPropertyListString" type="java.lang.String">Name=AdminServer,Type=ServerRuntime</Property>

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

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

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

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

The following XML result is from a WebLogic 11g application server and is truncated because of its size.

Query:

http://host1.contoso.com:7001/BeanSpy/MBeans?JMXQuery=com.bea:Type=ApplicationRuntime,Name=BeanSpy,\*

Result:

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

- <MBeans version="7.1.1010.0">

- <MBean Name="weblogic.j2ee.J2EEApplicationRuntimeMBeanImpl" objectName="com.bea:Name=BeanSpy,ServerRuntime=AdminServer,Type=ApplicationRuntime">

- <Properties>

  <Property Name="WseeRuntimes" type="[Ljavax.management.ObjectName;" />

  <Property Name="ApplicationName" type="java.lang.String">BeanSpy</Property>

- <Property Name="Parent" type="javax.management.ObjectName">

  <Property Name="canonicalKeyPropertyListString" type="java.lang.String">Name=AdminServer,Type=ServerRuntime</Property>

  <Property Name="canonicalName" type="java.lang.String">com.bea:Name=AdminServer,Type=ServerRuntime</Property>

  <Property Name="domain" type="java.lang.String">com.bea</Property>

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

- <Property Name="keyPropertyList" type="java.util.Hashtable">

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

  </Property>

  <Property Name="keyPropertyListString" type="java.lang.String">Name=AdminServer,Type=ServerRuntime</Property>

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

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

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

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

  </Property>

  <Property Name="EAR" type="java.lang.Boolean">true</Property>

  <Property Name="Type" type="java.lang.String">ApplicationRuntime</Property>

- <Property Name="WorkManagerRuntimes" type="[Ljavax.management.ObjectName;">

- <Property Name="WorkManagerRuntimes" index="0">

  <Property Name="canonicalKeyPropertyListString" type="java.lang.String">ApplicationRuntime=BeanSpy,Name=default,ServerRuntime=AdminServer,Type=WorkManagerRuntime</Property>

  <Property Name="canonicalName" type="java.lang.String">com.bea:ApplicationRuntime=BeanSpy,Name=default,ServerRuntime=AdminServer,Type=WorkManagerRuntime</Property>

  <Property Name="domain" type="java.lang.String">com.bea</Property>

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

- <Property Name="keyPropertyList" type="java.util.Hashtable">

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

  </Property>

# 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

WebLogic automatically generates a keystore when installed. The location of the default trust keystore for WebLogic 10gR3 and 11gR1 is $WEBLOGIC\_HOME\wlserver\_10.3\server\lib\DemoTrust.jks. The location of the default trust keystore for WebLogic 12cR1 is

$WEBLOGIC\_HOME\wlserver\_12.1\server\lib\DemoTrust.jks

However, this keystore is generated using the computer name, not the FQDN, so you will need to create a new keystore.

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 WebLogic application server:

To generate keys

|  |
| --- |
| 1. Create a folder named Certs in the following directory and open a command prompt at the Certs folder.  WebLogic 10/11: $WEBLOGIC\_HOME\wlserver\_10.3\server\lib\  WebLogic 12: $WEBLOGIC\_HOME\wlserver\_12.1\server\lib\  2. Generate a key for the WebLogic Admin Server:  a. Run the following command at the command prompt:  $JAVA\_HOME$\bin\keytool -genkey -alias adminserver -keyalg RSA -keystore .\myidentity.keystore  b. Enter and re-enter secret as the keystore password.  c. Enter the FQDN of the application server for the first and last name, for example: host1.contoso.com  d. Enter values for the following prompts:   Organizational Unit   Name of organization   City or locality   Sate or province   Two-letter county code  e. Press Y to confirm the responses.  f. Enter and re-enter the password for WebLogic Admin Server, which must be the same as the keystore key (secret).  3. Generate a key for the WebLogic Managed Server:  a. Run the following command at the command prompt:  $JAVA\_HOME$\bin\keytool -genkey -alias managedserver -keyalg RSA -keystore .\myidentity.keystore  b. Enter and re-enter secret as the keystore password.  c. Enter the FQDN of the application server for the first and last name, for example: host1.contoso.com  d. Enter values for the following prompts:   Organizational Unit   Name of organization   City or locality   Sate or province   Two-letter county code  e. Press Y to confirm the responses.  f. Enter and re-enter the password for WebLogic Managed Server, which must be the same as the keystore key (secret).  4. Generate a key for the WebLogic Node Manager:  a. Run the following command at the command prompt:  $JAVA\_HOME$\bin\keytool -genkey -alias nodemanager -keyalg RSA -keystore .\myidentity.keystore  b. Enter and re-enter secret as the keystore password.  c. Enter the FQDN of the application server for the first and last name, for example: host1.contoso.com  d. Enter values for the following prompts:   Organizational Unit   Name of organization   City or locality   Sate or province   Two-letter county code  e. Press Y to confirm the responses.  f. Enter and re-enter the password for WebLogic Node Manager, which must be the same as the keystore key (secret).  5. Generate WebLogic trust store by running the following commands in the command prompt:   keytool -export -alias adminserver -keystore myidentity.keystore -storepass secret -file .\adminserver.cer   keytool -export -alias managedserver -keystore myidentity.keystore -storepass secret -file .\managedserver.cer   keytool -export -alias nodemanager -keystore myidentity.keystore -storepass secret -file .\nodemanager.cer   keytool –import -alias adminserver -file .\adminserver.cer -keystore .\mytrust.keystore -storepass secret   keytool –import -alias managedserver -file .\managedserver.cer -keystore .\mytrust.keystore -storepass secret   keytool –import -alias nodemanager -file .\nodemanager.cer -keystore .\mytrust.keystore -storepass secret  6. Copy $WEBLOGIC\_HOME\wlserver\_10.3\server\lib\Certs\myidentity.keystore and $WEBLOGIC\_HOME\wlserver\_10.3\server\lib\Certs\mytrust.keystore to $WEBLOGIC\_HOME\wlserver\_10.3\common\nodemanager and edit the following block of text in nodemanager.properties file to have the following settings applicable to your configurations.  DomainsFile=C\:\\APPSER~1\\Oracle\\MIDDLE~1\\WLSERV~1.3\\common\\NODEMA~1\\nodemanager.domains  LogLimit=0  PropertiesVersion=10.3  javaHome=C\:\\Java64\\JDK16~1.0\_2  AuthenticationEnabled=true  NodeManagerHome=C\:\\APPSER~1\\Oracle\\MIDDLE~1\\WLSERV~1.3\\common\\NODEMA~1  JavaHome=C\:\\Java64\\JDK16~1.0\_2\\jre  LogLevel=FINE  DomainsFileEnabled=true  StartScriptName=startWebLogic.cmd  ListenAddress=host1.contoso.com  NativeVersionEnabled=true  ListenPort=5556  LogToStderr=true  SecureListener=true  LogCount=1  StopScriptEnabled=false  QuitEnabled=false  LogAppend=true  StateCheckInterval=500  CrashRecoveryEnabled=false  StartScriptEnabled=false  LogFile=C\:\\APPSER~1\\Oracle\\MIDDLE~1\\WLSERV~1.3\\common\\NODEMA~1\\nodemanager.log  LogFormatter=weblogic.nodemanager.server.LogFormatter  ListenBacklog=50  KeyStores=CustomIdentityAndCustomTrust  CustomIdentityAlias=nodemanager  CustomIdentityKeyStoreFileName=myidentity.keystore  CustomIdentityKeyStorePassPhrase=secret  CustomIdentityKeyStoreType=jks  CustomIdentityPrivateKeyPassPhrase=secret  CustomTrustKeyPassPhrase=secret  CustomTrustKeyStoreFileName=mytrust.keystore  CustomTrustKeyStorePassPhrase=  CustomTrustKeyStoreType=jks |

## 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. |