BEA WebLogic Server®

Node Manager Administrator's Guide

Version 10.0
Revised: March 30, 2007

Contents

3. General Node Manager Configuration

4. Configuring Java Node Manager

5. Configuring Script Node Manager

6. Using Node Manager

CHAPTER 1

Introduction and Roadmap

The following sections describe the contents and organization of this guide—*Node Manager Administrator's Guide*.

Document Scope and Audience

This document describes how to configure and use Node Manager to control and manage servers within a WebLogic Server environment.

This document is a resource for system administrators and operators responsible for using Node Manager. It is relevant to all phases of a software project, from development through test and production phases.

It is assumed that the reader is familiar with Java Platform, Enterprise Edition (Java EE) and Web technologies, object-oriented programming techniques, and the Java programming language.

Guide to This Document

The document is organized as follows:

- This chapter, "Introduction and Roadmap," describes the scope of the guide and lists related documentation.
- Chapter 2, "Node Manager Overview," provides a general description of Node Manager and describes how it works within a WebLogic Server domain.It also provides detailed description of the configuration and log files used by Node Manager.
- Chapter 3, "General Node Manager Configuration," describes configuration procedures that are applicable to the both the Java and scripted versions of Node Manager.
- Chapter 4, "Configuring Java Node Manager," describes the configuration procedures for the Java version of Node Manager.
- Chapter 5, "Configuring Script Node Manager," describes the configuration procedures for the scripted version of Node Manager.
- Chapter 6, "Using Node Manager," provides procedures for starting Node Manager and servers. This chapter also includes recommendations for starting servers to take advantage of WebLogic Server's failover and migration features.

Related Documentation

- *Creating WebLogic Domains Using the Configuration Wizard*
- *Understanding Domain Configuration*
- *Administration Console Online Help*

New and Changed Features for Managing Server Life Cycle

For information about the new and changed features in WebLogic Server 10.0, see What's New in WebLogic Server 10.0.

CHAPTER 2

Node Manager Overview

This chapter provides an introduction to Node Manager. The following topics are covered:

Introduction

Server instances in a WebLogic Server production environment are often distributed across multiple domains, machines, and geographic locations. Node Manager is a WebLogic Server utility that enables you to start, shut down, and restart Administration Server and Managed Server instances from a remote location. Although Node Manager is optional, it is recommended if your WebLogic Server environment hosts applications with high availability requirements.

A Node Manager process is not associated with a specific WebLogic domain but with a machine. You can use the same Node Manager process to control server instances in any WebLogic Server domain, as long as the server instances reside on the same machine as the Node Manager process.

Node Manager must run on each computer that hosts WebLogic Server instances -- whether Administration Server or Managed Server -- that you want to control with Node Manager.

Node Manager Versions

WebLogic Server provides two versions of Node Manager, Java-based and script-based, with similar functionality. However, each version has different configuration and security considerations.

Java-based Node Manager

Java-based Node Manager runs within a Java Virtual Machine (JVM) process. It is recommended that you run it as a Windows service on Windows platforms and as an operating service on UNIX platforms, allowing it to restart automatically when the system is rebooted.

BEA provides native Node Manager libraries for Windows, Solaris, HP UX, Linux on Intel, Linux on Z-Series, and AIX operating systems.

Note: Node Manager is not supported on Open VMS, OS/390, AS400, UnixWare, or Tru64 UNIX.

This version of Node Manager determines its configuration from the nodemanager.properties file. See "Reviewing nodemanager.properties" on page 4-3.

Java-based Node Manager provides more security than the script-based version. See "Configuring Java-based Node Manager Security" on page 4-2.

Script-based Node Manager

For UNIX and Linux systems, WebLogic Server provides a script-based version of Node Manager. This script is based on UNIX shell scripts, but uses SSH for increased security. SSH uses user-id based security.

For information on configuring the script version of Node Manager, see Chapter 5, "Configuring Script Node Manager."

This version does not provide as much security as the Java-based version. However, the advantage of the script-based Node Manager is that it can remotely manage servers over a network that has been configured to use SSH. No additional server installation is required. The scripts merely have to be copied to the remote machine.

Note: It is recommended that you run script-based Node Manager as an operating system service, which allows it to restart automatically when the system is rebooted.

Determining Which Node Manager Version to Use

Which version of Node Manager to use depends on the requirements of your WebLogic Server environment. The following considerations can help you decide which version is ideal for your environment:

- Automatic Server Migration is only supported using the scripted version of Node Manager. To incorporate Automatic Server Migration, you must use the scripted version of Node manager.
- If you are installing WebLogic Server on a Windows system, you must use the Java version of Node Manager. The scripted version of Node Manager is not supported on Windows.
- In order to use db-less leasing (consensus leasing) you may see faster performance when using the Java version of Node Manager.
- The script based Node Manager requires a much simpler security configuration than the Java version. RSH and SSH are general easier to configure than SSL which is the method of security used by the Java version of Node Manager. The script version of Node Manager also required a smaller footprint than the Java version
- The Java version of Node Manager can be used in conjunction with `inetd` on supported UNIX systems. `inetd` allows Node Manager to be automatically restarted on upon receiving a request on the configured port.

Accessing Node Manager

A Node Manager client can be local or remote to the Node Managers with which it communicates. You access either version of Node Manager—the Java version or the script-based (SSH) version—from the following clients: (In addition, an SSH client in the form of a shell command template is provided for use with the script-based Node Manager.)

- Administration Server
  - Administration Console, from the Environments>Machines>Configuration>Node Manager page.
  - JMX utilities

    For example, you can create JMX utilities that talk to the admin server and perform operations on the ServerLifeCycleRuntimeMBean which in turn uses Node Manager internal to perform operations on the For more information about JMX, see *Developing Custom Management Utilities with JMX*.

- WLST commands and scripts—WLST offline serves as a Node Manager command-line interface that can run in the absence of a running Administration Server. You can use WLST commands to start, stop, and monitor a server instance without connecting to an Administration Server. Starting the Administration Server is the main purpose of the stand-alone client. However, you can also use it to:
  - Stop a server instance that was started by Node Manager.
  - Start a Managed Server.
  - Access the contents of a Node Manager log file.
  - Obtain server status for a server that was started with Node Manager.
  - Retrieve the contents of server output log.

For more information on using WLST and Node Manager to control servers, see "Using Node Manager to Control Servers."

What You Can Do with Node Manager

The following sections describe basic Node Manager functionality.

Start, Shut Down, and Restart an Administration Server

Using the WebLogic Scripting Tool (or SSH client for Script-based Node Manager only), you connect to the Node Manager process on the machine that hosts the Administration Server and issue commands to start, shut down, or restart an Administrative Server. The relationship of an Administration Server to Node Manager varies for different scenarios.

- An Administration Server can be under Node Manager control—You can start it, monitor it, and restart it using Node Manager.
- An Administration Server can be a Node Manager client—When you start or stop Managed Servers from the Administration Console, you are accessing Node Manager via the Administration Server.
- An Administration Server supports the process of starting up a Managed Server with Node Manager—When you start a Managed Server with Node Manager, the Managed Server contacts the Administration Server to obtain outstanding configuration updates.

Start, Shut Down, Suspend, and Restart Managed Servers

From the WebLogic Server Scripting Tool (WLST) command line or scripts, you can issue commands to Node Manager to start, shut down, suspend, and restart Managed Server instances and clusters.

Node Manager can restart a Managed Server after failure even when the Administration Server is unavailable if Managed Server Independence (MSI) mode is enabled for that Managed Server instance. This is enabled by default.

Note: Node Manager cannot start a Managed Server for the first time in MSI mode, because at the Administration Server for the domain must be available so the Managed Server can obtain its configuration settings.

Note: Node Manager uses the same command arguments that you supply when starting a Managed Server with a script or at the command line. For information about startup arguments, see "weblogic.Server Command-Line Reference" in *WebLogic Server Command Reference*.

Restart Administration and Managed Servers

If a server instance that was started using Node Manager fails, Node Manager automatically restarts it.

Note: Node Manager can only restart a server that was started via Node Manager.

The restart feature is configurable. Node Manager's default behavior is to:

- Automatically restart server instances under its control that fail. You can disable this feature.
- Restart failed server instances no more than a specific number of times. You define the number of restarts by setting the `RestartMax` property in the Node Manager `startup.properties` file.

If Node Manager fails or is explicitly shut down, upon restart, it determines the server instances that were under its control when it exited. Node Manager can restart any failed server instances as necessary.

Note: It is advisable to run Node Manager as an operating system service, so that it restarts automatically if its host machine is restarted.

Monitor Servers and View Log Data

Node Manager creates a log file for the Node Manager process and a log file of server output for each server instance it controls. You can view these log files, as well as log files for a server instance using the Administration Console or WLST commands.

How Node Manager Works in the WebLogic Server Environment

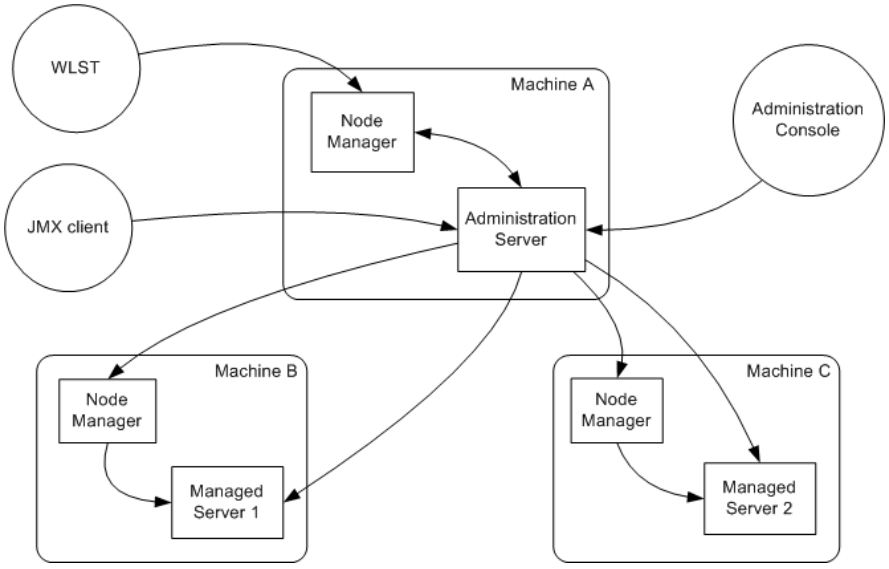
The following sections provide a "big picture" diagram of Node Manager's role in the WebLogic Server environment, as well as illustrations and descriptions of the processes Node Manager uses to communicate with servers:

- "Diagram of Node Manager and Servers" on page 2-6
- "How Node Manager Starts an Administration Server" on page 2-7
- "How Node Manager Starts a Managed Server" on page 2-8
- "How Node Manager Restarts an Administration Server" on page 2-10
- "How Node Manager Restarts a Managed Server" on page 2-10
- "How Node Manager Shuts Down a Server Instance" on page 2-12

Diagram of Node Manager and Servers

Figure 2-1 illustrates the relationship between Node Manager, its clients, and the server instances it controls.

Figure 2-1 Node Manager in the WebLogic Server Environment



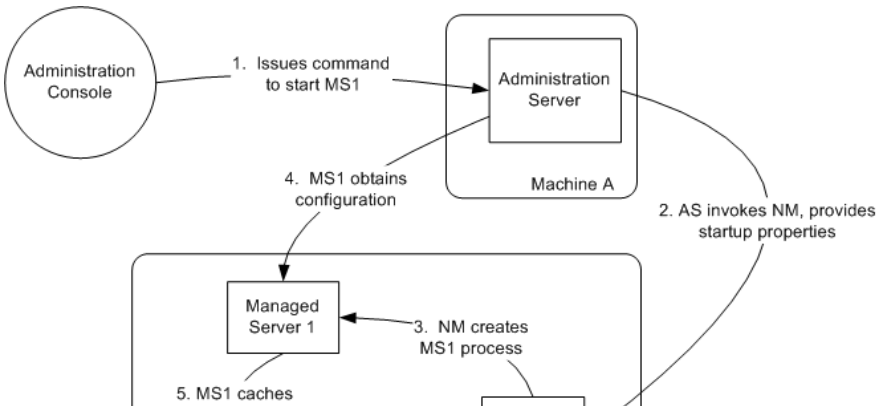
How Node Manager Starts an Administration Server

Figure 2-2 illustrates the process of starting an Administration Server with Node Manager.

This section assumes that you have installed the Administration Server and created its domain directory using the Configuration Wizard.

Node Manager is running on Machine A, which hosts the Administration Server. The stand-alone Node Manager client is remote.

Figure 2-2 Starting an Administration Server



1. An authorized user issues the WLST offline command, `nmConnect` to connect to the Node Manager process on the machine that hosts the Administration Server, and issues a command to start the Administration Server. (If the Node Manager instance is the SSH version, the user can connect using the SSH client).

   The start command identifies the domain and server instance to start, and in the case of the Java Node Manager, provides the Node Manager username and password.

   Note: If the user has previously connected to the Node Manager, a `boot.properties` file exists, and the user does not have to supply username and password.

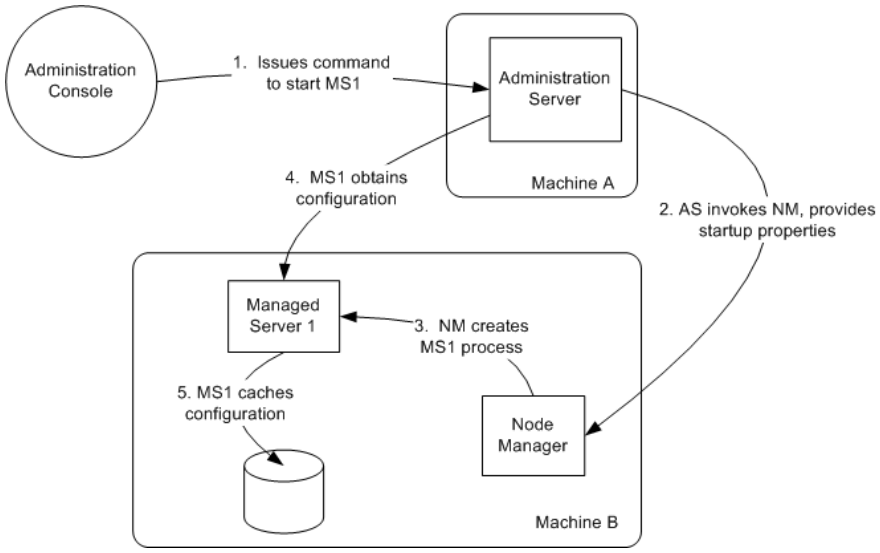
2. Node Manager looks up the domain directory in `nodemanager.domains`, and authenticates the user credentials using a local file that contains the encrypted username and password.
3. Node Manager creates the Administration Server process.
4. The Administration Server obtains the domain configuration from its `config` directory.

How Node Manager Starts a Managed Server

Figure 2-3 illustrates the process of starting a Managed Server with Node Manager.

Node Manager is running on Machine B, which hosts Managed Server 1. The Administration Server for the domain is running on Machine A.

Figure 2-3 Starting a Managed Server



1. From the Administration Console, the user issues a start command for Managed Server 1.

   Note: A stand-alone client can also issue a start command for a Managed Server.

2. The Administration Server issues a start command for Managed Server 1 to the Node Manager on the Machine B, providing the remote start properties configured for Managed Server 1. For information about the arguments and how to specify them, see "Step 5: Configuring Remote Startup Arguments" on page 3-4.

3. Node Manager starts Managed Server 1.

   Node Manager starts the Managed Server using the same root directory where the Node Manager process is running. To run the Managed Server in a different directory, set the Root Directory attribute in the Server—>Configuration—>Server Start console page.

4. Managed Server 1 contacts the Administration Server to check for updates to its configuration information.

5. If there are outstanding changes to the domain configuration, Managed Server 1 updates its local cache of configuration data.

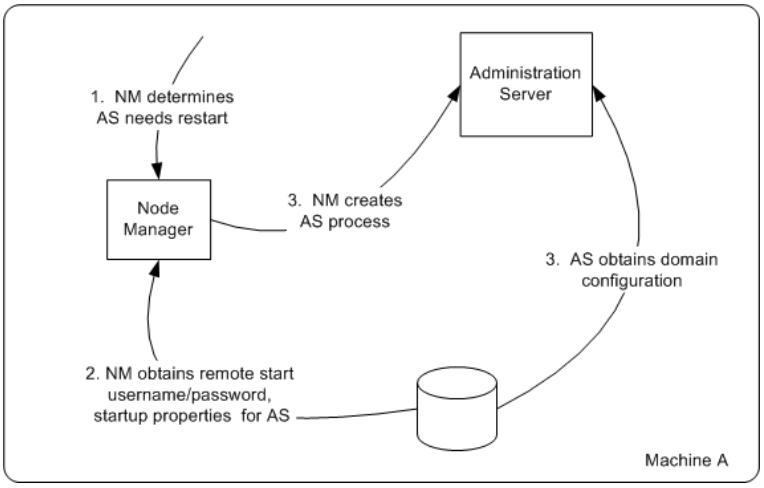
How Node Manager Restarts an Administration Server

Figure 2-4 illustrates the process of restarting an Administration Server with Node Manager.

Node Manager is running on the machine that hosts the Administration Server. The Administration Server, which was initially started with Node Manager, has exited. The Administration Server's `AutoRestart` attribute is set to `true`.

Note: If a server instance's `AutoRestart` attribute is set to `false`, Node Manager will not restart it.

Figure 2-4 Restarting an Administration Server



1. Node Manager determines from the Administration Server process exit code that it requires restart.
2. Node Manager obtains the username and password for starting the Administration Server from the `boot.properties` file, and the server startup properties from the `<server_name>/data/nodemanager/startup.properties` file.
3. Node Manager starts the Administration Server.
4. The Administration Server reads its configuration data and starts up.

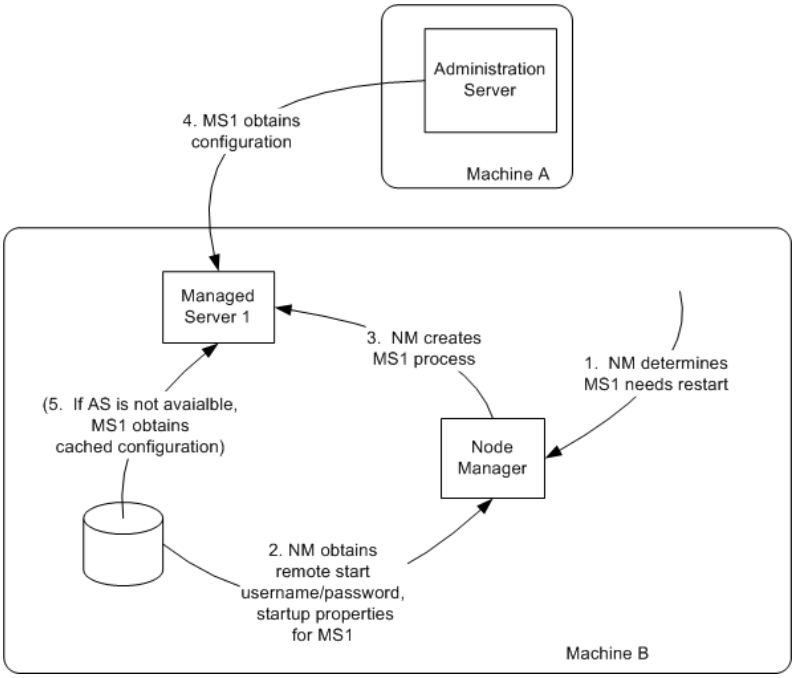
How Node Manager Restarts a Managed Server

Figure 2-5 illustrates process of restarting a Managed Server with Node Manager.

Node Manager is running on Machine B, which hosts Managed Server 1. Managed Server 1, which was initially started with Node Manager, has exited. Managed Server 1's `AutoRestart` attribute is set to `true`.

Note: If a server instance's `AutoRestart` attribute is set to `false`, Node Manager will not restart it.

Figure 2-5 Restarting a Managed Server



1. Node Manager determines from Managed Server 1's last known state that it requires restarting.
2. Node Manager obtains the username and password for starting Managed Server 1 from the `boot.properties` file, and the server startup properties from the `startup.properties` file. These server-specific files are located in the server directory for Managed Server 1.
3. Node Manager starts Managed Server 1.

   Note: Node Manager waits `RestartDelaySeconds` after a server instances fails before attempting to restart it.

4. Managed Server 1 attempts to contact the Administration Server to check for updates to its configuration data. If it contacts the Administration Server and obtains updated configuration data, it updates its local cache of the `config` directory.
5. If Managed Server 1 fails to contact the Administration Server, and if Managed Server Independence mode (MSI) is enabled, Managed Server 1 uses its locally cached configuration data.

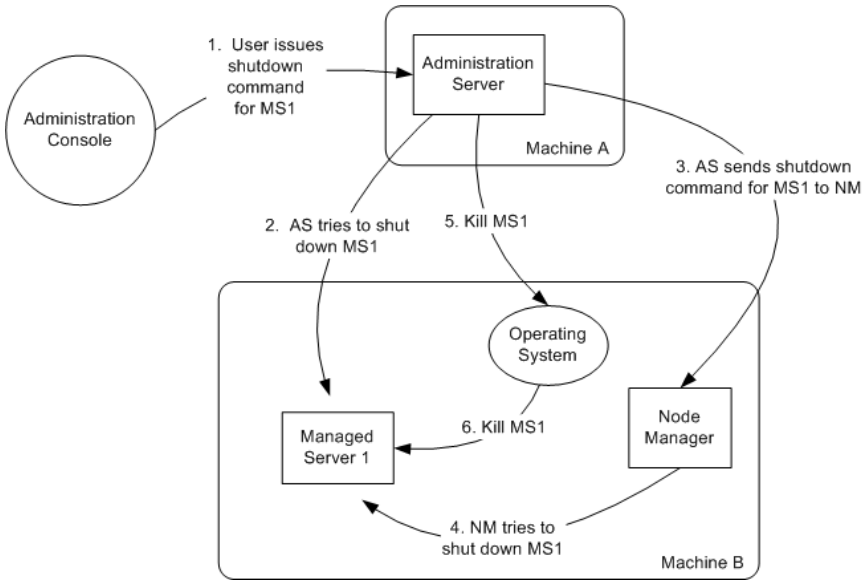
   Note: Managed Server Independence mode is enabled by default.

How Node Manager Shuts Down a Server Instance

Figure 2-6 illustrates the communications involved in shutting down a Managed Server that is under Node Manager control. Depending on the state and availability of the Managed Server, Node Manager might need to try alternative strategies to successfully initiate the shutdown.

Node Manager is running on Machine B, which hosts Managed Server 1.

Figure 2-6 Shutting Down a Server Instance Under Node Manager Control



1. Through the Administration Console, an authorized user issues a shutdown command for Managed Server 1.

2. The Administration Server issues the shutdown command directly to Managed Server 1. If it successfully contacts Managed Server 1, Managed Server 1 performs the shutdown sequence described in "Graceful Shutdown" in *Managing Server Startup and Shutdown*.
3. If, in the previous step, the Administration Server failed to contact Managed Server 1, it issues a shutdown command for Managed Server 1 to Node Manager on Machine B.
4. Node Manager issues a request to the operating system to kill Managed Server 1.
5. The operating system ends the Managed Server 1 process.

Node Manager and System Crash Recovery

To ensure that Node Manager properly restarts servers after a system crash, you must perform the following:

- Ensure that CrashRecoveryEnabled is set to true.

  The CrashRecoveryEnabled configuration property allows Node Manager to restart servers after a system crash. The property is not enabled by default.

- You should start the Administration Server via Node Manager.
- All managed servers should be started via the Administration Server. You can accomplish this via WLST or the Administration Console.

After the system is restarted, Node Manager checks each managed domain specified in the nodemanager.domains file to determine if there are any server instances that were not cleanly shutdown. This is determined by the presence of any lock files which are created by Node Manager when a WebLogic Server process is created. This lock file contains the process identifier for WebLogic Server startup script. If the lock file exists, but the process ID is not running, Node Manager will attempt to automatically restart the server.

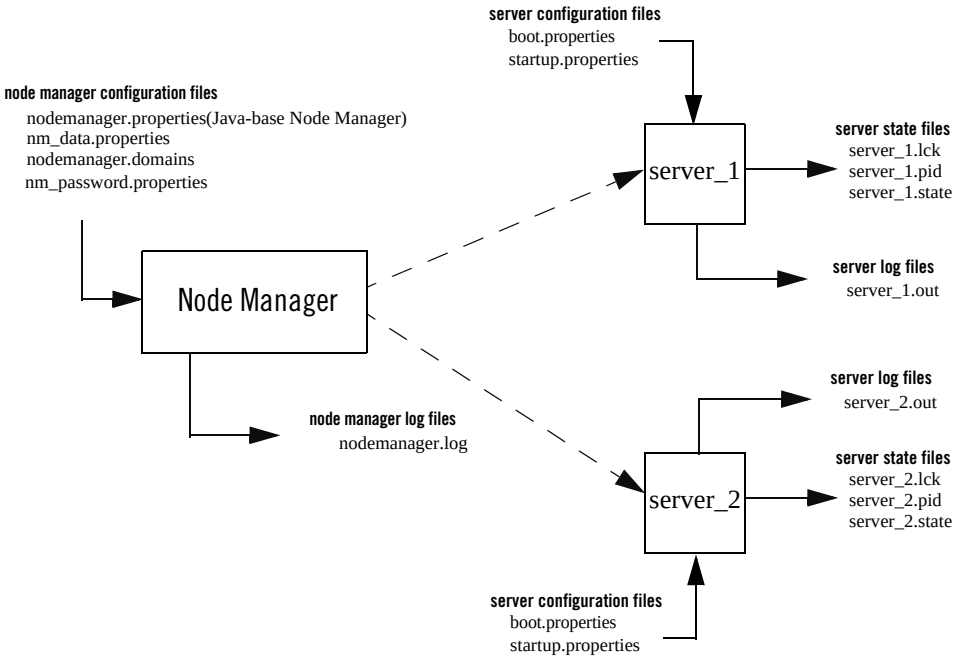
If the process is running, Node Manager performs an additional check to access the management servlet running in the process to verify that the process corresponding to the process ID is a WebLogic Server instance.

Note: When Node Manager performs a check to access the management servlet, an alert may appear in the server log regarding improper credentials.

Node Manager Configuration and Log Files

In managing multiple servers, Node Manager uses multiple configuration files and outputs log files to multiple directories, as shown in the following figure. For a description of these files, see "Node Manager Configuration and Log Files" on page 2-14.

Figure 2-7 Node Manager Configuration and Logging Environment



The following sections describe Node Manager configuration and log files:

- "Configuration Files" on page 2-14
- "Log Files" on page 2-17

Configuration Files

Except where noted, configuration files apply to both Java-based and script-based Node Manager.

nodemanager.properties

This is the configuration file used by the Java-based version of Node Manager.See "Reviewing nodemanager.properties" on page 4-3.

This file is located in `WL_HOME/common/nodemanager`.

nodemanager.domains

This file contains mappings between the names of domains managed by Node Manager and their corresponding directories. See "Step 4: Configuring nodemanager.domains File" on page 3-4.

This file is located in `WL_HOME/common/nodemanager`.

nm_data.properties

This file stores the encryption data the Node Manager uses a symmetric encryption key. The data is stored in encrypted form.

This file is located in `WL_HOME/common/nodemanager`.

nm_password.properties

This file stores the Node Manager user name and password. See "Step 2: Specify Node Manager Username and Password" on page 3-3.

This file is located in `DOMAIN_HOME/config/nodemanager`.

boot.properties

Node Manager uses this file to specify a boot identity when starting a server. See "General Node Manager Configuration" on page 3-1.

This file is located in `domain-name/servers/server_name/data/nodemanager`.

startup.properties

Each Managed Server instance has its own `startup.properties` file with properties that control how Node Manager starts up and controls the server. Node Manager automatically creates this file by using properties passed to Node Manager when the Administrative Server was last used to start the server. This allows a Node Manager client or startup scripts to restart a Managed Server using the same properties last used by the Administrative Server.

For more information on startup.properties, see "Step 6: Setting Server Startup Properties" on page 3-5. These properties correspond to the server startup attributes contained in ServerStartMBean and the health monitoring attributes in ServerStartMBean.

This file is located in `domain-name/servers/server_name/data/nodemanager`.

server_name.addr

server_name.addr stores the IP address added when a server starts or is migrated. This file is generated after the server IP address is successfully brought online during migration. *server_name*.addr is deleted when the IP address is brought offline. The server IP address is used to validate remove requests to prevent addresses being erroneously removed while shutting down the server.

This file is located in `domain-name/servers/server_name/data/nodemanager`.

server_name.lck

`server_name.lck` is generated by each server and contains an internally used lock ID.

This file is located in `domain-name/servers/server_name/data/nodemanager`

server_name.pid

`server_name.pid` is generated by each server and contains the process ID of the server. Node Manager checks the process ID generated by the server during crash recovery.

This file is located in `domain-name/servers/server_name/data/nodemanager`

server_name.state

`server_name.state` is generated by the server and contains the server's current state. Node Manager monitors the contents of this file to determine the current state of the server.

Note: Do not delete or alter this file. Without this file Node Manager cannot determine the current state of the server.

This file is located in `domain-name/servers/server_name/data/nodemanager`.

Log Files

Use the Node Manager and WebLogic Server log files to help troubleshoot problems in starting or stopping individual Managed Servers.f

Table 2-1 Node Manager Log File Locations

Log File	Location
Node Manager Log File	`NodeManagerHome/nodemanager.log`
Node Manager Server Instance Log Files	`domain-name/servers/<server-name>/logs/<server-name>.out`
Web Logic Server Log Files	`domain-name/servers/<server-name>/logs/<server_name>.log`

nodemanager.log

Node Manager creates a log file located in `NodeManagerHome/nodemanager.log`. This log file stores data about all of the domains administered by Node Manager.

This log file is generated by Node Manager and contains data for all domains that are controlled by Node Manager on a given physical machine. See "nodemanager.log" on page 2-17.

This file is located in `WL_HOME/common/nodemanager`.

Log output is appended to the current `nodemanager.log`. Log rotation is disabled by default, but can be enabled by setting `LogCount` in `nodemanager.properties`.

You can view the Node Manager log file by:

- Selecting Machines—>Monitoring—>Node Manager Log page in the Administration Console
- Using the WLST `nmLog` command

server_name.out

For each server instance that it controls, Node Manager maintains a log file that contains `stdout` and `stderr` messages generated by the server instance. If the remote start debug property is enabled as a remote start property for the server instance, or if the Node Manager debug property is enabled, Node Manager will include additional debut information in the server output log information.

Note: You cannot limit the size of the log files Node Manager creates. Logging to `stdout` is disabled by default.

This file is located in `domain_name/servers/<server_name>/logs`

Node Manager creates the server output log for a server instance in the server instance's `logs` directory, with the name:

`server-name``.out`

where *`server-name`* is the name of the server instance.

You can view the Node Manager log file for a particular server instance by:

- Selecting Diagnostics —>Log Files.
- Using the WLST `nmServerLog` command.

There is no limit to the number of server output logs that Node Manager can create.

WebLogic Server Log Files

A server instance under Node Manager control has its own log file, in addition to the log file created by Node Manager.

You can view the regular log file for a server instance by selecting Diagnostics->Log Files, selecting the server log file, and clicking View.

CHAPTER 3

General Node Manager Configuration

This chapter outlines general procedures that are applicable to both the Java and scripted version of Node Manager. The following topics are covered:

After you have performed the procedures outline in this chapter, see --- or --- to continue configuring Node Manager.

Overview of Node Manager Configuration

This section describes general Node Manager configuration that applies to the Java and script version of Node Manager. You should ensure than you have performed all of the items outlined in the following sections.

After you have you have performed general Node Manager configuration, you should perform the configuration procedures outlined in Chapter 4, "Configuring Java Node Manager" or Chapter 5, "Configuring Script Node Manager" depending on which version of Node Manager you are using.

Step 1: Configure Your Computer to Run Node Manager

Node Manager must run on each computer that hosts a WebLogic Server instance. Configure each computer as a Machine in WebLogic Server, and assign each server instance that you will control with Node Manager to the machine upon which it runs.

Ideally, Node Manager should run as an operating system service or daemon, so that it is automatically restarted in the event of system failure or reboot. For more information, see "Installing the Node Manager as a Windows Service" in the *Installation Guide*.

Node Manager is ready-to-run after WebLogic Server installation if you run Node Manager and the Administration Server on the same machine, and use the demonstration SSL configuration. By default, the following behaviors are configured:

- You can start a Managed Server using Node Manager through the Administration Console.
- Node Manager monitors the Managed Servers that it has started.
- Automatic restart of Managed Servers is enabled. Node Manager restarts server instances that it killed or were killed by another method.

Controlling and Configuring Node Manager Using WLST

The WebLogic Scripting Tool (WLST) is a command-line scripting interface that system administrators and operators use to monitor and manage WebLogic Server instances and domains. You can start, stop, and restart server instances remotely or locally, using WLST as a Node Manager client. In addition, WLST can obtain server status and retrieve the contents of the server output log.

Using nmConnect() in a Production Environment

By default, the `nmConnect()` command cannot be used in a production environment. You must perform the following procedures to use `nmConnect` in a production environment.

1. Start the administration server.

2. Using the Administration Console update the Node Manager credentials from the Advanced options under *domain_name*—>Security—>General.
3. Start WLST in online mode.
4. Run `nmEnroll()` using the following as an example:

   ```
   nmEnroll('C:/bea/user_projects/domains/prod_domain',
   'C:/bea/wlserver_10.0/common/nodemanager')
   ```

   Running `nmEnroll()` ensures that the correct NodeManager user and password token are supplied to each managed server. Once these are available for each managed server, you can use `nmConnect()` in a production environment.

   Note: You must run `nmEnroll()` on each machine that is running a managed server. Additionally, you should run `nmEnroll()` for each domain directory on each machine.

Step 2: Specify Node Manager Username and Password

The nm_password.properties file contains the Node Manager username and password. These are used to authenticate connection between a client (for example, the Administration server) and Node Manager.

Note: This username and password are only used to authenticate connections between Node Manager and clients. They are independent from the server admin ID and password.

This file is created when you use nmEnroll() to copy the necessary configurations files from one machine to another when creating a domain. After nm_password.properties is created, you can change the values for the Node Manager password and properties using the Administration Console. Changes are propagated to the nm_password.properties file and are picked up by Node Manager.

Note: If you edit nm_password.properties manually, you must restart Node Manager in order for the changes to take effect.

The nm_password.properties file must exist on each physical machine that runs Node Manager. However, the Node Manager username and password do not have to be identical on every machine within your domain.

Step 3: Configure a Machine to Use Node Manager

A WebLogic Server Machine resource associates a particular machine with the server instances it hosts, and specifies the connection attributes for the Node Manager process on that system.

Configure a machine definition for each machine that runs a Node Manager process using the Environment—>Machines—><*machine_name*>—>Node Manager page in the Administration Console. Enter the DNS name or IP address upon which Node Manager listens in the Listen Address box.

Step 4: Configuring nodemanager.domains File

The `nodemanager.domains` file specifies the domains that a Node Manager instance controls. Thus stand-alone clients do not need to specify the domain directory explicitly.

This file must contain an entry specifying the domain directory for each domain the Node Manager instance controls, in this form:

```
<domain-name>=<domain-directory>
```

When a user issues a command for a domain, Node Manager looks up the domain directory from `nodemanager.domains`.

This file provides additional security by restricting Node Manager client access to the domains listed in this file. The client can only execute commands for the domains listed in `nodemanager.domains`.

If you created your domain with the Configuration Wizard, the `nodemanager.domains` file was created automatically. If necessary, you can manually edit `nodemanager.domains` to add a domain.

Note: If you use the backslash character (\) in `nodemanager.domains`, you must escape it as (\\).

Step 5: Configuring Remote Startup Arguments

In the Server—>Configuration—>Server Start page for the Managed Server, specify the startup arguments that Node Manager will use to start a Managed Server. If you do not specify startup arguments for a Managed Server, Node Manager uses its own properties as defaults to start the Managed Server. For more information, see "Reviewing nodemanager.properties" on page 4-3. Although these defaults are sufficient to boot a Managed Server, to ensure a consistent and reliable boot process, configure startup arguments for each Managed Server instance.

If you will run Node Manager as a Windows Service, as described in "Installing the Node Manager as a Windows Service" in the *Installation Guide*, you must configure the following JVM property for each Managed Server that will be under Node Manager control:

- `-Xrs` for the Sun JVM, or

- `-Xnohup` for the Jrockit

If you do not set this option, Node Manager will not be able to restart a Managed Server after a system reboot, due to this sequence of events:

1. A reboot causes a running Managed Server to be killed before the Node Manager and Administration Server operating system services are shut down.
2. During the interval between the Managed Server being killed, and the Node Manager service being shut down, Node Manager continues to monitor the Managed Server, detects that it was killed, and attempts to restart it.
3. The operating system does not allow restart of the Managed Server because the machine is shutting down.
4. Node Manager marks the Managed Server as failed, and it will not start this server when the machine comes up again.

Starting a Managed Server with the `-Xrs` or `-Xnohup` option avoids this sequence of events by preventing the immediate shutdown of the Managed Server during machine shutdown.

Step 6: Setting Server Startup Properties

You can use Node Manager to set the startup properties for a server. These properties can be defined in startup.properties or passed as an object using administrative utilities such as WLST. The methods of setting startup properties and their valid values are outlined in the sections below.

startup.properties

Node Manager uses the startup.properties file to determine the startup and configuration when starting a server. This file is defined for each server instance and is located in:

`domain_home/servers/server_name/data/nodemanager/startup.properties`

The contents of `startup.properties` are derived from the Server Mbean, or the Cluster Mbean if the server is part of a cluster. For more information, see the Mbean reference.

Setting Startup Properties Using WLST

When using the WLST nmStart() command, the server configuration can not be determined directly. Therefore, you must pass the server start properties as a WLST properties object to the nmStart() command.

Server Startup Properties

The following server startup properties can be passed to a server when started via Node Manager.

Table 3-1 Server Startup Properties

Property	Description
JavaHome	Defines the Java home directory used when starting the server.
Arguments	The arguments used when starting the server.
SSLArguments	These arguments are used when you have enabled the domain-wide administration port.
RestartMax	The number of times Node Manager can attempt to restart the server.
RestartDelaySeconds	The number of seconds Node Manager should wait before attempting to restart the server.
ClassPath	The classpath to use when starting a server.
BEAHome	The BEA home directory to use when starting a server.
AdminURL	The URL of the administration server. **Note:** This value should only be specified in the startup.properties file for a managed server.
AutoRestart	Specifies whether Node Manager can automatically restart this server if it fails.
AutoKillIfFailed	Specifies whether Node Manager should automatically kill the server if its health status is `failed`.
SecurityPolicyFile	Specifies the security policy file to use when starting this server.
ServerIP	The IP address of the server.

Step 7: Define the Administration Server Address

Make sure that a Listen Address is defined for each Administration Server that will connect to the Node Manager process. If the Listen Address for an Administration Server is not defined, when Node Manager starts a Managed Server it will direct the Managed Server to contact localhost for its configuration information.

Set the Listen Address using the Servers—>Configuration—>General page in the Administration Console.

Step 8: Set the Node Manager Environment Variables

Node Manager requires you to set several environment variables before you start it.

You can set these variables manually on the command line or you can create a start script that sets them automatically. The sample start scripts provided with WebLogic Server — `startNodeManager.cmd` and `startNodeManager.sh` — set the required variables.

Table 3-2 Node Manager Environment Variables

Environment Variable	Description
`JAVA_HOME`	JDK root directory used by Node Manager. For example: `set JAVA_HOME=c:\bea\jdk131` Node Manager has the same JDK version requirements as WebLogic Server.
`WL_HOME`	WebLogic Server installation directory. For example: `set WL_HOME=c:\bea\wlserver_10.0`
`PATH`	Must include the WebLogic Server `bin` directory and path to your Java executable. For example: `set PATH=%WL_HOME%\server\bin;%JAVA_HOME%\bin;%PATH%`

`LD_LIBRARY_PATH` (UNIX only)	For HP UX and Solaris systems, you must include the path to the native Node Manager libraries.
	Solaris example:
	`LD_LIBRARY_PATH:$WL_HOME/server/lib/solaris:$WL_HOME/server/lib/solaris/oci816_8`
	HP UX example:
	`SHLIB_PATH=$SHLIB_PATH:$WL_HOME/server/lib/hpux11:$WL_HOME/server/lib/hpux11/oci816_8`

`CLASSPATH`	You can set the Node Manager `CLASSPATH` either as an option on the `java` command line used to start Node Manager, or as an environment variable. Windows NT example: `set CLASSPATH=.;%WL_HOME%\server\lib\weblogic_sp.jar;%WL_HOME%\server\lib\weblogic.jar`

CHAPTER 4

Configuring Java Node Manager

This chapter provides information on configuration the Java version of Node Manager. The following topics are covered:

Running Node Manager as a Service

It is recommended that you configure Node Manager to run as an operating system service or a Windows service on Windows systems. By default, the operating system service starts up Node Manager to listen on `localhost:5556`.

When you configure Node Manager to accept commands from remote systems, you must uninstall the default Node Manager service, then reinstall it to listen on a non-localhost Listen Address.

Depending on your platform, follow the instructions in "Reconfigure Startup Service for Windows Installations" or "Configuring Java-based Node Manager Security"

Reconfigure Startup Service for Windows Installations

The directory *WL_HOME*\server\bin (where *WL_HOME* is the top-level directory for the WebLogic Server installation) contains uninstallNodeMgrSvc.cmd, a script for uninstalling the Node Manager service, and installNodeMgrSvc.cmd, a script for installing Node Manager as a service.

1. Delete the service using uninstallNodeMgrSvc.cmd.
2. Edit installNodeMgrSvc.cmd to specify Node Manager's Listen Address and Listen Port.

   Make the same edits to uninstallNodeMgrSvc.cmd as you make to installNodeMgrSvc.cmd, so that you can successfully uninstall the service in the future, as desired.
3. Run installNodeMgrSvc.cmd to re-install Node Manager as a service, listening on the updated address and port.

Configuring Java-based Node Manager Security

Node Manager security relies on a one-way SSL connection between the client and server.

If you are establishing a command line connection to the Java Node Manager using the WebLogic Server Scripting Tool (WLST) nmConnect command, you provide the Node Manager user name and password. Node Manager verifies the username and password against the domain's nm_password.properties file. For more information on nm_password.properties, see "Step 2: Specify Node Manager Username and Password" on page 3-3.

Node Manager credentials are located on the Security>General>Advanced Options Console page.

Administration Console users do not need to explicitly provide credentials to connect to Node Manager—the Node Manager user name and password are available in the domain configuration and are provided automatically.

Remote Server Start Security for Java-based Node Manager

A remote start user name and password is required to start a server instance with Node Manager. These credentials are provided differently for Administration Servers and Managed Servers.

- Credentials for Managed Servers—When you invoke Node Manager to start a Managed Server it obtains its remote start name and password from the Administration Server.

- Credentials for Administration Servers—When you invoke Node Manager to start an Administration Server, the remote start user name can be provided on the command line, or obtained from the Administration Server's `boot.properties` file. The Configuration Wizard initializes the `boot.properties` file and the `startup.properties` file for an Administration Server when you create the domain.

Any server instance started by Node Manager encrypts and saves the credentials with which it started in a server-specific `boot.properties` file, for use in automatic restarts.

Reviewing nodemanager.properties

Node Manager properties define a variety of configuration settings for a Java-based Node Manager process. You can specify Node Manager properties on the command line or define them in the `nodemanager.properties` file, which is created in the directory where you start Node Manager the first time it starts up after installation of WebLogic Server. Values supplied on the command line override the values in `nodemanager.properties`.

`nodemanager.properties` is created in the directory specified in NodeManagerHome. If NodeManagerHome is not defined, nodemanager.properties is created in the current directory.

Each time you start Node Manager, it looks for `nodemanager.properties` in the current directory, and creates the file if it does not exist in that directory. You cannot access the file until Node Manager has started up once.

Table 4-1 describes Node Manager properties.

In many environments, the SSL-related properties in `nodemanager.properties` may be the only Node Manager properties that you must explicitly define. However, `nodemanager.properties` also contains non-SSL properties in that you might need to specify, depending on your environment and preferences. For example:

- For a non-Windows installation, it might be appropriate to specify the `StartScriptEnabled` and `NativeVersionEnabled` properties.

- If Node Manager runs on a multi-homed system, and you want to control which address and port it uses, define `ListenAddress` and `ListenPort`.

Table 4-1 Node Manager Properties

Node Manager Property	Description	Default
`LogFile`	Location of the Node Manager log file.	`NodeManagerHome/nodemanager.log`
`LogLimit`	Maximum size of the Node Manager Log specified as an integer. When this limit is reached, a new log file is started.	`unlimited`
`LogCount`	Maximum number of log files to create when `LogLimit` is exceeded.	1
`LogAppend`	If set to `true`, then a new log file is not created when the Node Manager restarts; the existing log is appended instead.	`true`
`LogToStderr`	If set to `true`, the log output is also sent to the standard error output.	`false`
`LogLevel`	Severity level of logging used for the Node Manager log. Node Manager uses the same logging levels as WebLogic server.	`INFO`
`LogFormatter`	Name of formatter class to use for NM log messages.	`weblogic.nodemanager.server.LogFormatter`
`CrashRecoveryEnabled`	Enables system crash recovery.	`false`
`SecureListener`	If set to `true`, use the SSL listener, otherwise use the plain socket	`true`
`CipherSuite`	The name of the cipher suite to use with the SSL listener.	`TLS_RSA_EXPORT_WITH_RC4_40_MD5`
`StartScriptEnabled`	If true, use the start script specified by `StartScriptName` to start a server. For more information, see "Configuring Node Manager to Use Start and Stop Scripts."	`false`

Table 4-1 Node Manager Properties

Node Manager Property	Description	Default
`StartScriptName`	The name of the start script, located in the domain directory	`startWebLogic.sh (UNIX)` or `startWebLogic.cmd (Windows)`
`StopScriptEnabled`	If `true`, execute the stop script specified by `StopScriptName` after the server has shutdown. For more information, see "Configuring Node Manager to Use Start and Stop Scripts."	`false`
`StopScriptName`	The name of the script to be executed after server shutdown.	`none`
`RestartInterval`	The amount of time Node Manager will spend attempting to restart a failed server. Within this period of time Node Manager will attempt to restart the failed server up to the number defined by `RestartMax`. By default, Node Manager will attempt to restart a server indefinitely until the `FAILED_NOT_RESTARTABLE` state is reached	`0`
`RestartMax`	The number of times Node Manager will attempt to restart a failed server within the interval defined by RestartInterval. `RestarMax` is only recognized if `RestartInterval` is defined.	`NA`
`DomainsFile`	The name of the `nodemanager.domains` file	`NodeManagerHome/nodemanager.domains`
`DomainsFileEnabled`	If set to `true`, use the file specified in `DomainsFile`. If `false`, assumes the domain of the current directory or of `WL_HOME`.	`true`

Table 4-1 Node Manager Properties

Node Manager Property	Description	Default
`StateCheckInterval`	Specifies the interval Node Manager waits to perform a check of the server state.	`500 milliseconds`
`CustomIdentityAlias`	Specifies the alias when loading the private key into the keystore. This property is required when the `Keystores` property is set as `CustomIdentityandCustomTrust` or `CustomIdentityAndJavaStandardTrust`.	`none`
`CustomIdentityKeyStoreFileName`	Specifies the file name of the Identity keystore (meaning the keystore that contains the private key for the Node Manager). This property is required when the `Keystores` property is set as `CustomIdentity` and `CustomTrust` or `CustomIdentityAndJavaStandardTrust`.	`none`
`CustomIdentityKeyStorePassPhrase`	Specifies the password defined when creating the Identity keystore. This field is optional or required depending on the type of keystore. All keystores require the passphrase in order to write to the keystore. However, some keystores do not require the passphrase to read from the keystore. WebLogic Server only reads from the keystore, so whether or not you define this property depends on the requirements of the keystore.	`none`
`CustomIdentityKeyStoreType`	Specifies the type of the Identity keystore. Generally, this is JKS. This property is optional.	`default keystore type from java.security`

Table 4-1 Node Manager Properties

Node Manager Property	Description	Default
`CustomIdentity PrivateKeyPassPhrase`	Specifies the password used to retrieve the private key for WebLogic Server from the Identity keystore. This property is required when the `Keystores` property is set as `CustomIdentityandCustomTrust` or `CustomIdentityAndJavaStandardTrust`.	`none`
`JavaHome`	The Java home directory that Node Manager uses to start a Managed Servers on this machine, if the Managed Server does not have a Java home configured in its Remote Start tab. If not specified in either place, Node Manager uses the Java home defined for the Node Manager process.	`none`
`JavaStandardTrustKey StorePassPhrase`	Specifies the password defined when creating the Trust keystore. This field is optional or required depending on the type of keystore. All keystores require the passphrase in order to write to the keystore. However, some keystores do not require the passphrase to read from the keystore. WebLogic Server only reads from the keystore, so whether or not you define this property depends on the requirements of the keystore.This property is required when the `Keystores` property is set as `CustomIdentityandJavaStandard` Trust or `DemoIdentityAndDemoTrust`.	`none`

Table 4-1 Node Manager Properties

Node Manager Property	Description	Default
`KeyStores`	Indicates the keystore configuration the Node Manager uses to find its identity (private key and digital certificate) and trust (trusted CA certificates). Possible values are: • `DemoIdentityAndDemoTrust` Use the demonstration Identity and Trust keystores located in the `BEA_HOME\server\lib` directory that are configured by default. The demonstration Trust keystore trusts all the certificate authorities in the Java Standard Trust keystore (`JAVA_HOME\jre\lib\security\cacerts`) • `CustomIdentityAndJavaStandardTrust` Uses a keystore you create, and the trusted CAs defined in the cacerts file in the `JAVA_HOME\jre\lib\security\cacerts` directory. • `CustomIdentityAndCustomTrust` Uses Identity and Trust keystores you create.	`DemoIdentityAndDemoTrust`
`ListenAddress`	Any address upon which the machine running Node Manager can listen for connection requests. This argument deprecates `weblogic.nodemanager.listenAddress`.	`null` With this setting, Node Manager will listen on any IP address on the machine
`ListenPort`	The TCP port number on which Node Manager listens for connection requests. This argument deprecates weblogic.nodemanager.listenPort.	`5556`

Table 4-1 Node Manager Properties

Node Manager Property	Description	Default
`NativeVersionEnabled`	A value of true causes native libraries for the operating system to be used. For UNIX systems other than Solaris, HP-UX, or Linux, set this property to false to run Node Manager in non-native mode. This will cause Node Manager to use the start script specified by the StartScriptEnabled property to start Managed Servers.	`true`
`NodeManagerHome`	Node Manager root directory which contains the following configuration and log files: • nm_data.properties • nodemanager.domains • nodemanager.log • nodemanager.properties For more information on these files, see "Node Manager Configuration and Log Files." **Note:** By default, `NodeManagerHome` is `WL_HOME/common/nodemanager`. In a production environment, you may want to customize the location of the Node Manager root directory.	`NodeManagerHome`
`WeblogicHome`	Root directory of the WebLogic Server installation. This is used as the default value of `-Dweblogic.RootDirectory` for a Managed Server that does not have a root directory configured in its Remote Start tab. If not specified in either place, Node Manager starts the Managed Server in the directory where Node Manager runs.	`none`

Table 4-1 Node Manager Properties

Node Manager Property	Description	Default
`keyFile`	The path to the private key file to use for SSL communication with the Administration Server. **Note:** This property is used only in the process of upgrading from WebLogic Server, Version 7.x to Version 9.x.	`none`
`keyPassword`	The password used to access the encrypted private key in the key file. **Note:** This property is used only in the process of upgrading from WebLogic Server, Version 7.x to Version 9.x.	`none`
`certificateFile`	Specifies the path to the certificate file used for SSL authentication. **Note:** This property is used only in the process of upgrading from WebLogic Server, Version 7.x to Version 9.x.	`none`

Deprecated Node Manager Properties

This section lists the Node Manager properties that are deprecated in WebLogic Server 9.x.

Note: These properties are published for backwards compatibility and should not be used. SSL configurations will continue to work when migrating to WebLogic Server 9.x. However, the trusted key store is not used when running Node Manager.

Table 4-2 Deprecated Node Manager Properties

Node Manager Property	Description	Reason Deprecated
`CustomTrustKeyPass Phrase (Deprecated)`	The password used to access the encrypted private key in the key file.	Using 1-way SSL, Node Manager does not need access to a trusted key store.
`CustomTrustKeyStore FileName (Deprecated)`	Specifies the file name of the Trust keystore (meaning the keystore that contains the trusted CA certificates for the Node Manager). This property is required when the Keystores property is set as CustomIdentityandCustomTrust.	Using 1-way SSL, Node Manager does not need access to a trusted key store.
`CustomTrustKeyStore PassPhrase (Deprecated)`	Specifies the password defined when creating the Trust keystore. This field is optional or required depending on the type of keystore. All keystores require the passphrase in order to write to the keystore. However, some keystores do not require the passphrase to read from the keystore. WebLogic Server only reads from the keystore, so whether or not you define this property depends on the requirements of the keystore.	Using 1-way SSL, Node Manager does not need access to a trusted key store.

Table 4-2 Deprecated Node Manager Properties

Node Manager Property	Description	Reason Deprecated
`CustomTrustKeyStore Type (Deprecated)`	Specifies the type of the Trust keystore. Generally, this is JKS. This property is optional.	Using 1-way SSL, Node Manager does not need access to a trusted key store.
`JavaStandardTrustKey StorePassPhrase (Deprecated)`	Specifies the password defined when creating the Trust keystore. This field is optional or required depending on the type of keystore. All keystores require the passphrase in order to write to the keystore. However, some keystores do not require the passphrase to read from the keystore. WebLogic Server only reads from the keystore, so whether or not you define this property depends on the requirements of the keystore.This property is required when the Keystores property is set as CustomIdentityandJavaStandardTrust or DemoIdentityAndDemoTrust.	Using 1-way SSL, Node Manager does not need access to a trusted key store.

Configuring Node Manager to Use Start and Stop Scripts

You can configure Node Manager to use a script to start a managed server or to execute a script after server shutdown has completed. These scripts can be used to perform tasks that need to be performed before a server is started or after it is shutdown. Mounting and unmounting remote disks is one example of a task that can be performed using scripts.

Note: Node Manager uses startup scripts to perform any required configuration, then start the server. In contrast, stop scripts are executed after the server has shutdown.

Script Location

Both the Start and Stop scripts should be placed in the following directory:

`<DOMAIN_HOME>/bin/service_migration`

Script execution should occur relative to this directory.

Best Practices when Using Start and Stop Scripts

When using start and stop scripts to control server behavior, BEA recommends that you only edit the top line of the scripts that are provided. This ensure that all of the necessary environment variables are used during script execution.

Using Start Scripts

You can use a start script allows you to specify required startup properties and perform any other work you need performed at start up. To define a start script:

1. In the `nodemanager.properties` file, set the `StartScriptEnabled` property to `true`. (The default is false.) If your start script is named `startWebLogic.sh` or `startWebLogic.cmd`, Node Manager uses one of those scripts as the default.
2. If you want to specify a custom start script, set the `StartScriptName` property to the name of your script in the `nodemanager.properties` file.

Using Stop Scripts

You can use a stop script to perform any tasks that are required after the server has failed.

Note: Stop scripts are used only to execute a script after a server fails and must be migrated.

To define a stop script:

1. In the `nodemanager.properties` file, set the `StopScriptEnabled` property to `true`.
2. Set the `StopScriptName` property to the name of your script in the `nodemanager.properties` file.

The following example shows a stop script that can be used to unmout a disk on UNIX systems:

```
#!/bin/sh
FS=/cluster/d2
if grep $FS /etc/mnttab > /dev/null 2>&1 ; then
sync
       PIDS=`/usr/local/bin/lsof $FS | awk
       '{if ($2 ~/[0-9]+/) { print $2} }' | sort -u`
kill -9 $PIDS
sleep 1
sync
```

```
        /usr/sbin/umount -f $FS
fi
```

Using SSL With Java-based Node Manager

Administration Servers and Managed Servers communicate with Java-based Node Manager using one-way SSL.

The default WebLogic Server installation includes demonstration Identity and Trust keystores that allow you to use SSL out of the box. The keystores—`DemoIdentity.jks` and `DemoTrust.jks`—are installed in *`WL_HOME`*`/server/lib`. For testing and development purposes, the keystore configuration is complete.

Configuring SSL for a production environment involves obtaining identity and trust for the Node Manager and each Administration and Managed Server with which the Node Manager will be communicating and then configuring the Node Manager, the Administration Server, and any Managed Servers with the proper identity and trust. In addition, the use of host name verification and the Administration port must be taken into consideration. To configure production SSL components, see "Configuring the SSL Protocol" in *Managing WebLogic Security*.

Configuring Node Manager on Multiple Machines

If you have a domain that has managed servers on multiple physical machines, you must ensure that Node Manager is installed and configured on each machine. You can use the WLST command nmEnroll to copy all of the required domain and configuration information from one machine to another. For more information, see "Accessing Node Manager" on page 2-3 and "nmEnroll() " in *WebLogic Scripting Tool*.

CHAPTER 5

Configuring Script Node Manager

The following sections describe how to configure script-based Node Manager:

Overview

The SSH Node Manager is a shell script, `wlscontrol.sh`, located in `{WL_HOME}/common/bin/`. `wlscontrol.sh` must exist on each machine that hosts server instances that you want to control with Node Manager. This script can be customized to meet site-specific requirements.

You must have an SSH client executable on each machine where Node Manager or a Node Manager client runs. This script must also be in the path of the user id running it. Typically, an SSH client is a standard part of a Unix or Linux installation.

Creating a Node Manager User

Before running Node Manager, you should create a dedicated UNIX user account for performing Node Manager functions. This user should be added to all machines that will host the SSH Node

Manager and to all machines that will host a Node Manager client, including the Administration Server.

Configuring Node Manager as an xinetd Service

When configuring Node Manager to run as an `inetd` or `xinetd` service, the following considerations apply:

- Ensure that `NodeManagerHome` and other system properties are defined.
- If xinetd is configured with `libwrap`, you should add the `NOLIBWRAP` flag.
- Ensure that the hosts.deny and hosts.allow files are configured correctly.
- Depending on your network environment, additional configuration may be necessary.

The following example shows how Node Manager can be configured within `xinetd`:

```
# default: off
# description:nodemanager as a service
service nodemgrsvc
{
   type            = UNLISTED
   disable         = no
   socket_type     = stream
   protocol        = tcp
   wait            = yes
   user            = <username>
   port            = 5556
   flags           = NOLIBWRAP
   log_on_success += DURATION HOST USERID
   server          = <path-to-jave>/java
   env             = CLASSPATH=<cp> LD_LIBRARY_PATH=<ldpath>
   server_args     = -client -DNodeManagerHome=<NMHome> <java options>
   <nodemanager options> weblogic.NodeManager -v
}
```

Overriding the Default SSH Port

The default SSH port used by Node Manager is 22. You can override that setting in the following ways:

- Set the `Port=` parameter in the `~/.ssh/config` file to set the default port for an individual user.
- Set the `Port=` parameter in the `/etc/ssh_config` file to set the default port across the entire system.
- Start the Administration Server using the following system property:

  ```
  -Dweblogic.nodemanager.ShellCommand="ssh -o PasswordAuthentication=no -p
  %P %H wlscontrol.sh -d %D -r %R -s %S %C"
  ```

  After starting the server, you can edit the SSH port in the Administration Server's configuration file.

Configuring Script-based Node Manager Security

The Node Manager SSH shell script relies on SSH user-based security to provide a secure trust relationship between users on different machines. Authentication is not required. You create a UNIX user account—typically one per domain—for running Node Manager commands and scripts. A user logged in as this user can issue Node Manager commands without providing a username and password.

Note: You must also ensure that the Node Manager and WebLogic Server commands are available in the path of the UNIX user ID used to run them.

Security for WebLogic Server Scripts

To perform Server migration and other tasks, the user ID executing scripts such as wlscontrol.sh must have enough sufficient security permissions. This includes being able to bring an IP address online or take an IP address offline via a network interface.

Server migration is performed by the cluster master when it detects that a server has failed. It then uses SSH to launch a script on the target machine to begin the migration. The script on the target machine runs as the same user ID running the server on the cluster master.

The commands required to perform server migration are ifconfig and arping. Since these scripts require elevated OS privileges, it is important to note that this can prevent a potential security hole.

Using sudo, you can configure your SSH to only allow ifconfig and arping to be run using elevated privileges.

Remote Server Start Security for Script-based Node Manager

A remote start user name and password is required to start a server instance with Node Manager. These credentials are provided differently for Administration Servers and Managed Servers.

- Credentials for Managed Servers—When you invoke Node Manager to start a Managed Server it obtains its remote start name and password from the Administration Server.
- Credentials for Administration Servers—When you invoke Node Manager to start an Administration Server, the remote start user name can be provided on the command line, or obtained from the Administration Server's `boot.properties` file. The Configuration Wizard initializes the `boot.properties` file and the `startup.properties` file for an Administration Server when you create the domain.

Any server instance started by Node Manager encrypts and saves the credentials with which it started in a server-specific `boot.properties` file, for use in automatic restarts.

Generating and Distributing Key Value Pairs

The script-based Node Manager uses two types of key value pairs. This section contains instructions for distributing key value pairs to the machines that will host a Node Manager client or server.

- "Shared Key Value Pair" on page 5-4.
- "Individual Key Value Pairs" on page 5-5

Shared Key Value Pair

This option distributes the same key value pair to all machines that will host a Node Manager client or server.

The simplest way to accomplish this is to set up your LAN to mount the Node Manager user home directory on each of the machines. This makes the key value pair available to the machines. Otherwise

1. Generate an RSA key value pair for the user with the `ssh-keygen` command provided with your SSH installation.

   The default location for the private and public keys are `~/.ssh/id_rsa` and `~/.ssh/id_rsa.pub` respectively.

If these keys are stored in a different location, modify the `ShellCommand` template, adding an option to the `ssh` command to specify the location of the keys.

2. Append the public key to the `~/.ssh/authorized_keys file` on the Node Manager machine. For example:

   ```
   command="/home/bea/server90/common/nodemanager/nodemanager.sh" 1024 33
   23...2323
   ```

   in which the you substitute the public key that you generated, as stored in `id_rsa.pub`, for the string shown in the example as

   ```
   1024 33 23...2323
   ```

   Note: The prefix `command=<command>` ensures that a user that establishes a session with the machine using the public key can only run the command specified—`nodemanager.sh`. This ensures that the user can only perform Node Manager functions, and prevents unauthorized access to data, system utilities, or other resources on the machine.

3. Manually distribute the key value pair to each machine that will host a Node Manager server instance or client.

4. Execute the following command on the client machine to check that the Node Manager client can access the Node Manager:

   ```
   /home/bea$ ssh montgomery VERSION
   ```

   This response indicates that the client accessed Node Manager successfully:

   ```
   +OK NodeManager v9.1.0
   ```

Individual Key Value Pairs

On each machine that will host a Node Manager client:

1. Generate a separate RSA key value pair for the Node Manager user as described in step one in the previous section.

Append the public key to the machine's `~/.ssh/authorized_keys` file user as described in step two in the previous section.

CHAPTER 6

Using Node Manager

This chapter provides information on how to start and stop the Java-based and script-based Node Manager. It also provides information on the recommended procedures for starting servers using Node Manager. The following topics are covered:

Starting Node Manager

Running Node Manager as a Startup Service

It is recommended that you install Node Manager to run as a startup service. This allows Node Manager to start up automatically each time the system is restarted.

By default, Node Manager listens only from the local host. If you want Node Manager to accept commands from remote systems, you must uninstall the default Node Manager service, then reinstall it to listen on a non-localhost Listen Address.

Starting Java-based Node Manager Using Scripts

Although running Node Manager as an operating system service is recommended, you can also start Node Manager manually at the command prompt or with a script. The environment variables Node Manager requires are described in "Step 8: Set the Node Manager Environment Variables" on page 3-7.

Sample start scripts for Node Manager are installed in the *WL_HOME*\server\bin directory, where *WL_HOME* is the top-level installation directory for WebLogic Server. Use startNodeManager.cmd on Windows systems and startNodeManager.sh on UNIX systems.

The scripts set the required environment variables and start Node Manager in *WL_HOME*/common/nodemanager. Node Manager uses this directory as a working directory for output and log files. To specify a different working directory, edit the start script with a text editor and set the value of the NODEMGR_HOME variable to the desired directory.

Edit the sample start script to make sure that the command qualifiers set the correct listen address and port number for your Node Manager process.

Command Syntax for Starting Java-based Node Manager

The syntax for starting Java-based Node Manager is:

```
java [java_option=value ...] -D[nodemanager_property=value]
-D[server_property=value] weblogic.NodeManager
```

where:

- *java_option* is a direct argument to the java executable, such as -ms or -mx.

  Note: If you did not set the CLASSPATH environment variable, use the -classpath option to identify required Node Manager classes.

- *nodemanager_property* is a Node Manager property. Instead of supplying Node Manager property values on the command line, you can edit the nodemanager.properties file, which is installed in the directory where you start Node Manager. For more information, see "Reviewing nodemanager.properties" on page 4-3.

  Node Manager property values you supply on the command line override the values in nodemanager.properties.

- *server_property* is a server-level property that Node Manager accepts on the command line, including:
  - bea.home—the BEA home directory that server instances on the current machine use.
  - java.security.policy— path to the security policy file that server instances on the current machine use.

Notes: For UNIX systems:

If you run Node Manager on a UNIX operating system other than Solaris or HP UX, you cannot have any white space characters in any of the parameters that will be passed to the

`java` command line when starting Node Manager. For example, this command fails due to the space character in the name "`big iron`".

```
-Dweblogic.Name="big iron"
```

For UNIX systems other than Solaris, HP-UX, and Linux operating systems, you must disable the `weblogic.nodemanager.nativeVersionEnabled` option at the command line when starting Node Manager (or set the property in `nodemanager.properties`) to use the pure Java version. For more information, see "Reviewing nodemanager.properties" on page 4-3.

Running Script-based Node Manager

To use the SSH Node Manager Command Shell, start the Administration Server using the following command line option:

```
-Dweblogic.nodemanager.ShellCommand='ssh -o PasswordAuthentication=no %H
wlscontrol.sh -d %D -r %R -s %S -x -c -f sample_custom_startscript.sh %C'
```

Note: `%C` must be the last argument supplied to wlscontrol.sh.

The `weblogic.nodemanager.ShellCommand` attribute specifies the command template to use to communicate with a remote SSH Node Manager and execute Node Manager functions for server instances under its control.

The template assumes that `wlscontrol.sh` is in the default path on the remote machine hosting Node Manager.

The `ShellCommand` syntax is:

```
ssh -o PasswordAuthentication=no %H wlscontrol.sh -d %D -r %R -s %S %C'
```

The possible command line options are listed in Table . The possible parameter values are listed in Table .

For example, if you type this command,

```
ssh -o PasswordAuthentication=no wlscontrol.sh myserver start
```

The listen address and port of the SSH server default to the listen address and port used by Node Manager on the remote machine. The domain name and domain directory are assumed to be the root directory specified for the target server instance, `myserver`.

This command:

```
ssh -o PasswordAuthentication=no 172.11.111.11 wlscontrol.sh -d
ProductionDomain -r ProductionDomain -s ServerA'
```

issues a START command to the server instance named ServerA, in the domain called ProductionDomain, located in the domains/ProductionDomain directory.

The ssh command must include the string:

```
-o PasswordAuthentication=no
```

This string passes the ssh PasswordAuthentication option. A value of yes causes the client to hang when it tries to read from the console.

Table 6-1 wlscontrol.sh Command Line Options

Parameter	Description
-n	Specifies the Node Manager root directory
-s	Specifies the server name
-r	Specifies the domain directory
-x	Sets the Node Manager debug flag.
-c	Enables a server start script.
-f	The name of the server start script.
-p	The name of the server stop script.
-h	Prints the usage for wlscontrol.sh.

Table 6-2 Shell Command Templates

Parameter	Description	Default
%H	Host name of the SSH server	NodeManagerMBean.ListenAddress
%N	Node Manager home directory	NodeManagerMbean.NodeManagerHome
%P	Port number of SSH server	NodeManagerMBean.ListenAddress 22
%S	WebLogic server name	none

Table 6-2 Shell Command Templates

%D	WebLogic domain name	`ServerStartMBean.RootDirectory`
%R	Domain directory (server root)	`ServerStartMBean.RootDirectory`
%C	Node manager script command • `START`—Start server • `KILL`—Kill server • `STAT`—Get server status • `GETLOG`—Retrieve server output log. • `VERSION`—Return Node Manager version. **Note:** This must be the last element in the command.	`none`

Stopping Node Manager

To stop Node Manager, close the command shell in which it is running.

Using Node Manager to Control Servers

This section describes the recommended procedures for starting servers using Node Manager and WLST.

Note: In general, it is recommended that you use the WebLogic Scripting Tool and Node Manager to start and stop the Administration Server and managed servers.For more information, see "Node Manager Commands" in *WebLogic Scripting Tool*

Starting the Administration Server Using Node Manager

The following general procedures are recommended for starting an Administration Server using WLST and Node Manager.

1. Connect to WLST using the `connect` command.
2. Connect to Node Manager using the `nmConnect` command.
3. Start the Administration Server using the `nmStart` command.

After the Administration Server has been started, you can use WLST to start the managed servers in your domain.

Note: Starting the server using the nmStart command allows Node Manager to monitor the state of your Administration Server and restart it in case of failure. Node Manager can only restart servers that were started in this way.

Starting Managed Servers

The following general procedures are recommended for starting a managed server using WLST and Node Manager.

1. Connect to WLST using the `connect` command.
2. Start you managed server using the WLST `start` command.

Using the start command causes WLST to contact the Administration Server to determine the managed servers startup properties. These are in turn passed to Node Manager and are used to start the managed server.

Note: Using nmStart allows you to pass specific properties to a server, but should only be used for debugging. Server properties passed through nmStart are not preserved the next time the server is restarted.

Starting Managed Servers without an Administration Server

The following general procedures are recommended for starting a managed server using WLST and Node Manager if you do not want to use the Administration Server to determine a managed server's startup properties:

1. Connect to WLST using the `connect` command.
2. Connect to Node Manager using the `nmConnect` command.
3. Start the managed server using the `nmStart` command.

Using the nmStart command allows you to restart a managed server without the Administration Server and to specify the server startup properties you want. However, the following considerations apply:

- If this is the first time you are starting the managed server, you must manually ensure that `boot.properties` and `startup.properties` are already defined.

- `nmStart` should not be used to permanently change the startup properties for a server. The next time a server is migrated or restarted from the Administration Server, these properties will not be used.
- When passing the server username and password via `nmStart`, these values are not encrypted.