Oracle® Fusion Middleware

Administering Node Manager for Oracle WebLogic Server

12*c* (12.2.1)

E55170-02

February 2016

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

ORACLE

Oracle Fusion Middleware Administering Node Manager for Oracle WebLogic Server, 12*c* (12.2.1)

E55170-02

Contents

Preface

This preface describes the document accessibility features and conventions used in this guide—*Administering Node Manager for Oracle WebLogic Server*.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1 Introduction and Roadmap

This chapter describes the contents and organization of this guide—*Administering Node Manager for Oracle WebLogic Server*.

This chapter includes the following sections:

1.1 Document Scope and Audience

This document describes how to configure and use Node Manager to control and manage server instances 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.

1.2 Guide to This Document

The document is organized as follows:

- This chapter, Chapter 1, "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 domain. It also provides detailed descriptions of the configuration and log files used by Node Manager.
- Chapter 3, "Node Manager Tutorial" provides a basic example of creating and using per domain Java-based Node Manager to start and stop WebLogic Server instances in a single-machine domain that hosts both an Administration Server and a Managed Server.
- Chapter 4, "Configuring Java Node Manager" describes the configuration procedures for the Java implementation of Node Manager.
- Chapter 5, "Configuring Script-Based Node Manager" describes the configuration procedures for the script-based implementation of Node Manager.

- Chapter 6, "Using Node Manager" provides procedures for starting Node Manager and server instances.

1.3 Related Documentation

- *Creating WebLogic Domains Using the Configuration Wizard*
- *Understanding Domain Configuration for Oracle WebLogic Server*
- *Oracle WebLogic Server Administration Console Online Help*
- *Administering Server Startup and Shutdown for Oracle WebLogic Server*

1.4 New and Changed Features in This Release

For a comprehensive listing of the new WebLogic Server features introduced in this release, see *What's New in Oracle WebLogic Server 12.2.1*.

2

Node Manager Overview

This chapter provides an introduction to Node Manager, a WebLogic Server utility. It also describes how Node Manager controls Administration Servers and Managed Servers.

This chapter includes the following sections:

2.1 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, Oracle recommends using it if your WebLogic Server environment hosts applications with high availability requirements because Node Manager allows you to control the running state of distributed server instances from a centralized location.

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. For more information, see Section 6.1.1, "Running Node Manager as a Startup Service."

For a basic tutorial demonstrating how to create a default per domain Node Manager instance in a single-machine domain to start and stop Managed Server, see Chapter 3, "Node Manager Tutorial."

2.2 What You Can Do with Node Manager

The following sections describe basic Node Manager functionality.

2.2.1 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 a Node Manager process on the machine that hosts the Administration Server and issue commands to start, shut down, or restart an Administration 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 WebLogic Server Administration Console, you are accessing Node Manager using 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 or a start script, the Managed Server contacts the Administration Server to obtain pending configuration updates.

2.2.2 Start, Shut Down, Suspend, and Restart Managed Servers

From the WebLogic Server Scripting Tool (WLST) command line, WebLogic Server Administration Console, 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: If using the `pack` and `unpack` commands, Node Manager can start a Managed Server for the first time in MSI mode. However, if using `nmEnroll`, Node Manager cannot start a Managed Server for the first time in MSI mode, because 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 *Command Reference for Oracle WebLogic Server*.

2.2.3 Restart Administration and Managed Servers Automatically

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

Note: Node Manager can only restart a server instance that was started using 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 a 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 needed.

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

2.2.4 Monitor Servers and View Log Data

Node Manager creates a log file for a Node Manager process and a log file of server output for each server instance it controls. For more information, see Section 2.7.2, "Log Files".

2.3 Node Manager Implementations

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

2.3.1 Java-based Node Manager

Java-based Node Manager runs within a Java Virtual Machine (JVM) process. Oracle recommends that you run it as a Windows service on Windows platforms and as an operating system service on UNIX platforms, allowing it to restart automatically when the system is rebooted. You can configure Java-based Node Manager using the Configuration Wizard or WLST offline.

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

> **Note:** Node Manager is not supported on OpenVMS, OS/390, AS400, UnixWare, or Tru64 UNIX.

This implementation of Node Manager determines its configuration from the `nodemanager.properties` file. See Section 4.8.4, "Reviewing nodemanager.properties".

Java-based Node Manager provides a more fine-grained security model, and the administrator credentials for accessing Node Manager are separate from those of the domain administrator. See Section 4.7, "Configuring Java-based Node Manager Security".

2.3.2 Script-based Node Manager

For UNIX and Linux systems, WebLogic Server provides a script-based implementation of Node Manager. This script is based on UNIX shell scripts.

For information on configuring the script implementation of Node Manager, see Section 5, "Configuring Script-Based Node Manager."

Script-based Node Manager is not recommended for production environments. However, depending on the security requirements for the environment in which you

are using Node Manager, the script-based implementation may be acceptable. The advantage of the script-based Node Manager is that it can remotely manage server instances 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: Oracle recommends that you run script-based Node Manager as an operating system service, which allows it to restart automatically when the system is rebooted.

2.3.3 Determining Which Node Manager Implementation to Use

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

- If you are installing WebLogic Server on a Windows system, you must use the Java implementation of Node Manager. The scripted implementation of Node Manager is not supported on Windows.
- Java-based Node Manager can be configured at the time you create the domain. Script-based Node Manager is configured after the domain has been created
- To use consensus leasing, you may see faster performance when using the Java implementation of Node Manager.
- The script-based Node Manager requires a much simpler security configuration than the Java implementation. RSH and SSH are generally easier to configure than SSL, which is the only way to secure Java-based Node Manager. The script implementation of Node Manager also requires a smaller footprint than the Java implementation.
- The Java implementation of Node Manager can be used in conjunction with `inetd` on supported UNIX systems. The `inetd` daemon allows Node Manager to be automatically restarted upon receiving a request on the configured port. You can also install the Java implementation of Node Manager as a Windows service.

2.4 Accessing Node Manager

You access Node Manager—the Java-based implementation or the script-based implementation— from either the WebLogic Server Administration Console or from WLST. In addition, you can access script-based Node Manager from a provided shell command template. You can also use JMX to communicate with the Administration Server.

- WebLogic Server Administration Console—Use the **Environments** > **Machines** > **Configuration** > **Node Manager** page.
- 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. For more information on using WLST and Node Manager to control server instances, see Section 6.2, "Using Node Manager to Control Servers."

2.5 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 server instances:

- Section 2.5.1, "Diagram of Node Manager and Servers"
- Section 2.5.2, "How Node Manager Starts an Administration Server"
- Section 2.5.3, "How Node Manager Starts a Managed Server"
- Section 2.5.4, "How Node Manager Restarts an Administration Server"
- Section 2.5.5, "How Node Manager Restarts a Managed Server"
- Section 2.5.6, "How Node Manager Shuts Down a Server Instance"

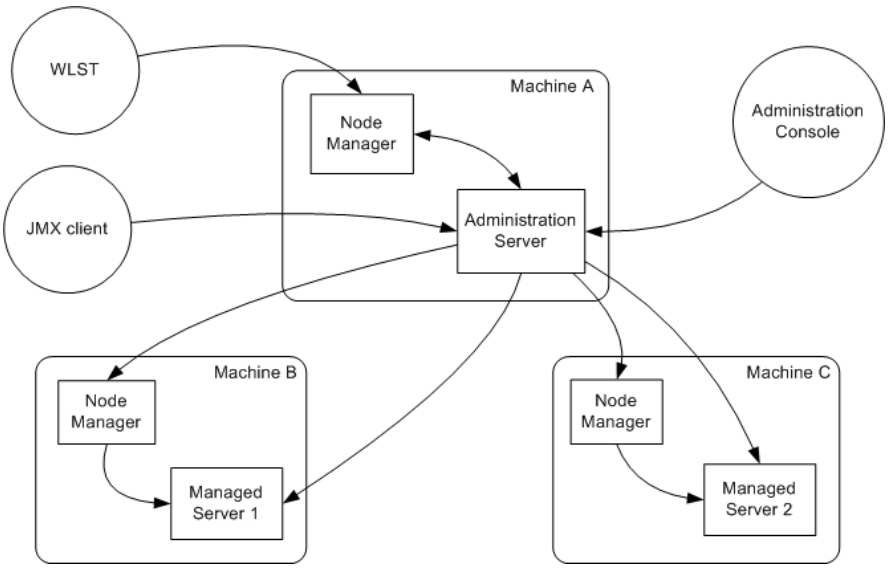
2.5.1 Diagram of Node Manager and Servers

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

In this diagram, Machine A hosts the Administration Server, and Machine B and Machine C host Managed Servers. Each machine contains a Node Manager instance. Using Node Manager, you can start, monitor and restart the Administration Server in Machine A. By using the Administration Server as a Node Manager client, you can start or stop Managed Servers in Machine B and Machine C.

You can use the WebLogic Server Administration Console, WLST, or a JMX client to access Node Manager. You can use WLST commands to start, stop, and monitor a server instance when WLST is either connected directly to the Node Manager instance, or when WLST is connected to the Administration Server. When using the WebLogic Server Administration Console, you access Node Manager using the Administration Server. You can also use a JMX client to communicate with the Administration Server.

Figure 2–1 Node Manager in the WebLogic Server Environment



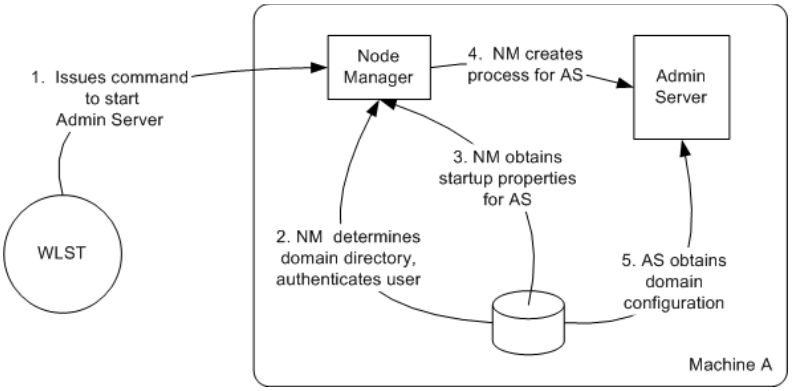
2.5.2 How Node Manager Starts an Administration Server

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 standalone 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 a Node Manager process on the machine that hosts the Administration Server. The `nmConnect` command provides a Node Manager user name and password that are used to authenticate the user with Node Manager.

   Note: If a Node Manager instance is the script-based implementation, the user can connect using the SSH client.

   Then, the user issues the `nmStart` command and provides the credentials for starting the Administration Server. For example:

   ```
   prps = makePropertiesObject(Username=username;Password=password")
   nmStart("AdminServer",props=prps)
   ```

   Note: A `boot.properties` file is generated if the user has already started the Administration Server and provided credentials.

   The `nmStart` command identifies the server instance to start.

2. Node Manager looks up the domain directory in `nodemanager.domains`, and authenticates the user credentials using a local file that contains the encrypted user name and password.

3. Node Manager obtains the startup properties for the Administration Server.

   The `nmStart` command can optionally pass properties that are used to start the Managed Server. When these properties are passed, they overwrite any previously stored properties that Node Manager may have used. If no properties are passed with the `nmStart` command, then Node Manager uses the values in the

`startup.properties` file that have been persisted from a previous startup or from using the `nmGenBootStartupProps` WLST command.

4. Node Manager creates the Administration Server process.
5. The Administration Server obtains the domain configuration from its `config` directory.

Note: After the Administration Server is running, you can update the user credentials and startup properties using the WLST online command, `nmGenBootStartupProps`.

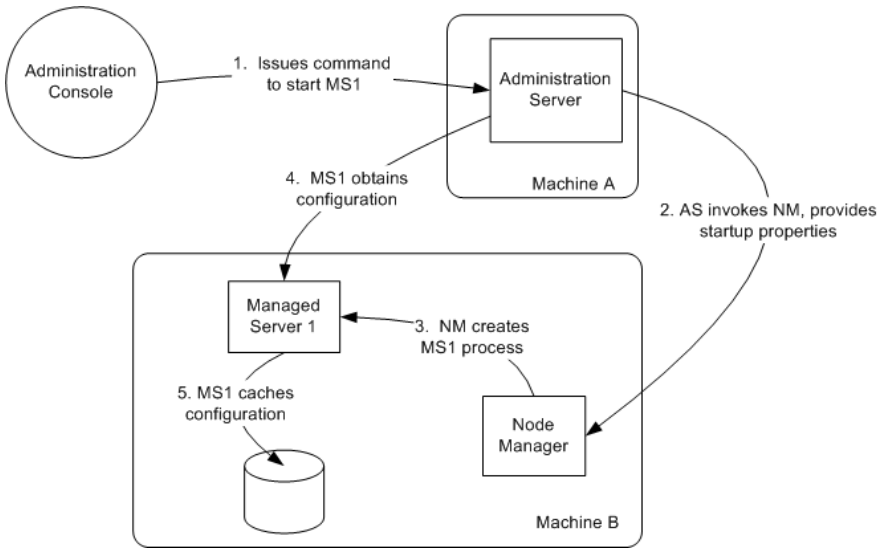
Alternatively, when the Administration Server and Node Manager are running, you can update the user credentials and startup properties in the WebLogic Server Administration Console, on the *AdminServer* > **Configuration > Server Start** page. The Administration Server then pushes the updates to the running Node Manager and Node Manager writes the information to the disk.

2.5.3 How Node Manager Starts a Managed Server

Figure 2–3 illustrates the process of starting a Managed Server with Node Manager using the Administration Console. You can also use WLST or a JMX client to connect to the Administration Server.

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 WebLogic Server Administration Console, the user issues a start command for Managed Server 1.
2. The Administration Server issues a start command for Managed Server 1 to 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 Section 4.8.5, "Configuring Remote Startup Arguments.".
3. Node Manager starts Managed Server 1.

Node Manager starts the Managed Server in the domain directory.

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.

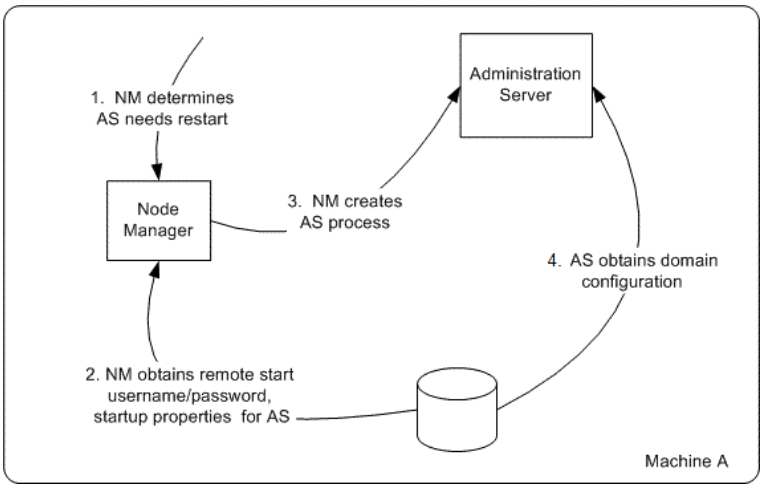
2.5.4 How Node Manager Restarts an Administration Server

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. However, the `CrashRecoveryEnabled` property takes precedence over the `AutoRestart` property in a crash recovery scenario. For example, if a server instance has `AutoRestart=false` but `CrashRecoveryEnabled=true`, when Node Manager restarts, Node Manager tries to recover the server instance if the server instance failed when Node Manager was not running.

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 user name 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.

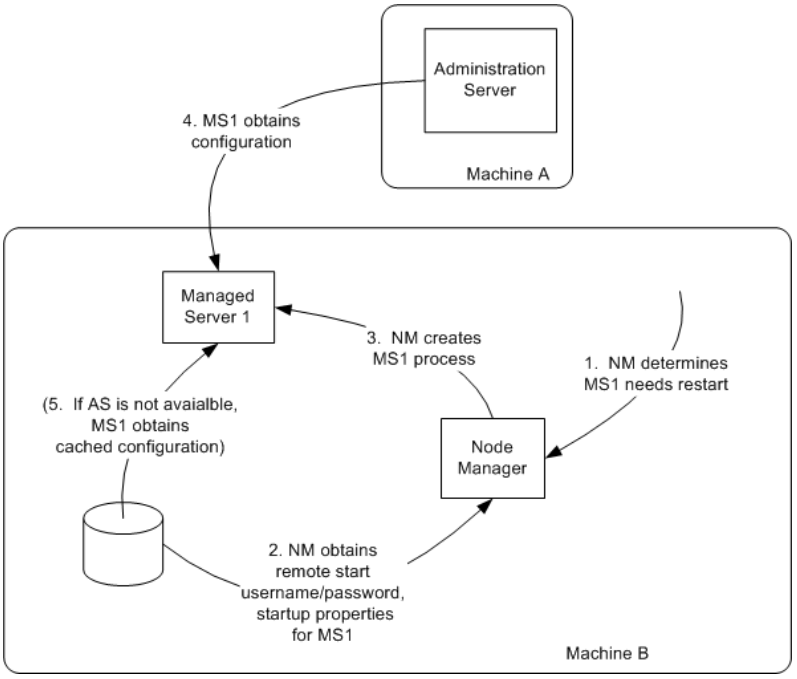
2.5.5 How Node Manager Restarts a Managed Server

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. However, the `CrashRecoveryEnabled` property takes precedence over the `AutoRestart` property in a crash recovery scenario. For example, if a server instance has `AutoRestart=false` but `CrashRecoveryEnabled=true`, when Node Manager restarts, Node Manager tries to recover the server instance if the server instance failed when Node Manager was not running.

Figure 2–5 Restarting a Managed Server



1. Node Manager determines from the Managed Server 1 process exit code that it requires restart.
2. Node Manager obtains the user name 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_name`*`/data/nodemanager/` 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 (MSI) mode is enabled by default. MSI mode is also enabled when:

- an `admin_url` is not provided, where `admin_url` specifies the listen address (host name, IP address, or DNS name) and port number of the domain's Administration Server.
- an `admin_url` is provided, but the Administration Server cannot be reached for any reason.

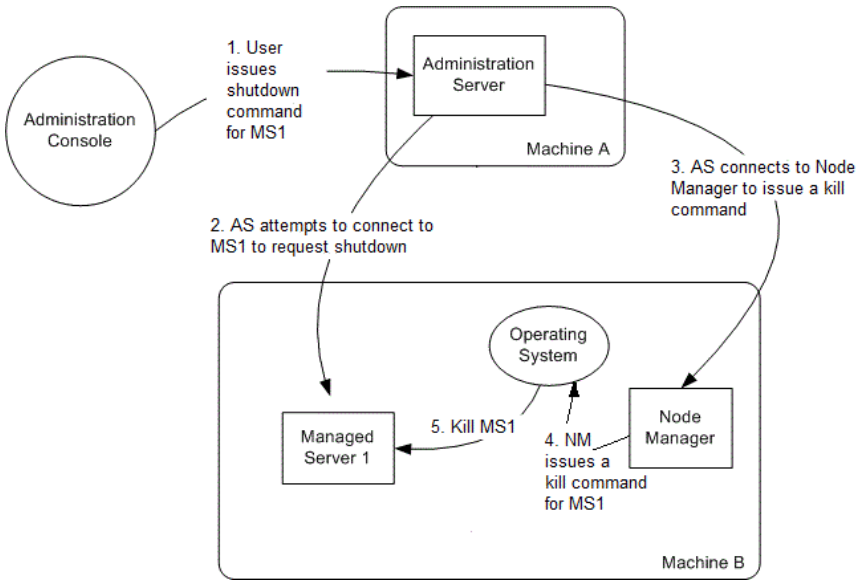
MSI mode can be temporary as the Managed Server intermittently attempts to reconnect to the Administration Server.

2.5.6 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 forcefully shut down the Managed Server. Node Manager cannot gracefully shut down a Managed Server.

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 WebLogic Server Administration Console, an authorized user issues a shutdown command for Managed Server 1.
2. The Administration Server attempts to connect to Managed Server 1 and issues the shutdown command directly to Managed Server 1. If the Administration Server

successfully contacts Managed Server 1, Managed Server 1 performs the shutdown sequence described in "Shutting Down Instances of WebLogic Server" in *Administering Server Startup and Shutdown for Oracle WebLogic Server*. For the Managed Server to gracefully shut down itself, it must be connected to the Administration Server.

3. If, in the previous step, the Administration Server fails to contact Managed Server 1, the Administration Server connects to Node Manager to issue a kill command for Managed Server 1.
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.

2.6 Node Manager and System Crash Recovery

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

- For Java-based Node Manager, ensure that `CrashRecoveryEnabled` is set to `true`.

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

  > **Note:** The `CrashRecoveryEnabled` configuration property takes precedence over the `AutoRestart` server startup property in a crash recovery scenario. For example, if a server instance has `AutoRestart=false` but `CrashRecoveryEnabled=true`, when Node Manager restarts, Node Manager tries to recover the server instance if the server instance failed when Node Manager was not running.

- For script-based Node Manager, place this line in machine start scripts or, if desired, run periodically on a given schedule:

  ```
  wlscontrol.sh -d domain_name CRASHRECOVERY
  ```

- You should start the Administration Server using Node Manager.
- All Managed Servers should be started using the Administration Server. You can accomplish this using WLST or the WebLogic Server 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 instance.

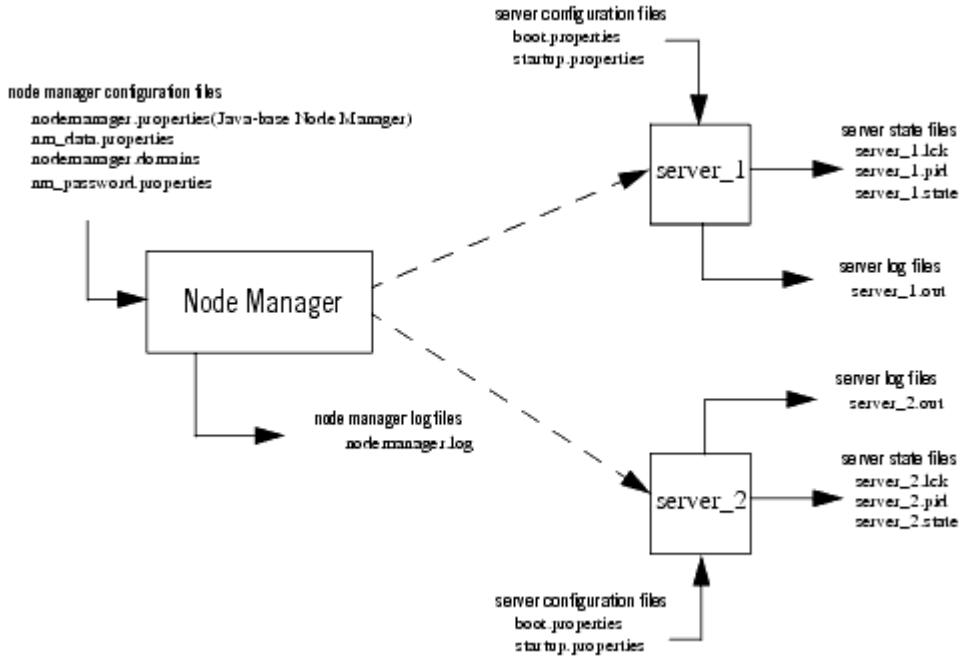
If the process is running, Node Manager performs an additional check to access the management servlet running in the process to verify that the process corresponding to the process ID is a WebLogic Server instance.

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

2.7 Node Manager Configuration and Log Files

In managing multiple server instances, Node Manager uses multiple configuration files and outputs log files to multiple directories, as shown in Figure 2–7.

Figure 2–7 Node Manager Configuration and Logging Environment



The following sections describe Node Manager configuration and log files:

- Section 2.7.1, "Configuration Files"
- Section 2.7.2, "Log Files"

2.7.1 Configuration Files

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

2.7.1.1 nodemanager.properties

This is the configuration file used by the Java-based implementation of Node Manager. See Section 4.8.4, "Reviewing nodemanager.properties".

By default, this file is located in *NodeManagerHome*, typically, *ORACLE_HOME*\user_projects\domains*domain_name*\nodemanager, where *ORACLE_HOME* is the location you specified as Oracle Home when you installed WebLogic Server.

2.7.1.2 nodemanager.domains

This file contains mappings between the names of domains managed by Node Manager and their corresponding directories. See Section 4.8.3, "Configuring nodemanager.domains File."

For the Java-based Node Manager, this file is located in *NodeManagerHome*, typically, *ORACLE_HOME*\user_projects\domains*domain_name*\nodemanager.

For the script-based Node Manager, this file's default *NodeManagerHome* location is *WL_HOME*/common/nodemanager, where *WL_HOME* is the location in which you installed WebLogic Server, for example, *ORACLE_HOME*/wlserver.

2.7.1.3 nm_password.properties

This file stores the Node Manager user name and password. See Section 4.7.1, "Specifying Node Manager User Name and Password."

This file is located in *DOMAIN_HOME*/config/nodemanager, where *DOMAIN_HOME* is the location of your WebLogic domain, typically, *ORACLE_HOME*\user_projects\domains*domain_name*.

2.7.1.4 boot.properties

Node Manager uses this file to specify user credentials when starting a server instance.

This file is located in *DOMAIN_HOME*/servers/*server_name*/data/nodemanager.

2.7.1.5 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 instance. Node Manager automatically creates this file by using properties passed to Node Manager when the Administration Server was last used to start the server instance. This allows a Node Manager client or startup scripts to restart a Managed Server using the same properties last used by the Administration Server.

For more information on startup.properties, see Section 4.8.6, "Setting Server Startup Properties." These properties correspond to the server startup attributes contained in ServerStartMBean and the health monitoring attributes in ServerStartMBean.

This file is located in *DOMAIN_HOME*/servers/*server_name*/data/nodemanager.

2.7.1.6 *server_name*.addr

server_name.addr stores the IP address added when a server instance 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 instance.

This file is located in *DOMAIN_HOME*/servers/*server_name*/data/nodemanager.

2.7.1.7 *server_name*.lck

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

This file is located in *DOMAIN_HOME*/servers/*server_name*/data/nodemanager.

2.7.1.8 *server_name*.pid

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

This file is located in *DOMAIN_HOME*/servers/*server_name*/data/nodemanager.

2.7.1.9 *server_name*.state

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

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

This file is located in *DOMAIN_HOME*/servers/*server_name*/data/nodemanager.

2.7.2 Log Files

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

Table 2–1 Node Manager Log File Locations

Log File	Location
Node Manager Log File	For Java-based Node Manager only, *NodeManagerHome*/nodemanager.log, where *NodeManagerHome* typically is *ORACLE_HOME*\user_projects\domains*domain_name*\nodemanager.
Node Manager Server Instance Log Files	*DOMAIN_HOME*/servers/*server_name*/logs/*server_name*.out, where *DOMAIN_HOME* is the location in which you installed your WebLogic domain, such as *ORACLE_HOME*\user_projects\domains*domain_name*.
WebLogic Server Log Files	*DOMAIN_HOME*/servers/*server_name*/logs/*server_name*.log

2.7.2.1 nodemanager.log

nodemanager.log is created for the Java-based Node Manager only; it is not created for the script-based Node Manager. This log file is generated by Node Manager and contains data for a given WebLogic domain that is controlled by Node Manager. The file typically is located in *ORACLE_HOME*\user_projects\domains*domain_name*\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 a Node Manager log file by:

- Selecting **Machines** > **Monitoring** > **Node Manager Log** page in the WebLogic Server Administration Console
- Using the WLST nmLog command

2.7.2.2 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 debug information in the server output log information. By default, this file is located in *DOMAIN_HOME*/servers/*server_name*/logs, where *server_name* is the name of the server instance.

However, if you add `-Dweblogic.Stdout=` in the `Arguments` field of the `ServerStartMBean`, then this value overrides the default location, and Node Manager uses the supplied path as the file for the output. You cannot override the `server_name.out` default location using the `weblogic.startup.Arguments.prepend` property in the `nodemanager.properties` file, as this property applies to multiple server instances.

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

server_name`.out`

You can view a 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.

2.7.2.3 Log File Rotation

Prior to WebLogic Server 12.1.3.0, you could configure the Node Manager log file rotation properties listed in Table 2–2 when `NativeVersionEnabled=false`. As of WebLogic Server 12.1.3.0, Node Manager uses the `LogFileMBean` properties for log file rotation, and the server output files created by Node Manager are rotated regardless of the `NativeVersionEnabled` setting.

For Coherence and other system components that have implemented a plug-in, Node Manager uses the log file rotation properties listed in Table 2–3.

Table 2–2 Node Manager Log File Rotation Properties

Property	Description	Default
`FileSizeKB`	This property is deprecated as of WebLogic Server 12.1.3.0 and may be removed in a future release. Use the server `LogFileMBean` to adjust log file rotation values or `process.FileSizeKB` in Table 2–3 for Coherence and system components.	`500`
`FileTimeSpan`	This property is deprecated as of WebLogic Server 12.1.3.0 and may be removed in a future release. Use the server `LogFileMBean` to adjust log file rotation values or `process.FileTimeSpan` in Table 2–3 for Coherence and system components.	`24`
`FileTimeSpanFactor`	This property is deprecated as of WebLogic Server 12.1.3.0 and may be removed in a future release. Use the server `LogFileMBean` to adjust log file rotation values or `process.FileTimeSpanFactor` in Table 2–3 for Coherence and system components.	`360000`
`NumberOfFilesLimited`	This property is deprecated as of WebLogic Server 12.1.3.0 and may be removed in a future release. Use the server `LogFileMBean` to adjust log file rotation values or `process.NumberOfFilesLimited` in Table 2–3 for Coherence and system components.	`true`

Table 2–2 (Cont.) Node Manager Log File Rotation Properties

Property	Description	Default
`RotatedFileCount`	This property is deprecated as of WebLogic Server 12.1.3.0 and may be removed in a future release. Use the server `LogFileMBean` to adjust log file rotation values or `process.RotatedFileCount` in Table 2–3 for Coherence and system components.	`7`
`RotationTimeStart`	This property is deprecated as of WebLogic Server 12.1.3.0 and may be removed in a future release. Use the server `LogFileMBean` to adjust log file rotation values or `process.RotationTimeStart` in Table 2–3 for Coherence and system components.	`00:00`
`RotationType`	This property is deprecated as of WebLogic Server 12.1.3.0 and may be removed in a future release. Use the server `LogFileMBean` to adjust log file rotation values or `process.RotationType` in Table 2–3 for Coherence and system components.	`SIZE`

Table 2–3 Node Manager Log File Rotation Properties for Coherence and System Components

Property	Description	Default
`process.FileSizeKB`	Specifies the maximum size of the file before it is rotated.	`500`
`process.FileTimeSpan`	The interval (in hours) at which the server instance saves old log messages to another file. Requires that you specify a `process.RotationType` of `TIME`.	`24`
`process.FileTimeSpanFactor`	Allows log rotation to be tested at a different frequency. **Note:** This property rarely needs changed or configured.	`360000`
`process.NumberOfFilesLimited`	Indicates whether to limit the number of log files that this server instance creates to store old messages. Requires that you specify a `process.RotationType` of `SIZE` or `TIME`. After the server instance reaches this limit, it deletes the oldest log file and creates a new log file with the latest suffix. If you do not enable this option, the server instance creates new files indefinitely and you must clean up these files as you require.	`true`

Table 2–3 (Cont.) Node Manager Log File Rotation Properties for Coherence and System Components

Property	Description	Default
`process.RotatedFileCount`	Specifies the maximum number of rotated files to keep when `process.NumberOfFilesLimited=true`.	`7`
`process.RotationTimeStart`	Determines the start time (hour and minute) for a time-based rotation sequence. At the time that this value specifies, the server instance renames the current log file. Thereafter, the server instance renames the log file at an interval that you specify in `process.FileTimeSpan`. Note that WebLogic Server sets a threshold size limit of 500 MB before it forces a hard rotation to prevent excessive log file growth. Use the following format: `H:mm`, where `H` is hour in day (0-23) and `mm` is the minute in hour.	`00:00`
`process.RotationType`	Specifies the criteria for moving old log messages to a separate file. ■ `NONE`: Messages accumulate in a single file. You must erase the contents of the file when the size is too large. Note that WebLogic Server sets a threshold size limit of 500 MB before it forces a hard rotation to prevent excessive log file growth. ■ `SIZE`: When the log file reaches the size that you specify in `FileMinSize`, the server instance renames the file as `SERVER_NAME.lognnnnn`. ■ `TIME`: At each time interval that you specify in `FileTimeSpan`, the server instance renames the file as `SERVER_NAME.lognnnnn`.	`SIZE`

3 Node Manager Tutorial

This chapter provides a basic example of using the default per domain Node Manager. This tutorial shows the steps to create and use Java-based Node Manager to start and stop WebLogic Server instances in a single-machine domain that hosts both an Administration Server and a Managed Server.

For more information about per domain Node Manager, see Section 4.2, "Default Node Manager Configuration."

This tutorial includes the following steps:

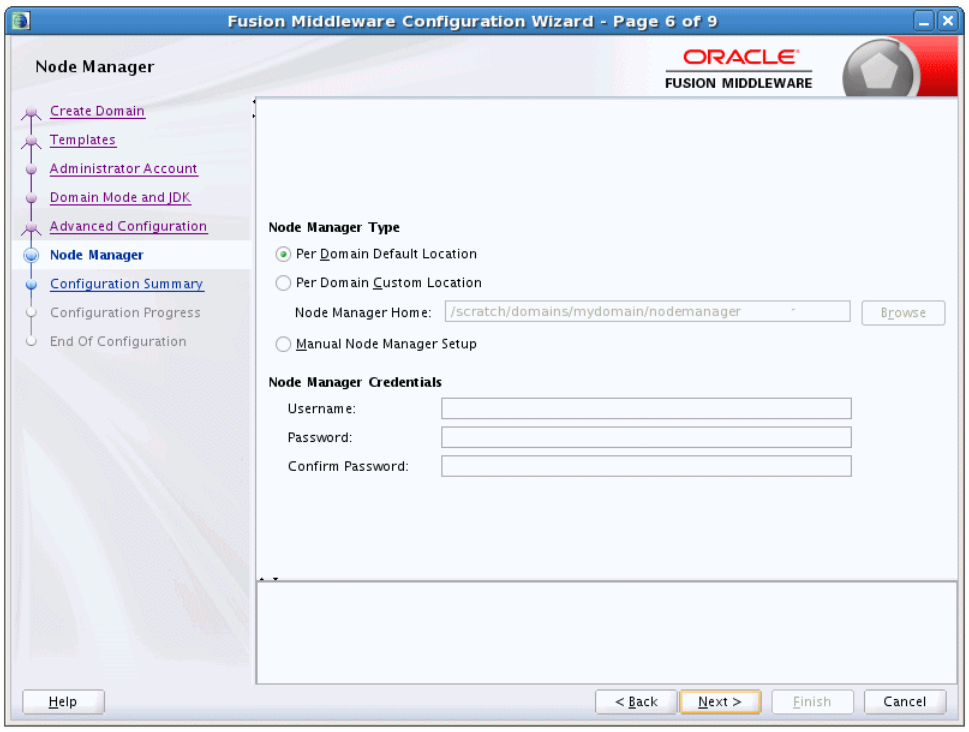
- Create Node Manager in a New Domain
- Start Node Manager
- Use Node Manager to Start a Managed Server

3.1 Create Node Manager in a New Domain

To create an instance of per domain, Java-based Node Manager in a new domain, complete the following steps:

1. Start the Configuration Wizard:
   a. From a command window, change to the *ORACLE_HOME*/oracle_common/common/bin directory, where ORACLE_HOME represents the directory you specified as the Oracle Home when you installed WebLogic Server.
   b. Execute the config.cmd command for Windows or the config.sh command for UNIX.
2. Use the Configuration Wizard to create a new, basic domain:
   a. On the Create Domain screen, select **Create a new domain** and select a domain directory location. This is your *DOMAIN_HOME* directory. Click **Next**.
   b. On the Templates screen, accept the default **Create Domain Using Product Templates**. Click **Next**.
   c. On the Administrator Account screen, enter the credentials of the WebLogic Server administrator. Click **Next**.
   d. On the Domain Mode and JDK screen, accept the default Domain Mode and JDK. Click **Next**.
3. Create a Node Manager instance, a Managed Server, and a machine in your domain:
   a. On the Advanced Configuration screen, select the **Node Manager** and **Managed Servers, Clusters and Coherence** options. Click **Next**.

b. On the Node Manager screen, accept **Per Domain Default Location** as the default Node Manager type and enter the credentials of the Node Manager administrator. Click **Next**.



c. On the Managed Servers screen, click **Add**. Accept the default server name, listen address, and listen port (7003). Click **Next**.

d. On the Clusters screen, click **Next**.

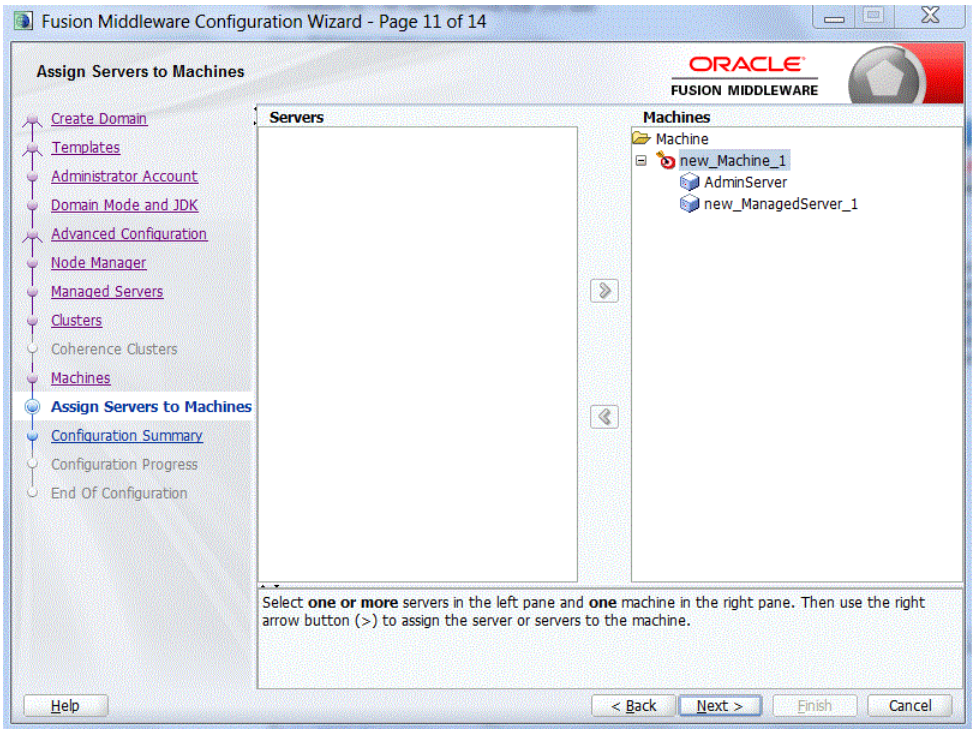
e. On the Machines screen, click **Add**. Accept the default machine name, Node Manager listen address, and Node Manager listen port. Click **Next**.

4. Assign the Administration Server and Managed Server to your machine:

a. On the Assign Servers to Machines screen, select your Administration Server and Managed Server in the left column.

b. Use the right arrow button to assign the Administration Server and Managed Server to your machine in the right column.

c. Click **Next**.



5. Complete your domain configuration:

   a. On the Configuration Summary screen, review your configuration and click **Create**.

   b. When the Configuration Process screen shows that your domain has been created successfully, click **Next**.

   c. On the Configuration Success screen, select **Start Admin Server** to start your domain. Click **Finish**.

3.2 Start Node Manager

To start Node Manager in your domain, complete the following steps:

1. From a command window, change to the *DOMAIN_HOME*`/bin` directory, where `DOMAIN_HOME` represents the directory in which your WebLogic Server domain is configured.
2. To set your environment, execute the `setDomainEnv.cmd` command for Windows or the `setDomainEnv.sh` command for UNIX.
3. Return to the *DOMAIN_HOME*`/bin` directory.
4. To start Node Manager in your domain, execute the `startNodeManager.cmd` command for Windows or the `startNodeManager.sh` command for UNIX.

   If Node Manager starts successfully, you will see output similar to the following:

```
<DATE TIME> <INFO> <12.2.1.0.0>
<DATE TIME> <INFO> <Secure socket listener started on port 5556, host
localhost/127.0.01>
```

3.3 Use Node Manager to Start a Managed Server

To start a Managed Server in your domain using Node Manager, complete the following steps:

1. Start the WebLogic Server Administration Console:
   a. Ensure the Administration Server is running.

      If you selected the **Start Admin Server** option in the Configuration Wizard, the Administration Server should be running.

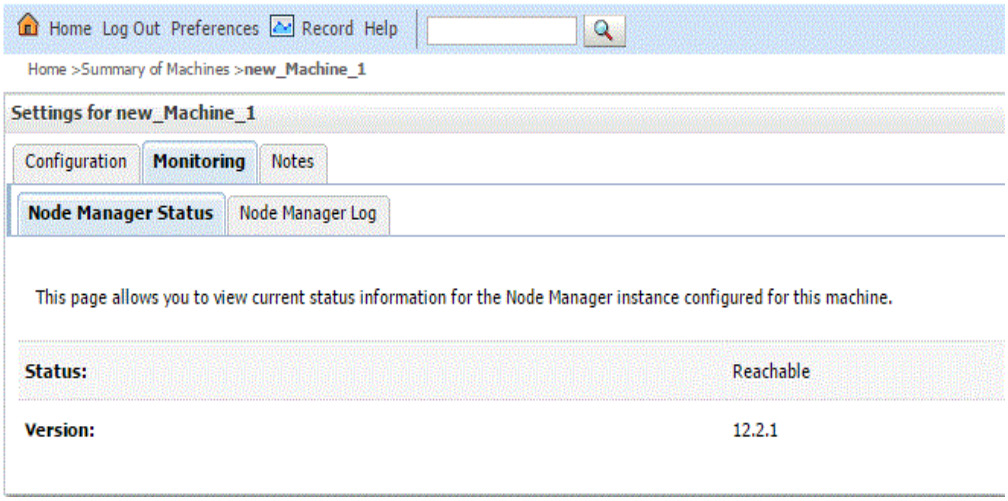
      If the Administration Server is not running, change to the *DOMAIN_HOME*/bin directory and execute the `startWebLogic.cmd` command for Windows or the `startWebLogic.sh` command for UNIX.

   b. Launch a Web browser and enter the following URL:

      `http://hostname:port/console`

      where *hostname* is the DNS name or IP address of the Administration Server and *port* is the listen port on which the Administration Server is listening for requests (port 7001 by default).

   c. When the login page appears, enter the user name and the password you entered when creating your domain in the Configuration Wizard.
2. Ensure Node Manager is running.
   a. In the left pane of the WebLogic Server Administration Console, select **Environment > Machines**.
   b. In the Machines table, select the name of your machine.
   c. Select **Monitoring > Node Manager Status**.
   d. If Node Manager is running, **Status** will be **Reachable**.

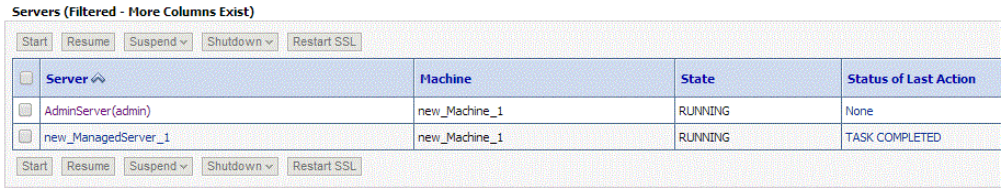


Note: To configure Node Manager settings in the WebLogic Server Administration Console, navigate to the following page: **Environment > Machines >** ***machine_name*** **> Configuration > Node Manager**.

3. In the left pane of the WebLogic Server Administration Console, select **Environment > Servers**.
4. Select the **Control** page.
5. In the Servers table, select the checkbox next to your Managed Server.
6. Click **Start**.

   A message appears confirming that Node Manager will start the selected server instance. Refresh the WebLogic Server Administration Console page, and the Managed Server state changes from `SHUTDOWN` to `RUNNING`.

   Your Managed Server is now running in your domain.



4

Configuring Java Node Manager

This chapter describes how to configure the Java implementation of Node Manager.

This chapter includes the following sections:

4.1 Overview

The Java implementation of Node Manager is configured by default to control all server instances belonging to the same domain, a *per domain* Node Manager. The server instances need not reside on the same machine. If a machine has multiple domains, using per domain Node Manager results in multiple Node Manager process instances. For more information about per domain Node Manager, see Section 4.2, "Default Node Manager Configuration."

In previous versions of WebLogic Server, Node Manager was not associated with a specific WebLogic domain but only with a host machine. You used the same Node Manager process to control server instances in any WebLogic domain, as long as the server instances resided on the same machine, a machine-scoped, a *per host* Node Manager. While you can still use per host Node Manager, additional configuration is required. Typically, per host Node Manager is used to manage multiple domains with a single Node Manager instance. For more information about per host Node Manager, see Section 4.3, "Configuring Per Host Node Manager."

If you are using per domain Node Manager, WebLogic Server provides the option to have the Node Manager configuration in a custom location. Using the Configuration Wizard or WLST offline, you can select a `PerDomain` or `CustomLocation` Java-based Node Manager configuration. For any type of Node Manager configuration, you can provide unique Node Manager credentials, but `NodeManagerHome` is the default location, as described in Section 4.2, "Default Node Manager Configuration." If you want a unique location for `NodeManagerHome`, select `CustomLocation` and specify an empty directory or select to create one.

If upgrading Node Manager from WebLogic Server 12.1.1 or earlier to the current version or when upgrading from WebLogic Server 12.1.2 or greater to the current version, see "Determining Node Manager Upgrade Procedure" in *Upgrading Oracle WebLogic Server*.

4.2 Default Node Manager Configuration

For each WebLogic Server domain you create, a domain-specific Node Manager instance is created by default. If you choose to use the default per domain Node Manager configuration, no additional steps are necessary to use Node Manager to start and stop server instances in your WebLogic Server domain.

Using the security credentials supplied for the Administration Server, `nm_password.properties` is created in *`DOMAIN_HOME`*`\config\nodemanager`, where *`DOMAIN_HOME`* is typically located at *`ORACLE_HOME`*`\user_projects\domains\`*`domain_name`*. The `nodemanager.properties` and `nodemanager.domains` files are created for you under *`DOMAIN_HOME`*`\nodemanager`. With the default Node Manager configuration, you cannot edit the *`NodeManagerHome`* location, *`DOMAIN_HOME`*`\nodemanager`

Domain-specific scripts to start, stop, install and uninstall Node Manager as a Windows service, are located under *`DOMAIN_HOME`*`\bin`. To install Node Manager as a Windows service, you may need to edit the `installNodeMgrSvc.cmd` script to specify appropriate listen address and listen port values:

1. Navigate to your *`DOMAIN_HOME`*`\bin` directory.
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.

Note: When configuring multiple per domain Node Manager instances on the same machine, you must use a unique Node Manager address (`hostname:port`) for each domain, either by using unique ports or unique host names. For example, if you have three per domain Node Manager instances running on the machine, use address `localhost:5556` for Domain 1, address `localhost:5557` for Domain 2, and `localhost:5558` for Domain 3.

4.3 Configuring Per Host Node Manager

If you want to use *per host* Node Manager, for which scripts are located in *`WL_HOME`*`\server\bin`, you must first perform the following prerequisite configuration steps:

Note: For per host Node Manager configurations, do not set `weblogic.RootDirectory` in `JAVA_OPTIONS` to the domain home. If `weblogic.RootDirectory` points to an existing domain, then the default location for the security subsystem is the domain instead of the Node Manager specific location. Node Manager may then use the domain-specific security settings by default, which could cause the SSL handshake to fail if a second domain attempts to communicate with Node Manager.

Additionally, if you want to use SSL for a per host Node Manager configuration, you must build your own certificate files.

1. Create a `nodemanager.domains` file that specifies the domains that you want this Node Manager instance to control, under *`ORACLE_HOME`*`\oracle_common\common\nodemanager`, the per host *`NodeManagerHome`* location.
   - You can manually create or copy this file. See Section 4.8.3, "Configuring nodemanager.domains File."
   - Alternatively, you can register WebLogic domains with Node Manager using the WLST command, `nmEnroll`.

   By specifying multiple domains in the `nodemanager.domains` file, you can configure a single, machine-scoped Node Manager process which manages server instances belonging to multiple WebLogic domains, similar to Node Manager functionality from prior WebLogic Server releases.
2. Configure a machine definition for each machine that runs a Node Manager process. For more information, see Section 6.2.3.1, "Configuring a Machine to Use Node Manager."
3. If you want to use the demonstration Identity and Trust keystores, recommended for development or testing purposes *only*, you can create them using the CertGen and ImportPrivateKey Java utilities as shown in the following examples:
   1. To properly set up the `PATH` and `CLASSPATH` variables, from a command prompt run *`WL_HOME`*`\server\bin\setWLSEnv.cmd`.

      Note: On UNIX operating systems, the `setWLSEnv.sh` command does not set the environment variables in all command shells. Oracle recommends that you execute this command using the Korn shell or bash shell.

   2. Generate a certificate and private key.

      ```
      java utils.CertGen -keyfilepass DemoIdentityPassPhrase -certfile democert
      -keyfile demokey
      ```

      By default `utils.CertGen` will use the short host name as the owner CN value in the generated certificate. To use the fully-qualified DN host name, add the `-cn` option to the above command. For example:

      ```
      java utils.CertGen -keyfilepass DemoIdentityPassPhrase -certfile democert
      -keyfile demokey -cn abc.oracle.com
      ```

   3. Import the private key and certificate.

      ```
      java utils.ImportPrivateKey -keystore DemoIdentity.jks -storepass
      ```

```
DemoIdentityKeyStorePassPhrase -keyfile demokey -keyfilepass
DemoIdentityPassPhrase -certfile democert.pem -keyfile demokey.pem -alias
demoidentity
```

The `DemoIdentity.jks` keystore now contains one private key and certificate entry. The other files can be deleted.

4. Copy the `DemoIdentity.jks` keystore to the *`NodeManagerHome`*`\security` directory.

For information about configuring SSL for Node Manager in production environments, see Section 4.7.3, "Using SSL With Java-based Node Manager."

For domains that include Oracle JRF, you can configure Node Manager to use the Oracle Platform Security Services Keystore Service (OPSS). See "Configuring Node Manager to Use the OPSS Keystore Service" in *Administering Oracle Fusion Middleware*.

Note: By default, using SSL with Node Manager is enabled. If not needed, you can disable it by changing to `SecureListener=false` in the `nodemanager.properties` file. To review the SSL-related properties in `nodemanager.properties`, see Table 4–1.

4.4 Configuring Node Manager on Multiple Machines

If you have a domain that has Managed Servers on multiple physical machines, perform the following steps:

1. Ensure Node Manager is installed and configured on each machine.

   You can use any type of Node Manager (per domain, per host, or custom) to configure Node Manager on multiple machines. However, you should use the same Node Manager type for all machines.

   Create the *`NodeManagerHome`* location and `nodemanager.properties` file. By default, *`NodeManagerHome`* is *`DOMAIN_HOME`*`\nodemanager`. In a production environment, you might want to customize the location of the Node Manager root directory. The `nodemanager.properties` file is created in the directory specified in *`NodeManagerHome`* and specifies Node Manager properties. See Section 4.8.4, "Reviewing nodemanager.properties."

   Note: The *`NodeManagerHome`* location should not be shared by multiple Node Manager instances.

   Therefore, in a scenario when you are using a shared drive, the *`NodeManagerHome`* location on each machine should be unique. For example, create a host name subdirectory under *`NodeManagerHome`*, and copy all files to the host name subdirectory.

   ---

2. To enroll a domain with Node Manager, Oracle recommends using the `pack` and `unpack` commands to copy all of the required domain and configuration information from one machine to another. See "Node Manager Configuration" in *Creating Templates and Domains Using the Pack and Unpack Commands*. You can also use the WLST command `nmEnroll` to perform this action.

3. Create the corresponding `MachineMBean` in the Administration Server that points to each Node Manager instance, so that you can start Managed Servers using Node Manager from any machine.

   When the Administration Server is involved in starting a Managed Server, it uses the information in the targeted `MachineMBean` to create a connection with the correct Node Manager instance. This occurs when using either the WebLogic Server Administration Console or WLST after running the `nmConnect` command. Creating a corresponding `MachineMBean` for each physical machine helps point to each Node Manager instance and associates your server instances with the appropriate `MachineMBean`.

   To create a `MachineMBean` in the Administration Server, perform one of the following actions:

   - In the WebLogic Server Administration Console, select **Environment** > **Machines** to create a new machine. Configure the information in the `MachineMBean` and its `NodeManagerMBean` to point to the Node Manager listen address and listen port. For more information, see "Create and configure machines" in *Oracle WebLogic Server Administration Console Online Help*.
   - Using WLST, create the `MachineMBean` with the create command. Then, change the settings of the `MachineMBean` to point to the Node Manager listen address and listen port.

4. Optionally, repeat any configuration steps you performed if you created Node Manager locally.

For more information, see Section 2.4, "Accessing Node Manager" and "`nmEnroll`" in *WLST Command Reference for WebLogic Server*.

4.5 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 and Node Manager log. For more information on WLST Node Manager commands, see "WLST Command and Variable Reference" in *WLST Command Reference for WebLogic Server*.

4.6 Configuring Node Manager Using WLST Offline

If needed, you can use WLST offline to perform the following Node Manager configuration tasks:

- Set the Node Manager user name and password
- Set Node Manager properties
- Set the Node Manager type

Example 4–1 shows how to set a domain's Node Manager listen address and listen port, the Node Manager user name and password, and the Node Manager type.

Note: If the Node Manager type is `ManualNodeManagerSetup`, you cannot use WLST offline to edit Node Manager properties.

Example 4–1 Configuring Node Manager

```
# set the Node Manager listen address and listen port.
cd('/')
cd('NMProperties')
set('ListenAddress','localhost')
set('ListenPort',9001)
# Set the Node Manager user name and password.
cd('/')
cd('SecurityConfiguration/domain_name')
set('NodeManagerUsername','username')
set('NodeManagerPasswordEncrypted','password')

# Set the Node Manager type to custom location type and set the custom location
Node Manager home.
setOption('NodeManagerType','CustomLocationNodeManager')
setOption('NodeManagerHome','C:/mydomains/nodemanager')
```

For more information see "`setOption`" in *WLST Command Reference for WebLogic Server*.

4.7 Configuring Java-based Node Manager Security

Java-based Node Manager security uses SSL by default and authenticates incoming connections against a set of credentials specific to each domain.

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 user name and password against the domain `nm_password.properties` file.

Node Manager credentials are located on the ***domain_name*** > **Security** > **General** > **Advanced Options** page in the WebLogic Server Administration Console.

WebLogic Server 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.

This section includes the following topics:

- Section 4.7.1, "Specifying Node Manager User Name and Password"
- Section 4.7.2, "Remote Server Start Security for Java-based Node Manager"
- Section 4.7.3, "Using SSL With Java-based Node Manager"

4.7.1 Specifying Node Manager User Name and Password

The `nm_password.properties` file contains the encrypted Node Manager user name and password. These are used to authenticate connection between a client (for example, the Administration Server) and Node Manager.

> **Note:** This user name and password are only used to authenticate connections between Node Manager and clients. They are independent from the server administration ID and password.

This file is created for you when you use `nmEnroll` to copy the necessary configurations files from one machine to another when creating a domain or when using the Configuration Wizard. The file is located in *DOMAIN_*

HOME/config/nodemanager, where *DOMAIN_HOME* is the location of your WebLogic domain, typically, *ORACLE_HOME*\user_projects\domains*domain_name*.

The Configuration Wizard prompts for a Node Manager user name and password for the initial configuration. This value is populated in the required file locally, however, in order to get it distributed remotely, you must use the nmEnroll command.

After nm_password.properties is created, you can change the values for the Node Manager password and properties using the WebLogic Server Administration Console. Changes are propagated to the nm_password.properties file and are picked up by Node Manager.

You can use the following steps to alter Node Manager credentials:

1. Start the Administration Server.
2. Using the WebLogic Server Administration Console, update the Node Manager credentials using the Advanced options under ***domain_name*** > **Security** > **General**.
3. Invoke WLST and connect to an Administration Server using the connect command. See "Using the WebLogic Scripting Tool" in *Understanding the WebLogic Scripting Tool*.
4. Run nmEnroll using the following syntax:

   ```
   nmEnroll([domainDir], [nmHome])
   ```

   For example,

   ```
   nmEnroll('C:/oracle/user_projects/domains/prod_domain',
   'C:/oracle/user_projects/domains/prod_domain/nodemanager')
   ```

   Running nmEnroll ensures that the correct Node Manager user and password token are supplied to each Managed Server.

   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.

   Note: If you edit nm_password.properties manually (not recommended), you must restart Node Manager in order for the changes to take effect, whereas a restart is not required if you modify the values using the WebLogic Server Administration Console with Node Manager running.

The nm_password.properties file must exist in the domain directory for each physical machine that runs Node Manager. If you change the domain's Node Manager user name and password, you should run nmEnroll on each machine to synchronize the nm_password.properties file. If you configure multiple domains on a machine, each domain can use a different Node Manager user name and password.

In a typical development environment, you may not be prompted to specify the Node Manager user name and password when you create your domain. The Node Manager user name and password default to the administrator credentials, which you can change from the WebLogic Server Administration Console or WLST. However, in a production environment, you must explicitly set the Node Manager user name and password.

4.7.2 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 user 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 and password can be provided in the following ways:
  - On the command line. See Section 2.5.2, "How Node Manager Starts an Administration Server."
  - From the Administration Server `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.
  - Generated for you in a secure, encrypted way with the following steps:
    * Start the Administration Server with the flag `-Dweblogic.nodemanager.ServiceEnabled=true`.
    * Create the *`DOMAIN_HOME`*`\servers\AdminServer\data\nodemanager` directory.
    * Update any startup properties or the server's credentials while the both the Administration Server and Node Manager are running.

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.

4.7.3 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. `DemoIdentity.jks` is installed in the *`DOMAIN_HOME`*`\security` directory and `DemoTrust.jks` is in *`WL_HOME`*`\server\lib`. For testing and development purposes, the keystore configuration is complete.

Configure the `CustomIdentityKeyStoreFileName` properties in `nodemanager.properties` to set up a certificate for Node Manager. Node Manager can have its own certificate or it can share a certificate with another aspect of the domain. However, in order to communicate with the Node Manager instance, clients must trust the Identity in the Node Manager certificate. You do not need a separate certificate for each Node Manager instance, but you can configure this option if desired. In production environments, Node Manager can use the same public certificate used for all server instances.

Configuring SSL for a production environment involves obtaining identity for Node Manager and then configuring both identity and trust for each Administration Server and Managed Server with which Node Manager will be communicating. In addition, the use of host name verification and the Administration port must be taken into consideration. To review the SSL-related properties in `nodemanager.properties`, see Table 4–1. To configure production SSL components, see "Configuring SSL" in

Administering Security for Oracle WebLogic Server 12c (12.2.1).

4.8 Advanced Node Manager Configuration

This section includes the following topics:

4.8.1 Defining the Administration Server Address

Make sure that a listen address is defined for each Administration Server that will connect to a 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 WebLogic Server Administration Console.

4.8.2 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 instance 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 instance. In contrast, stop scripts are executed after the server instance has shutdown.

4.8.2.1 Script Location

Both the start and stop scripts should be placed in the following directory:

DOMAIN_HOME`\bin`

Script execution should occur relative to this directory.

4.8.2.2 Best Practices When Using Start and Stop Scripts

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

4.8.2.3 Using Start Scripts

You can use a start script 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 `weblogic.StartScriptEnabled` property to `true`. (The default is `true`.) 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 `weblogic.StartScriptName` property to the name or path of your script in the `nodemanager.properties` file.

> **Note:** When creating a custom start script, start the server instance in place instead of running `startWebLogic.sh` in the background. This way, while the server instance is running, the custom script process is also running. For example:
>
> ```
> # custom stuff
> # custom stuff
> startWebLogic.sh
> ```

Node Manager sets the `JAVA_VENDOR`, `JAVA_HOME`, `JAVA_OPTIONS`, `SECURITY_POLICY`, `CLASSPATH`, and `ADMIN_URL`. It retrieves these values from the `ServerMBean`, `ServerStartMBean`, and `SSLMBean` when you use the WebLogic Server Administration Console to start the server instance, or WLST connected to the WebLogic Server Administration Server. When you use WLST connected directly to Node Manager, you can specify the values; otherwise, they are left empty.

Node Manager combines all of the command line startup options (-D flags) that are specified in the `ServerStartMBean Arguments` attribute, as well as the `SSLArguments` into a single environmental variable called `JAVA_OPTIONS`. `SSLArguments` are retrieved from the values in the `SSLMBean`. The `SSLMBean` is inspected for `ignoreHostnameVerification`, `HostnameVerifier`, and `ReverseDNSAllowed` values, then those values are appended to the -D flags. All of those flags comprise the `SSLArguments` parameter. All of the values for `SSLArguments` as well as `Arguments` in the `ServerStartMBean` comprise the `JAVA_OPTIONS` environment variable that is defined for the start script. In addition, the script will append any of its own defined values onto this environment variable.

If there are resulting overlaps in this set of values, it will appear to the Java command line like this:

```
java -Dflag1=value1 -Dflag1=value2 weblogic.Server
```

The Java invocation will resolve the duplicate values.

4.8.2.4 Using Stop Scripts

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

To define a stop script:

1. In the `nodemanager.properties` file, set the `weblogic.StopScriptEnabled` property to `true`.
2. Set the `weblogic.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 unmount 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
```

4.8.3 Configuring nodemanager.domains File

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

> **Note:** If using per domain Node Manager, you should not modify the `nodemanager.domains` file. However, if using per host Node Manager, or a custom Node Manager instance, you may need to edit the `nodemanager.domains` file to specify your domains.

This file must contain an entry specifying the domain directory for each domain a 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`.

For the Java-based Node Manager, this file is typically located under *ORACLE_HOME*`\user_projects\domains\`*domain_name*`\nodemanager`. If you created your domain with the Configuration Wizard, the `nodemanager.domains` file was created for you. If configuring a per host Node Manager instance, you must manually create or copy a `nodemanager.domains` file under *ORACLE_HOME*`\oracle_common\common\nodemanager`, the per host *NodeManagerHome* location. For more information, see Section 4.3, "Configuring Per Host Node Manager."

If necessary, you can manually edit `nodemanager.domains` to add domains or register multiple domain locations under a single domain name.

To configure multiple domain registration, manually enter the alternate paths in the `nodemanager.domains` file, in this form:

`domainName=`*primaryDomainPath*`;`*alternateDomainPath1*`;`*alternateDomainPath2*

The *primaryDomainPath* is the path to the domain location where Managed Servers exist and from where they will run, as the Administration Server does not typically pass a path to Node Manager to access a domain. The domain is only accessible by name.

An *alternateDomainPath* is only accessible by name and path and is typically the location of the Administration Server. Clients connecting directly to Node Manager can access the alternate domain path with both a domain name value and a domain path value.

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

Example 4–2 nodemanager.domains File

```
#Domains and directories created by Configuration Wizard
#Mon Jan 07 10:57:18 EST 2013
base_domain=C\:\\Oracle\\Middleware\\Oracle_Home\\user_projects\\domains\\base_
domain
prod_domain=C\:\\Oracle\\Middleware\\Oracle_Home\\user_projects\\domains\\prod_
domain
```

4.8.4 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. Values supplied on the command line override the values in `nodemanager.properties`.

`nodemanager.properties` is created in the directory specified in *NodeManagerHome*, where *NodeManagerHome* typically is *ORACLE_HOME*`\user_projects\domains\`*domain_name*`\nodemanager`. 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.

4.8.4.1 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 `weblogic.StartScriptEnabled` and `NativeVersionEnabled` properties.
- If Node Manager runs on a multihomed system, and you want to control which address and port it uses, define `ListenAddress` and `ListenPort`.

Table 4–1 describes Node Manager properties.

Table 4–1 Node Manager Properties

Node Manager Property	Description	Default
`AuthenticationEnabled`	If set to `true`, Node Manager authenticates the credentials against the domain.	`true`
`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`
`CipherSuite`	The name of the cipher suite to use with the SSL listener. This property is deprecated as of WebLogic Server 12.1.3.0 but remains fully supported in WebLogic Server 12.2.1. See `CipherSuites` for more information and current support limitations for the replacement property.	The default value is JDK and platform dependent.
`CipherSuites`	The name of the cipher suites to use with the SSL listener. You can specify multiple cipher suite values separated by a comma, for example: `CipherSuites=SUITE_A, SUITE_B, SUITE_C` This property introduced in WebLogic Server 12.1.3 is not currently supported by WLST offline or by the `pack` and `unpack` commands. When using those utilities, use `CipherSuite` instead.	The default value is JDK and platform dependent.
`CoherenceStartScriptEnabled`	When starting standalone Coherence servers, this property specifies whether Node Manager should use a start script for the Coherence instance or a direct Java invocation. This property is deprecated as of WebLogic Server 12.1.3.0 but remains fully supported in WebLogic Server 12.2.1. See `coherence.StartScriptEnabled` for more information and current support limitations for the replacement property.	`true`
`CoherenceStartScriptName`	Specifies the start script name or path used to start standalone Coherence servers when `coherence.StartScriptEnabled=true`. This property is deprecated as of WebLogic Server 12.1.3.0 but remains fully supported in WebLogic Server 12.2.1. See `coherence.StartScriptName` for more information and current support limitations for the replacement property.	`none`
`coherence.StartScriptEnabled`	When starting standalone Coherence servers, this property specifies whether Node Manager should use a start script for the Coherence instance or a direct Java invocation. This property introduced in WebLogic Server 12.1.3 is not currently supported by WLST offline or by the `pack` and `unpack` commands. When using those utilities, use `CoherenceStartScriptEnabled` instead.	`true`

Table 4–1 (Cont.) Node Manager Properties

Node Manager Property	Description	Default
`coherence.StartScriptName`	Specifies the start script name or path used to start standalone Coherence servers when `coherence.StartScriptEnabled=true`. This property introduced in WebLogic Server 12.1.3 is not currently supported by WLST offline or by the `pack` and `unpack` commands. When using those utilities, use `CoherenceStartScriptName` instead.	`none`
`coherence.startup.JavaHome`	Applies when `coherence.StartScriptEnabled=false` and Node Manager invokes Java directly.The Java home directory that Node Manager uses to start Managed Servers on this machine, if the Managed Server does not have a Java home configured in its Remote Start page. If not specified in either place, Node Manager uses the Java home defined for a Node Manager process. This property introduced in WebLogic Server 12.1.3 is not currently supported by WLST offline or by the `pack` and `unpack` commands. When using those utilities, use `JavaHome` instead.	`none`
`CrashRecoveryEnabled`	Enables system crash recovery. **Note:** The `CrashRecoveryEnabled` property takes precedence over the `AutoRestart` server startup property in a crash recovery scenario. For example, if a server instance has `AutoRestart=false` but `CrashRecoveryEnabled=true`, when Node Manager restarts, Node Manager tries to recover the server instance if the server failed when Node Manager was not running.	`false`
`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 a 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`

Table 4–1 (Cont.) Node Manager Properties

Node Manager Property	Description	Default
`CustomIdentity KeyStoreType`	Specifies the type of the Identity keystore. Generally, this is JKS. This property is optional.	default keystore type from `java.security`
`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`
`DomainsDirRemoteSharingEnabled`	Specifies whether Node Manager is monitoring a shared domain directory. As such, more than one Node Manager may be monitoring the shared directory from different machines. Set to `true` to indicate that you have a shared domain directory (mounted directory or Windows NFS) that multiple nodes will be sharing. Enabling this property allows multiple Node Managers to share the domain without affecting each other.	`false`
`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`
`DomainRegistrationEnabled`	This property is deprecated as of WebLogic Server 12.1.3.0 and may be removed in a future release. To register multiple domain locations in the `nodemanager.domains` file, see Section 4.8.3, "Configuring nodemanager.domains File."	`false`
`IfConfigDir`	This configuration property sets a different directory for the location of the `wlsifconfig.sh/.cmd` script. This property is deprecated as of WebLogic Server 12.1.3.0 but remains fully supported in WebLogic Server 12.2.1. See `weblogic.IfConfigDir` for more information and current support limitations for the replacement property.	By default, this location is set appropriately, but you can use this property to modify the script location.
`Interface`	The primary interface names used by migratable servers. For server migration, the primary interface names used by migratable servers must be the same. See the `<InterfaceName>` property for more flexibility specifying multiple interfaces and a corresponding range of IP addresses that should be bound to a specific interface. This property is deprecated as of WebLogic Server 12.1.3.0 but remains fully supported in WebLogic Server 12.2.1. See `<InterfaceName>` for more information.	`none`

Table 4–1 (Cont.) Node Manager Properties

Node Manager Property	Description	Default
<InterfaceName>	An interface name along with a corresponding range of IP addresses and optional netmask value that should be bound to this specific network interface when migratable servers are started. Syntax: <InterfaceName>=<IP_RANGE_MIN>-<IP_RANGE_MAX>,(optional) NetMask=<NETMASK_ADDRESS> For example, the syntax for binding addresses 1 - 4 to interface eth0 and addresses 5 - 8 to bond0 is: eth0=1-4,NetMask=255.255.255.0 bond0=5-8,NetMask=255.255.248.0 You can leave out the NetMask value, if desired, and simply enter: eth0=200.10.10.1-200.10.10.255 bond0=199.0.0.1-199.0.0.255 The original NetMask and Interface properties are still supported and when specified, would apply to any address that is not already defined in an IP range. For example, specifying these properties in the original format: Interface=oldEth0 NetMask=255.255.255.0 Would be the same as specifying this in the new format: oldEth0=*,Netmask=255.255.255.0 An asterisk (*) can be represent all IPs.	none
JavaHome	The Java home directory that Node Manager uses to start Managed Servers on this machine, if the Managed Server does not have a Java home configured in its Remote Start page. If not specified in either place, Node Manager uses the Java home defined for a Node Manager process. **Note:** Oracle recommends not setting this property with the WLST set command. If set through WLST, this property reverts to its default value in the nodemanager.properties file. This property is deprecated as of WebLogic Server 12.1.3.0 but remains fully supported in WebLogic Server 12.2.1. For more information and current support limitations for the replacement properties, see weblogic.startup.JavaHome for WebLogic Server processes and coherence.startup.JavaHome for Coherence processes.	none

Table 4–1 (Cont.) Node Manager Properties

Node Manager Property	Description	Default
JavaStandardTrustKeyStorePassPhrase	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
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
KeyStores	Indicates the keystore configuration 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 *DOMAIN_HOME*\security and *WL_HOME*\server\lib directories 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) ■ CustomIdentityAndJava ■ StandardTrust Uses a keystore you create, and the trusted CAs defined in the cacerts file in the JAVA_HOME\jre\lib\security\cacertsdirectory. ■ 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

Table 4–1 (Cont.) Node Manager Properties

Node Manager Property	Description	Default
`ListenBacklog`	The maximum number of Node Manager backlog requests that the listener will accept. Additional incoming requests will be dropped until the backlogged requests are handled. Typically, you need not adjust this property. However, under heavy load (more than 50 concurrent requests on Node Manager), some connections to Node Manager may fail. If you encounter dropped connections in large environments where Node Manager connects to many server instances, such as with consensus leasing in large clusters, then increase the `ListenBacklog` value.	`50`
`ListenPort`	The TCP port number on which Node Manager listens for connection requests. This argument deprecates `weblogic.nodemanager.listenPort`.	`5556`
`LogAppend`	If set to `true`, then a new log file is not created when Node Manager restarts; the existing log is appended instead.	`true`
`LogCount`	Maximum number of log files to create when `LogLimit` is exceeded.	`1`
`LogFile`	Location of the Node Manager log file.	*`NodeManagerHome`*`\nodemanager.log`
`LogFormatter`	Name of formatter class to use for NM log messages.	`weblogic.nodemanager.server.LogFormatter`
`LogLevel`	Severity level of logging used for the Node Manager log. Node Manager uses the standard logging levels from the `java.util.logging.level` package, `http://docs.oracle.com/javase/6/docs/api/java/util/logging/Level.html`.	`INFO`
`LogLimit`	Specifies an approximate maximum size of the Node Manager Log as an integer. When this limit is reached, a new log file is started.	`unlimited`
`LogToStderr`	If set to `true`, the log output is also sent to the standard error output.	`false`

Table 4–1 (Cont.) 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 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 `weblogic.StartScriptEnabled` property to start Managed Servers. Note that when `NativeVersionEnabled=false`: ■ Node Manager cannot query if a PID is alive nor kill a particular process ■ Node Manager does not have the ability to determine if there are existing processes that need to be monitored and crash recovery is not fully implemented ■ `nmKill` is not supported when `NativeVersionEnabled=false` and `weblogic.StartScriptEnabled=true`	`true`
`NetMask`	The subnet mask for your network. For server migration, each Managed Server must use the same subnet mask to enable unicast and multicast communication among servers. See the `<InterfaceName>` property for more flexibility entering multiple interfaces and corresponding netmask values. This property is deprecated as of WebLogic Server 12.1.3.0 but remains fully supported in WebLogic Server 12.2.1. See `<InterfaceName>` for more information.	`none`

Table 4–1 (Cont.) Node Manager Properties

Node Manager Property	Description	Default
`NodeManagerHome`	Node Manager root directory which contains the following configuration and log files: ■ `nodemanager.domains` ■ `nodemanager.log` ■ `nodemanager.properties` For more information on these files, see Section 2.7, "Node Manager Configuration and Log Files." Multiple Node Manager instances cannot share the same `NodeManagerHome` directory. For shared domain topologies, you must create a unique `NodeManagerHome` location for each Node Manager instance. For example, create a unique directory based on each host name, and then use the host name directories to specify each `NodeManagerHome` location. When Node Manager starts, it uses the unique host name directory, and each Node Manager instance maintains its own `NodeManagerHome` location and does not inappropriately share its resources. **Note:** By default, *`NodeManagerHome`* is *`DOMAIN_HOME`*`\nodemanager`. In a production environment, you might want to customize the location of the Node Manager root directory. **Note:** Oracle recommends not setting this property with the WLST `set` command. If set through WLST, this property reverts to its default value in the `nodemanager.properties` file.	*`NodeManagerHome`*
`PropertiesVersion`	Specifies the version of the `nodemanager.properties` file. Do not change this value.	`none`
`QuitEnabled`	If set to `true`, allow the user to remotely stop Node Manager. WLST overrides the default value, `false`, when stopping Node Manager using the `stopNodeManager()` command.	`false`
`SecureListener`	If set to `true`, use the SSL listener, otherwise use the plain socket.	`true`
`StartScriptEnabled`	If true, use the start script specified by `StartScriptName` to start a server instance. For more information, see Section 4.8.2, "Configuring Node Manager to Use Start and Stop Scripts." This property is deprecated as of WebLogic Server 12.1.3.0 but remains fully supported in WebLogic Server 12.2.1. See `weblogic.StartScriptEnabled` for more information and current support limitations for the replacement property.	`true`

Table 4–1 (Cont.) Node Manager Properties

Node Manager Property	Description	Default

`StartScriptName`	The name or path of the start script, located in the domain directory.
	This property is deprecated as of WebLogic Server 12.1.3.0 but remains fully supported in WebLogic Server 12.2.1. See `weblogic.StartScriptName` for more information and current support limitations for the replacement property.
		`startWebLogic.sh (UNIX)`
		or
		`startWebLogic.cmd (Windows)`

`StateCheckInterval`	Specifies the interval Node Manager waits to perform a check of the server state.	`500 milliseconds`
`StopScriptEnabled`	If `true`, execute the stop script specified by `StopScriptName` after the server instance has shutdown. For more information, see Section 4.8.2, "Configuring Node Manager to Use Start and Stop Scripts." This property is deprecated as of WebLogic Server 12.1.3.0 but remains fully supported in WebLogic Server 12.2.1. See `weblogic.StopScriptEnabled` for more information and current support limitations for the replacement property.	`false`
`StopScriptName`	The name of the script to be executed after server shutdown. This property is deprecated as of WebLogic Server 12.1.3.0 but remains fully supported in WebLogic Server 12.2.1. See `weblogic.StopScriptName` for more information and current support limitations for the replacement property.	`none`
`UseMACBroadcast`	Specifies whether or not to use a node's MAC address when sending ARP packets, that is, whether or not to use the -b flag in the `arping` command.When Node Manager binds IP addresses for automatic server migration, it also uses the `arping` command to attempt to notify nodes in the network that the machine is now using the virtual IP address. When `weblogic.UseMACBroadcast=true`, Node Manager uses the `arping` command with the -b flag. This property is deprecated as of WebLogic Server 12.1.3.0 but remains fully supported in WebLogic Server 12.2.1. See `weblogic.UseMACBroadcast` for more information and current support limitations for the replacement property.	`false`
`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 page. If not specified in either place, Node Manager starts the Managed Server in the directory where Node Manager runs.	`none`

Table 4–1 (Cont.) Node Manager Properties

Node Manager Property	Description	Default
`weblogic.IfConfigDir`	This configuration property sets a different directory for the location of the `wlsifconfig.sh/.cmd` script. This property introduced in WebLogic Server 12.1.3 is not currently supported by WLST offline or by the `pack` and `unpack` commands. When using those utilities, use `IfConfigDir` instead.	By default, this location is set appropriately, but you can use this property to modify the script location.
`weblogic.StartScriptEnabled`	If true, use the start script specified by `weblogic.StartScriptName` to start a server instance. For more information, see Section 4.8.2, "Configuring Node Manager to Use Start and Stop Scripts." This property introduced in WebLogic Server 12.1.3 is not currently supported by WLST offline or by the `pack` and `unpack` commands. When using those utilities, use `StartScriptEnabled` instead.	`true`

`weblogic.StartScriptName`	The name or path of the start script, located in the domain directory.
	This property introduced in WebLogic Server 12.1.3 is not currently supported by WLST offline or by the `pack` and `unpack` commands. When using those utilities, use `StartScriptName` instead.
		`startWebLogic.sh (UNIX)`
		or
		`startWebLogic.cmd (Windows)`

`weblogic.startup.JavaHome`	Applies when `weblogic.StartScriptEnabled=false` and Node Manager invokes Java directly. Specifies the Java home directory that Node Manager uses to start Managed Servers on this machine, if the Managed Server does not have a Java home configured in its Remote Start page. If not specified in either place, Node Manager uses the Java home defined for a Node Manager process. This property introduced in WebLogic Server 12.1.3 is not currently supported by WLST offline or by the `pack` and `unpack` commands. When using those utilities, use `JavaHome` instead.	`none`

Table 4–1 (Cont.) Node Manager Properties

Node Manager Property	Description	Default
`weblogic.StopScriptEnabled`	If `true`, execute the stop script specified by `weblogic.StopScriptName` after the server instance has shutdown. For more information, see Section 4.8.2, "Configuring Node Manager to Use Start and Stop Scripts." This property introduced in WebLogic Server 12.1.3 is not currently supported by WLST offline or by the `pack` and `unpack` commands. When using those utilities, use `StopScriptEnabled` instead.	`false`
`weblogic.StopScriptName`	The name of the script to be executed after server shutdown. This property introduced in WebLogic Server 12.1.3 is not currently supported by WLST offline or by the `pack` and `unpack` commands. When using those utilities, use `StopScriptName` instead.	`none`
`weblogic.UseMACBroadcast`	Specifies whether or not to use a node's MAC address when sending ARP packets, that is, whether or not to use the -b flag in the `arping` command.When Node Manager binds IP addresses for automatic server migration, it also uses the `arping` command to attempt to notify nodes in the network that the machine is now using the virtual IP address. When `weblogic.UseMACBroadcast=true`, Node Manager uses the `arping` command with the -b flag. This property introduced in WebLogic Server 12.1.3 is not currently supported by WLST offline or by the `pack` and `unpack` commands. When using those utilities, use `UseMACBroadcast` instead.	`false`

4.8.4.2 Machine-Level Node Manager Settings for a Group of Server Instances

If you have a group of server instances started by the same Node Manager instance, you can configure certain machine-level settings once in the `nodemanager.properties` file and these settings will apply to all server instances in that group. Setting machine-level attributes simplifies configuration, as you do not have to configure common settings in the `ServerStartMBean` for each server instance.

If you configure specific settings for a server instance directly in the `ServerStartMBean`, that value takes precedence over any values configured in the `nodemanager.properties` file.

Table 4–2 describes the machine-level settings you can configure in `nodemanager.properties` for a group of server instances started by the same Node Manager instance.

Table 4–2 Machine-Level Node Manager Properties for a Group of Server Instances

Property Name	Description
coherence.startup.Arguments	The Java arguments to use when starting the server instance. These are the first arguments appended immediately after java portion of the startup command. For example, you can set Java heap memory or specify any weblogic.nodemanager.server.provider.WeblogicCacheServer option. Separate arguments with a space.
coherence.startup.MW_Home	The MW_HOME directory (the path on the machine running Node Manager) to use when starting this server instance. Specifies the directory on the Node Manager machine under which all Oracle Middleware products are installed. For example, C:\Oracle\Middleware.
weblogic.startup.Arguments	The Java arguments to use when starting the server instance. Node Manager passes this value to a start script using the JAVA_OPTIONS environment variable. When issuing a Java command line to start the server instance, Node Manager passes the arguments as options.
weblogic.startup.Arguments.prepend	Prepends flags to any arguments configured for the server instances.
weblogic.startup.ClassPath	The classpath (the path on the machine running Node Manager) to use when starting this server instance. Node Manager passes this value to a start script using the CLASSPATH environment variable. When issuing a Java command line to start the server instance, Node Manager defines the classpath with -Djava.class.path=.
weblogic.startup.JavaVendor	The Java vendor value to use when starting the server instance. Node Manager does not pass this value when invoking a Java command line to start the server instance. It does pass this value in the environment variable JAVA_VENDOR to the start script.
weblogic.startup.MW_Home	The MW_HOME directory (the path on the machine running Node Manager) to use when starting this server instance. Node Manager does not pass this value to start scripts. When issuing a Java command line to start the server instance, Node Manager specifies MW_HOME with -Dbea.home=.
weblogic.startup.SecurityPolicyFile	The security policy file (the directory and file name on the machine running Node Manager) to use when starting this server instance. When Node Manager uses a start script, the security policy file is defined in an environment variable, SECURITY_POLICY. When issuing a Java command line to start the server instance, Node Manager defines the security policy with -Djava.security.policy=.
weblogic.startup.ServerGID	The group ID for the server instance.
weblogic.startup.ServerUID	The user ID for the server instance.

4.8.5 Configuring Remote Startup Arguments

In the WebLogic Server Administration Console, on 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, default values are used, as appropriate for the Managed Server. If using the Java-based implementation, Node Manager uses its own properties as defaults to start the Managed Server. If using the script-based implementation, the start script sets the default values. For more information, see Section 4.8.4, "Reviewing nodemanager.properties."

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. The specified startup arguments are used for starting Managed Servers only. They will not be used by an Administration Server instance that is started by Node Manager.

If you will run Node Manager as a Windows service, as described in Section 6.1.1, "Running Node Manager as a Startup Service," you must configure the `-Xrs` JVM property for each Managed Server that will be under Node Manager control.

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 Node Manager and Administration Server operating system services are shut down.
2. During the interval between the Managed Server being killed, and a 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.

4.8.6 Setting Server Startup Properties

You can use Node Manager to set the startup properties for a server instance. 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.

4.8.6.1 startup.properties

Node Manager uses the `startup.properties` file to determine the startup configuration when starting a server instance. 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 instance is part of a cluster. For more information, see the *MBean Reference for Oracle WebLogic Server*.

4.8.6.2 Setting Startup Properties Using WLST

When using the WLST `nmStart` command, the server configuration cannot be determined directly. Therefore, you must pass the server start properties as a WLST properties object to the `nmStart` command.

4.8.6.3 Server Startup Properties

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

Table 4–3 Server Startup Properties

Property	Description
`AdminURL`	The URL of the Administration Server. **Note:** This value should only be specified in the `startup.properties` file for a Managed Server.
`Arguments`	The arguments used when starting the server instance.
`AutoKillIfFailed`	When a server instance is started by Node Manager, this attribute signals that the `OverloadProtectionMBean` settings will be changed to `FailureAction == FORCE_SHUTDOWN` and `PanicAction == SYSTEM_EXIT`. For more information about `FailureAction` and `PanicAction`, see `OverloadProtectionMBean`.
`AutoRestart`	Specifies whether Node Manager can automatically restart this server instance if it fails. **Note:** The `CrashRecoveryEnabled` configuration property takes precedence over the `AutoRestart` property in a crash recovery scenario. For example, if a server instance has `AutoRestart=false` but `CrashRecoveryEnabled=true`, when Node Manager restarts, Node Manager tries to recover the server instance if the server failed when Node Manager was not running.
`ClassPath`	The classpath to use when starting a server instance.
`JavaHome`	Defines the Java home directory used when starting the server instance.
`OracleHome`	The Oracle home directory to use when starting a server instance.
`RestartDelaySeconds`	The number of seconds Node Manager should wait before attempting to restart the server instance.
`RestartInterval`	The amount of time Node Manager will spend attempting to restart a failed server instance. 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 instance indefinitely until the `FAILED_NOT_RESTARTABLE` state is reached.
`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.
`SecurityPolicyFile`	Specifies the security policy file to use when starting this server.
`ServerIP`	The IP address of the server.
`SSLArguments`	These arguments are used when you have enabled the domain-wide administration port.

4.8.7 Set Node Manager Environment Variables

By default, you need not set any additional environment variables before starting Node Manager. The sample Node Manager start scripts and install service scripts provided with WebLogic Server set the required variables and start Node Manager listening on the default address, `localhost`.

To start Node Manager listening on a non-default address, you can use one of the following methods:

- Edit the `nodemanager.properties` file.

  Set the `LISTEN_ADDRESS` variable to `<host>` and the `LISTEN_PORT` variable to `<port>` before calling the `startNodeManager` script. See Section 4.8.4, "Reviewing nodemanager.properties."

- Set the values when executing the `startNodeManager` script.

  The `startNodeManager` scripts will set the first two positional parameters to `LISTEN_ADDRESS` and `LISTEN_PORT` when entered on the command line.

  For example, enter this command to start Node Manager on host `llama` and port `7777`:

  ```
  startNodeManager.cmd llama 7777 (Windows)
  sh startNodeManager.sh llama 7777 (UNIX)
  ```

  Enter this command to start Node Manager on host `llama`:

  ```
  startNodeManager.cmd llama (Windows)
  sh startNodeManager.sh llama (UNIX)
  ```

Configuring a non-default listening address for Node Manager is most useful in production environments so that traffic from other machines can potentially reach it. Also, if you have a multihomed machine or a machine with multiple network interface cards, Node Manager can be listening on any one of the addresses on the machine.

Table 4–4 Node Manager Environment Variables

Environment Variable	Description
`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`
`JAVA_HOME`	JDK root directory used by Node Manager. For example: `set JAVA_HOME=c:\jdk1.7.0_06` Node Manager has the same JDK version requirements as WebLogic Server.

`LD_LIBRARY_PATH` (UNIX and Linux)	For 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`
	Linux example:
	`LD_LIBRARY_PATH:$WL_HOME/server/native/linux:$WL_HOME/server/native/linux/i686`
	Note: Linux can be `i686`, `ia64`, or `x86_64` architecture. The path would change to correspond with the appropriate architecture.

`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%`
`WL_HOME`	WebLogic Server installation directory. For example: `set WL_HOME=c:\Oracle\Middleware\Oracle_Home\wlserver`

4.8.8 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`:

```
#
# Create the $domaindir/bin/startNMService.sh script or the
# $WL_HOME/server/bin/startNMService.sh script to produce output to a file.
#

#!/bin/sh

$domaindir/bin/startNodeManager.sh >> $NM_HOME/nmservice.out 2>&1

#
# The service can now take advantage of the startNMService script.
#

# default: off
# description:nodemanager as a service

service nodemanager-svc
{
   type            = UNLISTED
   disable         = yes
   socket_type     = stream
   protocol        = tcp
   wait            = yes
   user            = <username>
   port            = 5556
   flags           = NOLIBWRAP
   log_on_success += DURATION HOST USERID
   server          = /scratch/jdorr/dom1213/bin/startNMService.sh
   env             = MW_HOME=/Oracle/Middleware/Oracle_Home_WLS_12.2.1 JAVA_
HOME=/Java/jdk1.7.0_51LD_LIBRARY_PATH=/Oracle/Middleware/Oracle_Home_WLS_
12.2.1/bea/wlserver/server/native/linux/x86_64:/usr/lib:/lib:/usr/X11R6/lib
}
```

4.8.9 Configuring Node Manager as an init.d Service

You can configure Node Manager as an `init.d` service using `init` startup scripts. This section includes the following topics:

- Configuring Per Domain Node Manager as an init.d Service
- Configuring Per Host Node Manager as an init.d Service

4.8.9.1 Configuring Per Domain Node Manager as an init.d Service

To install and configure a per domain Node Manager as an `init.d` service, complete the following steps. Execute all actions as the root user.

Note: When Node Manager runs as an `init.d` service, the launched Managed Servers are owned by the root user. To start Managed Servers as non-root user, first use the Administration Console to enable the Post-Bind UID and Post-Bind GID attributes on the **Domain > Environment > Machines > Configuration > General** page. Then, restart Node Manager and the Administration Server before restarting the Managed Servers.

1. Determine the service name, referred to as `<service name>` in the following steps.

   Oracle recommends using the naming pattern `DOMAIN_NAME-nodemanager`, for example, `base_domain-nodemanager`.

2. Copy the sample script in Example 4–3 to `/etc/init.d/<service name>`.
3. Edit `/etc/init.d/<service name>` and change the following variables according to your environment and installation:
   - the `Provides` setting in the `###BEGIN INIT INFO` section in Example 4–3
   - the `DOMAIN_HOME` setting in the `###BEGIN CUSTOM ENV` section in Example 4–3
4. Make `/etc/init.d/<service name>` executable by using the `chmod +x` command.
5. Add your `init.d` service: `sudo /sbin/chkconfig --add <service name>`
6. To start, stop, restart, or show the status of your service, use the following commands:

To start the service:	`sudo /sbin/service <service name> start`
To stop the service:	`sudo /sbin/service <service name> stop`
To restart the service:	`sudo /sbin/service <service name> restart`
To show the status of the service:	`sudo /sbin/service <service name> status`

Example 4–3 Configuring a Per Domain Node Manager as an init.d Service

```
#!/bin/sh
#
# chkconfig:   345 85 15
# description: per domain Oracle Weblogic Node Manager service init script

### BEGIN INIT INFO
# Provides: jtest-dom1213-nodemanager
# Required-Start: $network $local_fs
# Required-Stop:
# Should-Start:
# Should-Stop:
# Default-Start: 3 4 5
# Default-Stop: 0 1 2 6
# Short-Description: per domain Oracle Weblogic Node Manager service.
# Description: Starts and stops per domain Oracle Weblogic Node Manager.
### END INIT INFO

. /etc/rc.d/init.d/functions

#### BEGIN CUSTOM ENV
DOMAIN_HOME="/scratch/jdorr/dom1213"
MW_HOME=/Oracle/Middleware/Oracle_Home_WLS_12.2.1
```

```
JAVA_HOME=/Java/jdk1.7.0_51
LD_LIBRARY_PATH=/Oracle/Middleware/Oracle_Home_WLS_
12.2.1/bea/wlserver/server/native/linux/x86_64:/usr/lib:/lib:/usr/X11R6/lib
#### END CUSTOM ENV

PROCESS_STRING="^.*$DOMAIN_HOME.*weblogic.NodeManager.*"
PROGRAM_START="$DOMAIN_HOME/bin/startNodeManager.sh"
NODEMANAGER_DIR=$DOMAIN_HOME/nodemanager
NODEMANAGER_LOCKFILE="$NODEMANAGER_DIR/nodemanager.log.lck"
OUT_FILE="$NODEMANAGER_DIR/nodemanager.out"

SERVICE_NAME=`/bin/basename $0`
LOCKFILE="/var/lock/subsys/$SERVICE_NAME"

RETVAL=0

start() {
        OLDPID=`/usr/bin/pgrep -f $PROCESS_STRING`
        if [ ! -z "$OLDPID" ]; then
            echo "$SERVICE_NAME is already running (pid $OLDPID) !"
            echo
            exit
        fi
        echo -n $"Starting $SERVICE_NAME ... "
        echo "`date` Starting $SERVICE_NAME ... " > $OUT_FILE 2>&1
        export MW_HOME
        export JAVA_HOME
export LD_LIBRARY_PATH
$PROGRAM_START >> $OUT_FILE 2>&1  &

        RETVAL=$?
        if [ $RETVAL -eq 0 ] ; then
          wait_for "socket listener started on port"
        else
          echo "FAILED: $RETVAL. Please check $OUT_FILE for more information."
        fi
        echo
}

wait_for() {
    res=$(cat "$OUT_FILE" | fgrep -c "$1")
    count=60
    while [[ ! $res -gt 0 ]] && [[ $count -gt 0 ]]
    do
        sleep 1
        count=$(($count - 1))
        res=$(cat "$OUT_FILE" | fgrep -c "$1")
    done
    res=$(cat "$OUT_FILE" | fgrep -c "$1")
    if [ ! $res -gt 0 ]; then
        echo "FAILED or took too long time to start. Please check $OUT_FILE for
more information."
    else
        echo "OK"
        touch $LOCKFILE
    fi
}

stop() {
        echo -n $"Stopping $SERVICE_NAME ... "
```

```
        OLDPID=`/usr/bin/pgrep -f $PROCESS_STRING`
        if [ "$OLDPID" != "" ]; then
            echo -n "(pid $OLDPID) "
            /bin/kill -TERM $OLDPID

            RETVAL=$?
            echo "OK"
            /bin/rm -f $NODEMANAGER_LOCKFILE
            [ $RETVAL -eq 0 ] && rm -f $LOCKFILE
        else
            /bin/echo "$SERVICE_NAME is stopped"
        fi
        echo
}

restart() {
        stop
        sleep 10
        start
}

case "$1" in
  start)
        start
        ;;
  stop)
        stop
        ;;
  restart|force-reload|reload)
        restart
        ;;
  condrestart|try-restart)
        [ -f $LOCKFILE ] && restart
        ;;
  status)
        OLDPID=`/usr/bin/pgrep -f $PROCESS_STRING`
        if [ "$OLDPID" != "" ]; then
            /bin/echo "$SERVICE_NAME is running (pid: $OLDPID)"
        else
            /bin/echo "$SERVICE_NAME is stopped"
        fi
        echo
        RETVAL=$?
        ;;
  *)
        echo $"Usage: $0
{start|stop|status|restart|reload|force-reload|condrestart}"
        exit 1
esac

exit $RETVAL
```

4.8.9.2 Configuring Per Host Node Manager as an init.d Service

To install and configure a per host Node Manager as an `init.d` service, complete the following steps. Execute all actions as the root user.

Before you begin, ensure the per host Node Manager is correctly configured in your domain before using this script. See Section 4.3, "Configuring Per Host Node Manager."

1. Determine the service name, referred to as `<service name>` in the following steps.

   Oracle recommends using the naming pattern `<a brief name for MW_HOME>-nodemanager`, for example, `wls12.2.1-nodemanager`.

2. Copy the sample script in Example 4–4 to `/etc/init.d/<service name>`.
3. Edit `/etc/init.d/<service name>` and change the following variables according to your environment and installation:
   - the `Provides` setting in the `###BEGIN INIT INFO` section in Example 4–4
   - the `MW_HOME` setting in the `###BEGIN CUSTOM ENV` section in Example 4–4
4. Make `/etc/init.d/<service name>` executable by using the `chmod +x` command.
5. Add your `init.d` service: `sudo /sbin/chkconfig --add <service name>`
6. To start, stop, restart, or show the status of your service, use the following commands:

To start the service:	`sudo /sbin/service <service name> start`
To stop the service:	`sudo /sbin/service <service name> stop`
To restart the service:	`sudo /sbin/service <service name> restart`
To show the status of the service:	`sudo /sbin/service <service name> status`

Example 4–4 Configuring a Per Host Node Manager as an init.d Service

```
#!/bin/sh
#
# chkconfig:   345 85 15
# description: global (per host) Oracle Weblogic Node Manager service init script

### BEGIN INIT INFO
# Provides: jjjnm
# Required-Start: $network $local_fs
# Required-Stop:
# Should-Start:
# Should-Stop:
# Default-Start: 3 4 5
# Default-Stop: 0 1 2 6
# Short-Description: per host Oracle Weblogic Node Manager service
# Description: Starts and stops per host Oracle Weblogic Node Manager
### END INIT INFO

. /etc/rc.d/init.d/functions

#### BEGIN CUSTOM ENV
MW_HOME=/Oracle/Middleware/Oracle_Home_WLS_12.2.1
JAVA_HOME=/Java/jdk1.7.0_51
LD_LIBRARY_PATH=/Oracle/Middleware/Oracle_Home_WLS_
12.2.1/bea/wlserver/server/native/linux/x86_64:/usr/lib:/lib:/usr/X11R6/lib
#### END CUSTOM ENV

PROGRAM_START="$MW_HOME/wlserver/server/bin/startNodeManager.sh"
NODEMANAGER_DIR="$MW_HOME/wlserver/../oracle_common/common/nodemanager"
NODEMANAGER_LOCKFILE="$NODEMANAGER_DIR/nodemanager.log.lck"
OUT_FILE="$NODEMANAGER_DIR/nodemanager.out"
PROCESS_STRING="^.*\-Dweblogic.RootDirectory=$NODEMANAGER_
DIR.*weblogic.NodeManager.*"
```

```
SERVICE_NAME=`/bin/basename $0`
LOCKFILE="/var/lock/subsys/$SERVICE_NAME"

RETVAL=0

start() {
        OLDPID=`ps -e -o pid,command:1000 | grep "$PROCESS_STRING" | grep -v "
grep " | awk '{print $1}'`
        if [ ! -z "$OLDPID" ]; then
            echo "$SERVICE_NAME is already running (pid $OLDPID) !"
            echo
            exit
        fi

        if [ ! -e "$NODEMANAGER_DIR" ]; then
            echo "$NODEMANAGER_DIR does not exist. The per host Node Manager is
not configured correctly."
            echo "Please follow steps in Pre-Condition section before using this
script."
            echo
            exit
        fi

        echo -n $"Starting $SERVICE_NAME ... "
        echo "`date` Starting $SERVICE_NAME ... " > $OUT_FILE 2>&1
        export MW_HOME
export JAVA_HOME
export LD_LIBRARY_PATH
  $PROGRAM_START >> $OUT_FILE 2>&1  &

        RETVAL=$?
        if [ $RETVAL -eq 0 ] ; then
          wait_for "socket listener started on port"
        else
          echo "FAILED: $RETVAL. Please check $OUT_FILE for more information."
        fi
        echo
}

wait_for() {
    res=$(cat "$OUT_FILE" | fgrep -c "$1")
    count=60
    while [[ ! $res -gt 0 ]] && [[ $count -gt 0 ]]
    do
        sleep 1
        count=$(($count - 1))
        res=$(cat "$OUT_FILE" | fgrep -c "$1")
    done
    res=$(cat "$OUT_FILE" | fgrep -c "$1")
    if [ ! $res -gt 0 ]; then
        echo "FAILED or took too long time to start. Please check $OUT_FILE for
more information."
    else
        echo "OK"
        touch $LOCKFILE
    fi
}

stop() {
        echo -n $"Stopping $SERVICE_NAME ... "
```

```
        OLDPID=`ps -e -o pid,command:1000 | grep "$PROCESS_STRING" | grep -v "
grep " | awk '{print $1}'`
        if [ "$OLDPID" != "" ]; then
            echo -n "(pid $OLDPID) "
            /bin/kill -TERM $OLDPID

            RETVAL=$?
            echo "OK"
            /bin/rm -f $NODEMANAGER_LOCKFILE
            [ $RETVAL -eq 0 ] && rm -f $LOCKFILE
        else
            /bin/echo "$SERVICE_NAME is stopped"
        fi
        echo
}

restart() {
        stop
        sleep 10
        start
}

case "$1" in
  start)
        start
        ;;
  stop)
        stop
        ;;
  restart|force-reload|reload)
        restart
        ;;
  condrestart|try-restart)
        [ -f $LOCKFILE ] && restart
        ;;
  status)
        OLDPID=`ps -e -o pid,command:1000 | grep "$PROCESS_STRING" |grep -v " grep
" | awk '{print $1}'`
        if [ "$OLDPID" != "" ]; then
            /bin/echo "$SERVICE_NAME is running (pid: $OLDPID)"
        else
            /bin/echo "$SERVICE_NAME is stopped"
        fi
        echo
        RETVAL=$?
        ;;
  *)
        echo $"Usage: $0
{start|stop|status|restart|reload|force-reload|condrestart}"
        exit 1
esac

exit $RETVAL
```

5 Configuring Script-Based Node Manager

This chapter describes how to configure script-based Node Manager.

This chapter includes the following sections:

5.1 Overview

Script-based Node Manager is a shell script, `wlscontrol.sh`, located in *`WL_HOME`*`/common/bin/`. The `wlscontrol.sh` file 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.

5.2 Step 1: Create User Accounts

Before running Node Manager, you should create a dedicated UNIX user account for performing Node Manager functions. Add this user to all machines that will host the script-based Node Manager and to all machines that will host a Node Manager client, including the Administration Server.

> **Note:** On UNIX platforms, Oracle does not recommend running Node Manager as the root user. However, to achieve Post-Bind GID, you must start Node Manager as the root user. Post-Bind GID enables a server running on your machine to bind to a UNIX group ID (GID) after it finishes all privileged startup actions.

For example:

1. On each host machine, as the root user, create two new operating system (OS) users: **bea** and **ndmgr**, both associated with a new group called **bea**.
   - Use **bea** for installing WebLogic Server only.
   - Use **ndmgr** to create a WebLogic domain and start the Administration Server and remote Managed Servers using Node Manager.
2. Both OS users should have the same OS group (**bea**) to ensure that the correct permissions are in place for **ndmgr** to run WebLogic scripts and executables.

   For example:

   ```
   > groupadd bea
   > useradd -g bea -m bea
   > passwd bea
   > useradd -g bea -m ndmgr
   > passwd ndmgr
   ```

5.3 Step 2: Configure 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 user name 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. Change the environment file of the user to contain the path to *WL_HOME*/common/bin/ or *DOMAIN_HOME*/bin/server_migration.
>
> For example:
>
> ```
> PATH=/usr/bin:/bin:ORACLE_HOME/user_projects/domains/domain_name/bin/server_
> migration
> ```

Configure SSH trust between the **ndmgr** user on each machine that will run a WebLogic Server instance and the same **ndmgr** user on the same machine, plus the corresponding **ndmgr** user on every other machine.

In other words, any **ndmgr** user on one machine must be able to establish an SSH session without being prompted for security credentials, with a **ndmgr** user of the same name on the same or a different machine. This is necessary because any Managed Server can become the cluster master for migratable servers and issue commands to start other remote Managed Servers in the cluster using SSH. For Managed Server migration to work, the **ndmgr** user needs only to be able to run the `wlscontrol.sh` script using SSH. For more information, see Section 5.13.2, "Configuring Security for WebLogic Server Scripts".

For example, to configure one instance of a user to trust another instance of a user for SSH version2:

1. From a terminal logged in as **ndmgr** user:

   ```
   > ssh-keygen -t dsa
   ```

2. When prompted, accept the default locations and press **Enter** for passphrase so that no passphrase is specified.

3. Copy the **ndmgr** user's public key to the **ndmgr** user's home on the same machine and all other machines.

   ```
   > scp .ssh/id_dsa.pub ndmgr@192.168.1.101:./
   ```

4. Establish an SSH session with the target machine as the **ndmgr** user and set up trust for the remote **ndmgr** user.

   ```
   > ssh -l ndmgr 192.168.1.101 (you should be prompted for password)
   > mkdir .ssh
   > chmod 700 .ssh
   > touch .ssh/authorized_keys
   > chmod 700 .ssh/authorized_keys
   > cat id_dsa.pub >> .ssh/authorized_keys
   > rm id_dsa.pub
   > exit
   ```

5. Test that you can establish an SSH session with the **ndmgr** user on the remote machine without requiring a password.

   ```
   > ssh -l ndmgr 192.168.1.101
   ```

6. Repeat this process for all combinations of machines.

Alternatively, you can achieve the same result by generating a key value pair on each machine, concatenating all of the public keys into an `authorized_keys` file, and copying (`scp`) that file to all machines. Try establishing SSH sessions between all combinations of machines to ensure that the `~/.ssh/known_hosts` files are correctly configured. For more information, see Section 5.13.4, "Generating and Distributing Key Value Pairs".

5.4 Step 3: Install WebLogic Server

As the **bea** user, install a WebLogic Server instance in the base directory, *`ORACLE_HOME`*`/wlserver`, on all the machines that will run WebLogic Server, where ORACLE_HOME represents the directory you specify as the Oracle Home when you install WebLogic Server, for example, `C:\Oracle\Middleware\Oracle_Home`.

For example:

```
> java -jar wls_jrf_generic.jar
```

For more information, see "Starting the Installation Program" in *Installing and Configuring Oracle WebLogic Server and Coherence*.

5.5 Step 4: Create a WebLogic Domain

In the **ndmgr** user's home directory, create a WebLogic domain on the machine which will host the Administration Server only.

Subsequently, when you start the Administration Server, it will use the configuration in the `config` subdirectory of this domain directory to determine the settings for the Administration Server and the domain.

It is likely that most Managed Server instances will be run remotely with respect to the Administration Server. Therefore, these Managed Servers will not have direct access to the domain configuration directory of the Administration Server. Instead they will run from a skeleton domain directory in their respective machine's **ndmgr** home directory and will obtain their configuration over the network on startup from the remotely running Administration Server.

As the **ndmgr** user, create the WebLogic domain.

For example:

1. Run the Configuration Wizard:

   ```
   > ORACLE_HOME/wlserver/common/bin/config.sh
   ```

2. Create a new WebLogic domain based on the default WebLogic Server template.
3. For the Administration Server, specify a fixed IP address (for example, `192.168.1.100`).
4. In Customize Environment and Service Settings, select **Yes**.
5. In Configure Managed Servers, add two Managed Servers, `MS1` and `MS2`.

   For the Managed Servers, specify floating IP addresses (for example, `192.168.1.201` and `192.168.1.202`).
6. In Configure Clusters, add a cluster, `CLUST`, and then assign `MS1` and `MS2` to it.

   Do not specify any Machines or UNIX Machines; you will do this manually in a subsequent step.
7. Name the domain `clustdomain` and save it to `ORACLE_HOME/clustdomain`

5.6 Step 5: Configure nodemanager.domains File

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

This file must contain an entry specifying the domain directory for each domain a 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`.

For the script-based Node Manager, this file's default location is *`WL_HOME`*`/common/nodemanager`, where *`WL_HOME`* is the location in which you installed WebLogic Server, for example, *`ORACLE_HOME`*`/wlserver`.

If you are using the script-based implementation of Node Manager, you must create or copy into *`NodeManagerHome`*, a `nodemanager.domains` file that specifies the domains that you want a Node Manager instance to control. Alternatively, you can register WebLogic domains with the script-based Node Manager using the WLST command, `nmEnroll`. For more information, see Section 4.8.3, "Configuring nodemanager.domains File."

5.7 Step 6: Start the Administration Server

Aside from the standard method of starting the Administration Server by using the provided `startWebLogic` scripts, you can instead have the script-based implementation of Node Manager monitor the Administration Server to ensure high availability. If using a production domain, create `boot.properties` for the Administration Server for successful server startup. As the **ndmgr** user, start the Administration Server locally from a terminal using the `wlscontrol.sh` Node Manager script.

For example:

```
> ORACLE_HOME/wlserver/common/bin/wlscontrol.sh -d clustdomain -r
 ORACLE_HOME/clustdomain -c -f startWebLogic.sh -s AdminServer START
```

For verbose logging to standard out, add the `-v` parameter to the command.

Note: Once the Administration Server has been successfully started, and the `boot.properties` file has been created, you can start the Administration Server remotely. Stop the Administration Server using the `stopWebLogic.sh` Node Manager script. Then, start the Administration Server remotely using SSH.

For example:

```
> ssh -l ndmgr -o PasswordAuthentication=no 192.168.1.100
 ORACLE_HOME/wlserver/common/bin/wlscontrol.sh -d clustdomain -r
/home/ndmgr/clustdomain -c -f startWebLogic.sh -s AdminServer START
```

5.8 Step 7: Configure Node Manager on the Managed Servers

Each machine that will host a Managed Server will have a skeleton domain created and configured.

1. From a local terminal, create a new empty directory (`clustdomain`) in the home directory for the **ndmgr** user for each of the Managed Server host machines and also a back-up machine. For example:

   ```
   > mkdir clustdomain
   ```

2. For each of the Managed Server host machines and the back-up machine, as the **ndmgr** user, use WLST to enroll the user's home directory as being the base directory for remotely run servers and for Node Manager.

For example:

```
> ORACLE_HOME/wlserver/common/bin/wlst.sh

> connect('weblogic','weblogic','t3://192.168.1.100:7001')

> nmEnroll('/home/ndmgr/clustdomain','/home/ndmgr')

> exit()
```

Be sure to run `nmEnroll` on each remote machine. This command creates a property file, `/home/ndmgr/nodemanager.domains`, which maps domain names to home directories, and creates the required domain configuration and security information so that Managed Servers can communicate with the Administration Server.

The `nodemanager.domains` file removes the need to specify the domain home directory (with `-r`) when starting `wlscontrol.sh`. However, since you changed the Node Manager home directory, you must specify `-n /home/ndmgr`. The default Node Manager home directory is `ORACLE_HOME/wlserver/common/nodemanager`; you might not want to use this directory as it is in the product installation directory and owned by another user.

Note: By default, you can start a Node Manager from any directory. A warning will be issued if no `nodemanager.domains` file is found. You must create or copy into *`NodeManagerHome`*, a `nodemanager.domains` file that specifies the domains that you want a Node Manager instance to control or register WebLogic domains using the WLST command, `nmEnroll`.

5.9 Step 8: Test Node Manager Setup and Configuration by Starting Managed Servers

1. Copy the scripts from the Administration Server's domain `bin` directory to the corresponding domain `bin` directory on each Node Manager machine (for example, `/home/ndmgr/bin`). For example:

   ```
   > scp ndmgr@192.168.1.100:"~/clustdomain/bin/*"
   ndmgr@192.168.1.101:~/clustdomain/bin
   ```

2. For each Node Manager machine (including the back-up machine), edit the shell scripts in the `bin` directory to reflect the proper path for the local domain home, and the remote Administration Server's IP address.

   For example:

   a. Edit the *`DOMAIN_HOME`* and `LONG_`*`DOMAIN_HOME`* variables in the `setDomainEnv.sh` script to correctly reflect this remote domain home directory: *`DOMAIN_HOME`*`=/home/ndmgr/clustdomain`

   b. `LONG_`*`DOMAIN_HOME`*`=/home/ndmgr/clustdomain`

   c. Similarly, edit the *`DOMAIN_HOME`* variable in `startWebLogic.sh`

   d. Edit the *`DOMAIN_HOME`* and `ADMIN_URL` (for example, `t3://192.168.1.100:7001`) variables in `startManagedWebLogic.sh`).

3. For each of the Managed Server host machines (including the back-up machine), as the **ndmgr** user, create a `server/security` subdirectory in the domain directory.

   For example, for the Managed Server `MS1`:

```
> mkdir -p ~/clustdomain/servers/MS1/security
```

> **Note:** For the back-up machine, create a server directory for every migratable Managed Server (for example, both `MS1` and `MS2`).

4. Create a new `boot.properties` file with the appropriate user name and password variables specified in each Managed Server's `security` directory (for example, `/home/ndmgr/clustdomain/servers/MS1/security`).

   For example:

   ```
   username=weblogic

   password=password
   ```

   > **Note:** When a Managed Server is first started using the script-based Node Manager, the values in this file will be encrypted.

5. For each of the Managed Server machines, as the **ndmgr** user, start the Managed Server locally from a terminal using the `wlscontrol.sh` Node Manager script.

   For example, to start the Managed Server, `MS1`:

   ```
   > ORACLE_HOME/wlserver/common/bin/wlscontrol.sh -d clustdomain -n
   /home/ndmgr -c -f startManagedWebLogic.sh -s MS1 START
   ```

   For verbose logging to standard out, add the `-v` parameter to the command.

6. Once successfully started, stop the Managed Servers and then, as the **ndmgr** user, attempt to start the Managed Servers remotely using SSH.

   For example to start `MS1`:

   ```
   > ssh -l ndmgr -o PasswordAuthentication=no -p 22 192.168.1.101 ORACLE_
   HOME/wlserver/common/bin/wlscontrol.sh -d clustdomain -n /home/ndmgr -c
   -f startManagedWebLogic.sh -s MS1 START
   ```

7. Once successfully started, stop the Managed Servers again and then repeat the process by trying to start each Managed Server (`MS1`) on the back-up machine instead. Again, stop this server once it successfully starts.

5.10 Step 9: Configure UNIX Machines

Using the WebLogic Server Administration Console, add a new UNIX Machine for each machine which will host an Administration or Managed Server (including the back-up machine) and include the following settings:

Table 5–1 UNIX Machine Settings

Property	Value
O.S. Type	UNIX
Node Manager Type	`SSH`
Node Manager Listen Address	*<primary-ip-address>* (not floating IP address)
Node Manager Listen Port	`22`
Node Manager Home	`/home/ndmgr`

Table 5–1 (Cont.) UNIX Machine Settings

Property	Value
Node Manager Shell Command	`ssh -l ndmgr -o PasswordAuthentication=no -p %P %H ORACLE_HOME/wlserver/common/bin/wlscontrol.sh -d %D -n /home/ndmgr -c -f startManagedWebLogic.sh -s %S %C`
Node Manager Debug Enabled	`true`
Servers	*<name of the Administration or Managed Servers hosted on machine>*

5.11 Step 10: Assign Servers to Machines

Once all of the UNIX Machines are created, use the WebLogic Server Administration Console to set the Machine property for each server, to ensure each server is associated with its corresponding UNIX Machine. See "Assign server instances to machines" in the *Oracle WebLogic Server Administration Console Online Help*.

5.12 Step 11: Start Managed Servers

In the WebLogic Server Administration Console, start each Managed Server. See "Start Managed Servers from the Administration Console" in the *Oracle WebLogic Server Administration Console Online Help*.

Check the server logs in the `/home/ndmgr/clustdomain/servers/managed_server_name/logs` directory of each Managed Server to ensure that the server has started with no errors.

5.13 Configuring Script-Based Node Manager Security

This section includes the following topics:

- Section 5.13.1, "Overriding the Default SSH Port"
- Section 5.13.2, "Configuring Security for WebLogic Server Scripts"
- Section 5.13.3, "Configuring Remote Server Start Security for Script-based Node Manager"
- Section 5.13.4, "Generating and Distributing Key Value Pairs"

5.13.1 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.

5.13.2 Configuring Security for WebLogic Server Scripts

To perform server migration and other tasks, the user-id executing scripts such as `wlscontrol.sh` must have sufficient security permissions. This includes being able to bring an IP address online or take an IP address offline using 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 `wlsifconfig` 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 `wlsifconfig` and `arping` to be run using elevated privileges.

The scripts are located in the *`WL_HOME`*`/common/bin/` directory or the *`DOMAIN_HOME`*`/bin/server_migration` directory. See Step 2: Configure Node Manager Security.

5.13.3 Configuring 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.

5.13.4 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.

- Section 5.13.4.1, "Shared Key Value Pair"
- Section 5.13.4.2, "Individual Key Value Pairs"

5.13.4.1 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="ORACLE_HOME/wlserver/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 Node Manager:

   ```
   $ ssh montgomery wlscontrol.sh VERSION
   ```

   This response indicates that the client accessed Node Manager successfully:

   ```
   NodeManager version 12.1
   ```

5.13.4.2 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.

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

6 Using Node Manager

This chapter describes how to start and stop the Java-based and script-based Node Manager. It also provides information on the recommended procedures for starting server instances using Node Manager.

This chapter includes the following sections:

- Section 6.1, "Starting and Stopping Node Manager"
- Section 6.2, "Using Node Manager to Control Servers"

6.1 Starting and Stopping Node Manager

Use the following methods for starting and stopping Node Manager:

- Section 6.1.1, "Running Node Manager as a Startup Service"
- Section 6.1.2, "Starting Java-based Node Manager Using Scripts"
- Section 6.1.3, "Running Script-based Node Manager"
- Section 6.1.4, "Stopping Node Manager"

6.1.1 Running Node Manager as a Startup Service

Node Manager must run on each computer that hosts a WebLogic Server instance that you want to control with Node Manager. 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.

By default, the operating system service starts up Node Manager to listen on `localhost:5556`. If you want Node Manager to accept commands from remote systems, you must edit the script to listen on a non-localhost listen address.

Oracle recommends 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. See Section 4.8.8, "Configuring Node Manager as an xinetd Service" and Section 4.8.9, "Configuring Node Manager as an init.d Service."

Note: On UNIX platforms, Oracle does not recommend running Node Manager as the root user. However, to achieve Post-Bind GID, you must start Node Manager as the root user. Post-Bind GID enables a server instance running on your machine to bind to a UNIX group ID (GID) after it finishes all privileged startup actions.

On Windows machines, use the following steps to install a per domain Node Manager Windows service:

1. Log in to the machine with Administrator privileges.
2. Open a DOS command prompt window.
3. Change to the *DOMAIN_HOME*\bin directory.
4. Enter the following command:

   ```
   installNodeMgrSvc.cmd
   ```

5. After a few seconds, the following message is displayed:

   ```
   Oracle WebLogic <domain-name> NodeManager installed.
   ```

   The service is installed using the default Node Manager listen port (5556). If this listen port is already in use, the program prompts you to enter a different listen port.

> **Note:** If the Node Manager Windows service is already installed, the following message is displayed instead:
>
> ```
> CreateService failed - The specified service already exists.
> ```

If you want to uninstall a per domain Node Manager Windows service, use the following steps:

1. Log in to the machine with Administrator privileges.
2. Open a DOS command prompt window.
3. Change to the *DOMAIN_HOME*\bin directory.
4. Enter the following command:

   ```
   uninstallNodeMgrSvc.cmd
   ```

5. After a few seconds, the following message is displayed:

   ```
   Oracle WebLogic <domain-name> NodeManager removed.
   ```

By default, NODEMGR_HOST is set to localhost in installNodeMgrSvc.cmd, which means that Node Manager will listen only on the local host. If you do not want Node Manager listening on the local host, set NODEMGR_HOST to a valid hostname or IP address before installing the Node Manager service.

> **Note:** If you select to run a per host Node Manager as a Windows service, using *WL_HOME*\server\bin\installNodeMgrSvc.cmd, you must first perform the prerequisite configuration steps described in Section 4.3, "Configuring Per Host Node Manager".

6.1.2 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 Section 4.8.7, "Set Node Manager Environment Variables."

Sample start scripts for Node Manager are installed in each *DOMAIN_HOME*/bin and the *WL_HOME*\server\bin directory, where *WL_HOME* is the top-level installation directory for

WebLogic Server. However, if you select to use the script in *WL_HOME*\server\bin, you must first perform the prerequisite steps described in Section 4.3, "Configuring Per Host Node Manager".

Use startNodeManager.cmd on Windows systems and startNodeManager.sh on UNIX systems.

The scripts set the required environment variables and start Node Manager in the appropriate *NodeManagerHome* directory. 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.

6.1.2.1 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 created in the *NodeManagerHome* directory. For more information, see Section 4.8.4, "Reviewing nodemanager.properties".

  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.

Note: For UNIX systems:

If you run Node Manager on a UNIX operating system other than Solaris, 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 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 implementation. For more information, see Section 4.8.4, "Reviewing nodemanager.properties".

6.1.3 Running Script-based Node Manager

Note: In this release of WebLogic Server, prior to running the script-based implementation of Node Manager, you must create or copy into *`NodeManagerHome`*, a `nodemanager.domains` file that specifies the domains that you want a Node Manager instance to control. See Section 4.8.3, "Configuring nodemanager.domains File." Alternatively, you can register WebLogic domains with Node Manager using the WLST command, `nmEnroll`.

If not specified, the default *`NodeManagerHome`* location is *`WL_HOME`*`/common/nodemanager`.

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 script-based 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 6–1. The possible parameter values are listed in Table 6–2.

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 d`omains/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
-d	Specifies the domain name
-r	Specifies the domain directory
-c	Enables a server start script
-f	The name of the server start script
-p	The name of the server stop script
-v	Enables verbose output
-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.ListenPort` `22`
%S	WebLogic Server name	`none`
%D	WebLogic domain name	none
%R	Domain directory (server root)	`ServerStartMBean.RootDirectory`

Table 6–2 (Cont.) Shell Command Templates

Parameter	Description	Default
%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

Note: If you notice that it takes a long time to start Node Manager or a WebLogic Server instance, it might be because of low entropy on your machine.

To check entropy:

```
cat /proc/sys/kernel/random/entropy_avail
```

Any result below 500 indicates that your system is at risk of running out of entropy. You can run `rngd` which replenishes random bytes to `/dev/random` using `/dev/urandom` as the source. Start the `rngd` (as root). This will ensure that your system does not run out of entropy:

```
$ rngd -r /dev/urandom -o /dev/random -b
```

You can configure the process by editing `/etc/sysconfig/rngd` with:

```
EXTRA OPTIONS="-i -r /dev/urandom -o /dev/random -b -t 60 -W 2048"
```

Every 60 seconds, this will add bits to the entropy pool until the size is 2048. You can change the interval and size using the `-t 60` and `-W 2048` parameters.

For more information, you can refer to these blogs that describe this topic in detail:

- "WebLogic Server and Entropy" at http://theheat.dk/blog/?p=1539
- "Why Does my Weblogic Server Take a Long Time to Start?" at https://blogs.oracle.com/LuzMestre/entry/why_does_my_weblogic_server

6.1.4 Stopping Node Manager

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

Alternatively, after having set the `nodemanager.properties` attribute `QuitEnabled` to `true` (the default is `false`), you can use WLST to connect to Node Manager and shut it down. Use the `stopNodeManager` command to stop the Node Manager process, as in the following example:

```
wls:/nm/mydomain> stopNodeManager()
Stopped Node Manager Process successfully
wls:/offline>
```

For more information, see `stopNodeManager` in the *WLST Command Reference for WebLogic Server*.

6.2 Using Node Manager to Control Servers

In general, Oracle recommends that you use the WebLogic Scripting Tool and Node Manager to start and stop the Administration Server and Managed Servers. This section describes the recommended procedures for starting server instances using Node Manager and WLST.

For more information, see "Using WLST and Node Manager to Manage Servers" and the "Node Manager Commands" in *WLST Command Reference for WebLogic Server*.

For information about changing the credentials used for starting and stopping WebLogic Server instances, see "Changing the Credentials Used for Starting a Server" in *Administering Server Startup and Shutdown for Oracle WebLogic Server*.

6.2.1 Starting the Administration Server Using Node Manager

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

Establish startup information using the `nmGenBootStartupProps` command. This command generates the Node Manager property files `boot.properties` and `startup.properties`.

You must complete all of the following steps when starting the Administration Server for the first time in your domain, or if you make any configuration changes to the Administration Server.

For subsequent startups of the Administration Server, jump to step 7 to start the Administration Server directly with Node Manager.

1. Start the Administration Server manually by performing one of the following actions:
   - Run the `startWebLogic` script from the *`DOMAIN_HOME`*`/bin` directory, where `DOMAIN_HOME` represents the directory in which your WebLogic Server domain is configured.
   - Invoke WLST and use the WLST `startServer` command to start the Administration Server.
2. (optional) Add any additional configuration information for the Administration Server.
3. Invoke WLST, if not already running.

   On Windows, you can use a shortcut on the Start menu to set your environment variables and invoke WLST.
4. Connect WLST to the Administration Server instance using the `connect` command.
5. Establish startup information using the `nmGenBootStartupProps` command. This command generates the Node Manager property files `boot.properties` and `startup.properties`.
6. Shut down the Administration Server, which also disconnects WLST.
7. Start Node Manager. See Section 6.1, "Starting and Stopping Node Manager".
8. Invoke WLST.

9. Connect WLST to Node Manager using the `nmConnect` command.
10. Start the Administration Server using the `nmStart` command.

    The following example starts the Administration Server in the specified WebLogic domain using Node Manager. In this example, the `prps` variable stores the system property settings and is passed to the command using the `props` argument.

```
wls:/nm/mydomain> prps = makePropertiesObject("AdminURL=http://
listen_address;listen_port;Username=username;Password=password
;weblogic.ListenPort=8001")
wls:/nm/mydomain> nmStart("AdminServer",props=prps)
Starting server AdminServer...
Server AdminServer started successfully
wls:/nm/mydomain>
```

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

Note: Starting the server instance 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 server instances that were started in this way.

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

6.2.2 Starting Managed Servers Using WLST

The following general procedures are recommended for starting a Managed Server using WLST and Node Manager.

1. Start Node Manager. See Section 6.1, "Starting and Stopping Node Manager".
2. Start an Administration Server. See "Starting an Administration Server with a Startup Script" in *Administering Server Startup and Shutdown for Oracle WebLogic Server*.
3. Invoke WLST and connect to an Administration Server using the `connect` command.
4. Start your 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.

6.2.3 Starting Managed Servers Using the Administration Console

The following general procedures are recommended for starting a Managed Server using the WebLogic Server Administration Console:

1. If you have not already done so, create a Managed Server.
2. Node Manager must run on each computer that hosts WebLogic Server instances that you want to control with Node Manager. Configure each computer as a machine in WebLogic Server. See Section 6.2.3.1, "Configuring a Machine to Use Node Manager."

3. Assign each server instance that you will control with Node Manager to the machine upon which it runs. See Section 6.2.3.2, "Assigning Server Instances to a Machine."
4. Start Node Manager on the computer that you want to host the Managed Server. The WebLogic Server custom installation process optionally installs and starts Node Manager as a Windows service on Windows systems. If it's not already running, you can start Node Manager manually at a command prompt or with a script.

For more information, see "Start Managed Servers from the Administration Console" in *Oracle WebLogic Server Administration Console Online Help*.

6.2.3.1 Configuring 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 a 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 WebLogic Server Administration Console. Enter the following values:

1. The DNS name or IP address upon which Node Manager listens in the `Listen Address` field.
2. The port number in the `Listen Port` field. Note that specifying the port number is especially important if you have modified it from the default value.

> **Note:** The Node Manager **Listen Address** value reflects client-side configuration. It tells the Administration Server or clients on the Administration Server how to connect to Node Manager. This attribute does not configure Node Manager.
>
> The listen address you specify must match exactly the host name appearing in the CN component of the Node Manager SSL server digital certificate subject DN.

6.2.3.2 Assigning Server Instances to a Machine

After configuring each computer as a machine resource, you must assign each server instance that you will control with Node Manager to the machine upon which it runs.

1. In the WebLogic Server Administration Console, select **Environment** > **Servers** > ***server_name*** > **Configuration** > **General**.
2. In the **Machine** field, select the machine to which you want to assign the server instance.

> **Note:** You cannot change the machine of the Administration Server using the WebLogic Server Administration Console. You cannot change the cluster or machine of a running server instance.

6.2.4 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. Start Node Manager. See Section 6.1, "Starting and Stopping Node Manager".
2. Invoke WLST and connect to Node Manager using the `nmConnect` command.
3. Start the Managed Server using the WLST `nmStart` command.

> **Note:** If you use the default security providers, the first time you start a Managed Server instance, it must be able to contact the Administration Server.

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:

- In order to start a server instance with `nmStart`, you must ensure that `boot.properties` and `startup.properties` are already defined.
- `nmStart` should not be used to permanently change the startup properties for a server instance. The next time a server instance is migrated or restarted from the Administration Server, these properties will not be used.
- When passing the server instance user name and password using `nmStart`, these values are not encrypted.

The following example starts the `managed1` server in the current WebLogic domain using Node Manager:

```
wls:/nm/mydomain> nmStart("managed1")
Starting server managed1 ...
Server managed1 started successfully
wls:/nm/mydomain>
```