BEA WebLogic Server™

Creating and Configuring WebLogic Server Domains

Release 7.0
Revised: September 4, 2002

Creating and Configuring WebLogic Server Domains

Part Number	Date	Software Version
N/A	September 4, 2002	BEA WebLogic Server Version 7.0

Contents

4. Recovering Failed Servers

About This Document

This document describes how to configure, manage, and monitor resources in a BEA WebLogic Server™ domain.

The document is organized as follows:

- Chapter 1, "Overview of WebLogic Server Domains," introduces the contents of WebLogic Server domains and describes common domain management tasks.
- Chapter 2, "Creating New Domains Using the Configuration Wizard," describes how to create new domains using the Configuration Wizard.
- Chapter 3, "Configuring Network Resources," describes how to configure available network resources for use with servers and clusters in a domain.
- Chapter 4, "Recovering Failed Servers," explains how to prepare for and handle server failures in a domain.
- Chapter 5, "Managing Server Availability with Node Manager," explains how to configure Node Manager to manually and automatically start and stop managed servers in a domain.
- Chapter 6, "Monitoring a WebLogic Server Domain," explains how to monitor managed servers in a domain.

Audience

This document is written for WebLogic Server administrators who want to manage multiple servers or clusters in a domain.

e-docs Web Site

BEA product documentation is available on the BEA corporate Web site. From the BEA Home page, click on Product Documentation.

How to Print the Document

You can print a copy of this document from a Web browser, one main topic at a time, by using the File→Print option on your Web browser.

A PDF version of this document is available on the WebLogic Server documentation Home page on the e-docs Web site (and also on the documentation CD). You can open the PDF in Adobe Acrobat Reader and print the entire document (or a portion of it) in book format. To access the PDFs, open the WebLogic Server documentation Home page, click Download Documentation, and select the document you want to print.

Adobe Acrobat Reader is available at no charge from the Adobe Web site at http://www.adobe.com.

Related Information

The BEA corporate Web site provides all documentation for WebLogic Server. For additional documentation related to WebLogic Server system administration, see the System Administration page.

Contact Us!

Your feedback on BEA documentation is important to us. Send us e-mail at docsupport@bea.com if you have questions or comments. Your comments will be reviewed directly by the BEA professionals who create and update the documentation.

In your e-mail message, please indicate the software name and version you are using, as well as the title and document date of your documentation. If you have any questions about this version of BEA WebLogic Server, or if you have problems installing and running BEA WebLogic Server, contact BEA Customer Support through BEA WebSupport at http://www.bea.com. You can also contact Customer Support by using the contact information provided on the Customer Support Card, which is included in the product package.

When contacting Customer Support, be prepared to provide the following information:

- Your name, e-mail address, phone number, and fax number
- Your company name and company address
- Your machine type and authorization codes
- The name and version of the product you are using
- A description of the problem and the content of pertinent error messages

Documentation Conventions

The following documentation conventions are used throughout this document.

Convention	Usage
Ctrl+Tab	Keys you press simultaneously.
italics	Emphasis and book titles.

Convention	Usage

`monospace text`	Code samples, commands and their options, Java classes, data types, directories, and file names and their extensions. Monospace text also indicates text that you enter from the keyboard.
	Examples:
	`import java.util.Enumeration;`
	`chmod u+w *`
	`config/examples/applications`
	`.java`
	`config.xml`
	`float`

`monospace italic text`	Variables in code. *Example*: `String` *`CustomerName`*`;`
UPPERCASE TEXT	Device names, environment variables, and logical operators. *Examples*: LPT1 BEA_HOME OR
`{ }`	A set of choices in a syntax line.

`[ ]`	Optional items in a syntax line. *Example*:
	`java utils.MulticastTest -n` *`name`* `-a` *`address`*
	`[-p` *`portnumber`*`] [-t` *`timeout`*`] [-s` *`send`*`]`

`|`	Separates mutually exclusive choices in a syntax line. *Example*:
	`java weblogic.deploy [list|deploy|undeploy|update]`
	`password {application} {source}`

`...`	Indicates one of the following in a command line: ■ An argument can be repeated several times in the command line. ■ The statement omits additional optional arguments. ■ You can enter additional parameters, values, or other information

Convention	Usage
. . .	Indicates the omission of items from a code example or from a syntax line.

CHAPTER

1 Overview of WebLogic Server Domains

The following sections describe WebLogic Server domains and their contents:

What Is a Domain?

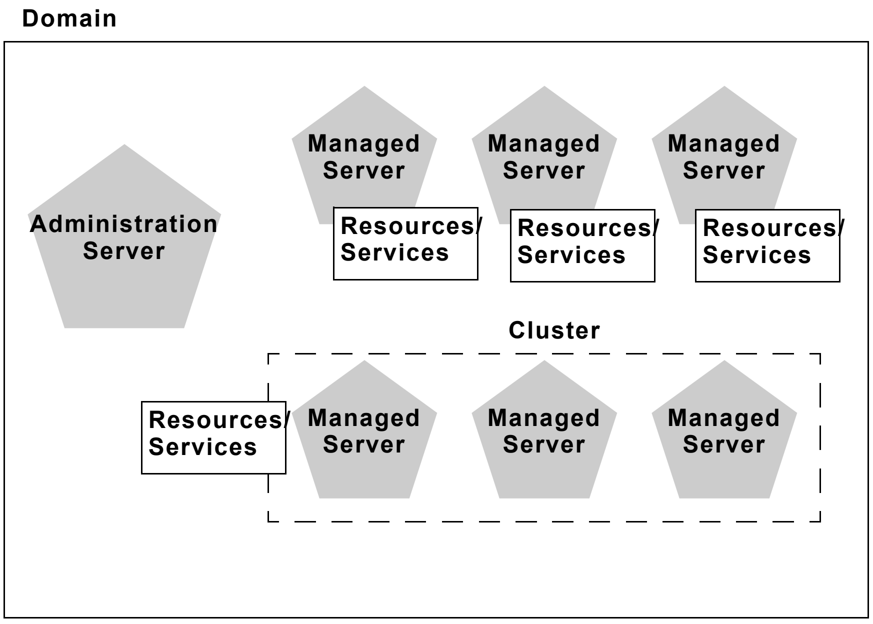
A *domain* is the basic administration unit for WebLogic Server instances. A domain consists of one or more WebLogic Server instances (and their associated resources) that you manage with a single Administration Server. You can define multiple domains based on different system administrators' responsibilities, application boundaries, or geographical locations of servers. Conversely, you can use a single domain to centralize all WebLogic Server administration activities.

If you create multiple domains, keep in mind that each domain is represented in a separate configuration file (`config.xml`), and you cannot perform configuration or deployment tasks in multiple domains at the same time. When you use the Administration Console to perform a configuration task, the changes you make apply to the currently selected domain. To make changes in a different domain, you must select that domain. For this reason, the servers instances, applications, and resources in one domain should be treated as being independent of servers, applications, and resources in a different domain.

Contents of a Domain

A domain can include multiple WebLogic Server clusters and non-clustered WebLogic Server instances. Strictly speaking, a domain could consist of only one WebLogic Server instance—however, in that case that sole server instance would be an Administration Server, because each domain must have exactly one Administration Server. Although the scope and purpose of a domain can vary significantly, most WebLogic Server domains contain the components described in this section.

The following figure shows a production environment that contains an Administration Server, three standalone Managed Servers, and a cluster of three Managed Servers.



Administration Server

Each WebLogic Server domain must have one server instance that acts as the Administration Server. You use the Administration Server, programmatically or via the Administration Console, to configure all other server instances and resources in the domain.

Role of the Administration Server

Before you start the Managed Servers in a domain, you start the Administration Server. When you start a standalone or clustered Managed Server, it contacts the Administration Server for its configuration information. In this way, the Administration Server operates as the central control entity for the configuration of the entire domain.

You can invoke the services of the Administration Server in the following ways:

- Domain Configuration Wizard—The Domain Configuration Wizard is the recommended tool for creating a new domain or cluster.
- WebLogic Server Administration Console—The Administration Console is a graphical user interface (GUI) to the Administration Server.
- WebLogic Server Application Programming Interface (API)—You can write a program to modify configuration attributes using the API provided with WebLogic Server.
- WebLogic Server command-line utility—This utility allows you to create scripts to automate domain management.

Whichever method is used, the Administration Server for a domain must be running to modify the domain configuration.

When the Administration Server starts, it loads the `config.xml` for the domain. It looks for `config.xml` in the current directory. Unless you specify another directory when you create a domain, `config.xml` is stored in:

```
BEA_HOME/user_projects/mydomain
```

where `mydomain` is a domain-specific directory, with the same name as the domain.

Each time the Administration Server starts successfully, a backup configuration file named `config.xml.booted` is created in the domain directory. In the unlikely event that the `config.xml` file should be corrupted during the lifetime of the server instance, it is possible to revert to this previous configuration.

For more information about the Administration Server and its role in the WebLogic Server JMX management system, see "System Administration Tools" in the *Administration Guide.*

What Happens if the Administration Server Fails?

The failure of an Administration Server for a domain does not affect the operation of Managed Servers in the domain. If an Administration Server for a domain becomes unavailable while the server instances it manages—clustered or otherwise—are up and running, those Managed Servers continue to run. If the domain contains clustered server instances, the load balancing and failover capabilities supported by the domain configuration remain available, even if the Administration Server fails.

If an Administration Server fails because of a hardware or software failure on its host machine, other server instances on the same machine may be similarly affected. However, the failure of an Administration Server itself does not interrupt the operation of Managed Servers in the domain.

For instructions on re-starting an Administration Server, see "Restarting an Administration Server When Managed Servers are Running" on page 4-9 .

Managed Servers and Clustered Managed Servers

In a domain, server instances other than the Administration Server are referred to as Managed Servers. Managed Servers host the components and associated resources that constitute your applications—for example, JSPs and EJBs. When a Managed Server starts up, it connects to the domain's Administration Server to obtain configuration and deployment settings.

Note: Managed Servers in a domain can start up independently of the Administration Server if the Administration Server is unavailable. See "Recovering Failed Servers" on page 4-1 for more information.

Two or more Managed Servers can be configured as a WebLogic Server cluster to increase application scalability and availability. In a WebLogic Server cluster, most resources and services are deployed to each Managed Server (as opposed to a single

Managed Server,) enabling failover and load balancing. To learn which component types and services can be clustered—deployed to all server instances in a cluster—see "What Types of Objects Can Be Clustered?" in *Using WebLogic Server Clusters.*

You can create a non-clustered Managed Server and add it to a cluster by configuring pertinent configuration parameters for the server instance and the cluster. Conversely, you can remove a Managed Server from a cluster by re-configuring the parameters appropriately. The key difference between clustered and non-clustered Managed Servers is support for failover and load balancing—these features are available only in a cluster of Managed Servers.

Your requirements for scalability and reliability drive the decision on whether or not to cluster Managed Servers. For example, if your application is not subject to variable loads, and potential interruptions in application service are acceptable, clustering may be unnecessary.

For more information about the benefits and capabilities of a WebLogic Server cluster, see "Introduction to WebLogic Server Clustering" in *Using WebLogic Server Clusters* A single domain can contain multiple WebLogic Server clusters, as well as multiple Managed Servers that are not configured as clusters.

Resources and Services

In addition to the Administration Server and Managed Servers, a domain also contains the resources and services required by Managed Servers and hosted applications deployed in the domain.

Examples of domain-level resources include:

- Machine definition identify a particular, physical piece of hardware. A Machine definition is used to associate a computer with the Managed Server(s) it hosts. This information is used by Node Manager in restarting a failed Managed Server, and by a clustered Managed Server in selecting the best location for storing replicated session data. For more information about Node Manager, see "Managing Server Availability with Node Manager" on page 5-1 .
- Network channels, an optional resource that can be used to define default ports, protocols, and protocol settings. After creating a network channel, you can assign it to any number of Managed Servers and clusters in the domain. See"Configuring Network Resources" on page 3-1 for more information.

Managed Servers in the domain host their own resources and services. You can deploy resources and services to selected Managed Servers or to a cluster. Examples of deployable resources include:

- application components, such as EJBs
- connectors, startup classes,
- JDBC connection pools,
- JMS servers

Common Domain Types

There are two basic types of domains:

- **Domain with Managed Servers**: A simple production environment can consist of a domain with several Managed Servers that host applications, and an Administration Server to perform management operations. In this configuration, applications and resources are deployed to individual Managed Servers; similarly, clients that access the application connect to an individual Managed Server.

  Production environments that require increased application performance, throughput, or availability may configure two or more of Managed Servers as a cluster. Clustering allows multiple Managed Servers to operate as a single unit to host applications and resources. For more information about the difference between a standalone and clustered Managed Servers, see "Managed Servers and Clustered Managed Servers" on page 1-4.

- **Standalone Server Domain**: For development or test environments, you may want to deploy a single application and server independently from servers in a production domain. In this case, you can deploy a simple domain consisting of a single server instance that acts as an Administration Server that, and also hosts the applications you are developing. The examples domain that you can install with WebLogic Server is an example of a standalone server domain.

Note: In production environments, BEA recommends that you deploy applications only on Managed Servers in the domain; the Administration Server should be reserved for management tasks.

Domain Restrictions

Many WebLogic Server installations consist of a single domain that includes all the Managed Servers required to host applications. If you create more than one domain, note the following restrictions:

- Each domain requires its own Administration Server for performing management activities. When you are using the Administration Console to perform management and monitoring tasks, you can switch back and forth between domains, but in doing so, you are connecting to different Administration Servers.
- All Managed Servers in a cluster must reside in the same domain; you cannot "split" a cluster over multiple domains.
- You cannot share a configured resource or subsystem between domains. For example, if you create a JDBC connection pool in one domain, you cannot use it with a Managed Server or cluster in another domain. (Instead, you must create a similar connection pool in the second domain.)

Domain Directory and config.xml

The configuration of a domain is stored in the `config.xml` file for the domain. `config.xml` specifies the name of the domain and the configuration parameter settings for each server instance, cluster, resource, and service in the domain. The `config.xml` for a domain is stored in the domain directory, which you specify when you create the domain.

The domain directory also contains default script files that you can use to start the Administration Server and Managed Servers in the domain. For more information about these scripts and other methods of starting a server instance, see "Starting and Stopping WebLogic Server" in the *Administration Guide*.

Domain Directories Structure

In releases prior to WebLogic Server 7.0, domain directories were created within the directory structure of the Weblogic Server installation. With WebLogic Server 7.0 and later, you can set up domain directories outside the product installation directory tree, in any location on the system that can access the Weblogic Server installation and the JDK.

This new directory structure is more flexible. It allows you to store your application code in a directory structure separate from WebLogic Server executables and related files—this practice is recommended.

The domain directory structure should have:

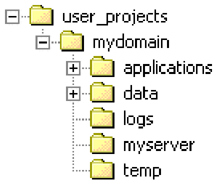
- A root directory with the same name as the domain, such as `mydomain` or `petstore`. This directory should contain the following:
  - The configuration file (usually `config.xml`) for the domain.
  - Any scripts you use to start server instances and establish your environment.
- A subdirectory for storing applications for the domain, typically named `applications`.

Note: If you plan to use the WebLogic Server's auto-deployment feature—available when a domain is running in development mode—the subdirectory for applications *must* be named `applications`. For information about auto-deployment, see "Auto-Deployment" in *Developing WebLogic Server Applications*.

You can create other directories within the domain directory structure, as desired. When you start a server instance in a domain for the first time, Weblogic Server creates the following subdirectories in the domain directory:

- `data` for storing security information
- `logs` for storing domain-level logs
- *`server_name`* for each server running in the domain, for storing server-level logs
- `temp` for storing temporary files

For example:



You can use the Configuration Wizard to create and configure domains. At the end of a custom installation, you are prompted to run the Configuration Wizard. You can also start the Configuration Wizard from the Start menu or by using a script. For more information, see "Creating New Domains Using the Configuration Wizard" on page 2-1.

When you use the Configuration Wizard to create a domain, it creates the `user_projects` directory in the BEA Home directory as a container for your domains. It also creates a root directory for the new domain and any other directories specified in the domain template that you select to create the domain.

Domain Administration Tasks

Refer to the following sections to learn about common domain administration tasks:

- "Creating New Domains Using the Configuration Wizard" on page 2-1
- "Configuring Network Resources" on page 3-1
- "Recovering Failed Servers" on page 4-1
- "Managing Server Availability with Node Manager" on page 5-1

CHAPTER

2 Creating New Domains Using the Configuration Wizard

The following sections describe how to create WebLogic Server domains using the Configuration Wizard:

- "Introduction to the Configuration Wizard" on page 2-1
- "Determine the Server Name and Listen Address" on page 2-6
- "Using the Configuration Wizard" on page 2-8

Introduction to the Configuration Wizard

The Configuration Wizard is a standalone application—you do not have to start WebLogic Server to run the Configuration Wizard—that helps you create new, customized WebLogic Server domains. It is automatically installed when you install WebLogic Server.

At the end of a Custom Installation of WebLogic Server, you have the option, but are not required, to run the Configuration Wizard. You can also invoke the Configuration Wizard any time after an installation using the instructions in "Starting the Configuration Wizard" on page 2-8.

The Configuration Wizard presents dialogs that help you create or extend domains by adding individual servers. You begin by selecting a domain template, which defines rough characteristics for the new domain such as:

- Whether the domain includes the pre-configured WebLogic Server examples application or the Pet Store application.
- The number of Managed Servers to configure for the domain.
- Whether you want to configure the new Managed Servers as a cluster.

The Configuration Wizard eases the process of creating domains using underlying templates and dialogs that allow you to specify the characteristics of your domain.

What Does the Configuration Wizard Create?

Based on the responses you provide to the dialogs presented by the Configuration Wizard, the wizard creates a `config.xml` file for the domain. The Configuration Wizard also creates startup scripts for the server instances in the domain, and other helper files and directories to help you start and use the new domain and its servers.

The Configuration Wizard stores the `config.xml` file and all other generated components in a domain directory that you specify during the dialogs (by default `/user_projects/mydomain`).

The following table describes the files and directories that the Configuration Wizard creates.

Table 2-1 Components Installed with Configuration Wizard

Component	Function
`config.xml`	The config.xml file is an XML document that describes the configuration of a WebLogic Server domain. The content and structure of the config.xml is defined in the associated Document Type Definition (DTD), config.dtd. config.xml consists of a series of XML elements. The Domain element is the top-level element, and all elements in the Domain descend from the Domain element. The Domain element includes child elements, such as the Server, Cluster, and Application elements. These child elements may have children of their own. For example, the Server element includes the child elements WebServer, SSL and Log. The Application element includes the child elements EJBComponent and WebAppComponent. Each element has one or more configurable attributes. An attribute defined in config.dtd has a corresponding attribute in the configuration API. For example, the Server element has a ListenPort attribute, and likewise, the weblogic.management.configuration.ServerMBean has a ListenPort attribute. Configurable attributes are readable and writable, that is, ServerMBean has a getListenPort and a setListenPort method. To learn more about config.xml, see *BEA WebLogic Server Configuration Reference*
`/applications`	Stores the default Web Application for servers in the domain. If you select a template that includes WebLogic Server examples, then this directory also stores JAR and EAR files for individual examples. Note that if you select a template that installs WebLogic Server examples or the Petstore application, the `examplesWebApp` and `petstore` application files remain in the `\samples\server\stage` subdirectory of the WebLogic Server installation directory.
`/logs`	Stores log files for servers in the domain.

Component	Function
`setEnv.cmd`, `setEnv.sh`	Set environment variables for domain servers.
`setExamplesEnv.cmd`, `setExamplesEnv.sh`	Set environment variables for domains that include the WebLogic Server example applications.
`startWebLogic.cmd`, `startWebLogic.sh`	Start the Administration Server for a custom domain.
`startManagedWebLogic.cmd`, `startManagedWebLogic.sh`	Start a Managed Server in a custom domain.
`startExamplesServer.cmd`, `startExamplesServer.sh`	Start a server that hosts the WebLogic Server example applications.
`startPetStore.cmd`, `startPetStore.sh`	Start a server that hosts the Pet Store example applications.
`demokey.pem`, `democert.pem`	Provide sample SSL protocol support for servers in the domain.
Windows Start Menu Items	Provide Start menu support for starting domain servers in a Windows environment.

Understanding Configuration Wizard Templates

The templates available in the Configuration Wizard vary, based on which WebLogic Platform components are installed in your environment. This section describes the templates typically available in a WebLogic Server installation. If you have other WebLogic Platform components installed, other templates may be available in the Configuration Wizard. For information about other templates, see *Configuration Wizard Template Reference*.

The Configuration Wizard templates for WebLogic Server installations are organized into the following main categories:

- WLS Domain— These templates create a new domain that contains one server instance and no applications.
- WLS Examples—These templates create a new domain that contains one server instance. The server instance runs as an Administration Server and is configured

to run the WebLogic Server example applications. This server is similar to the `examplesServer` created when you choose the Typical Install option in the WebLogic Server installation program.

- WLS Petstore—These templates create a new domain that contains one server instance. The server instance runs as an Administration Server and is configured to run the Pet Store sample application. This server is similar to the `petstoreServer` created with a typical WebLogic Server installation.

If you install other BEA products, the Configuration Wizard provides additional templates.

Within each template category, the Configuration Wizard offers several different domain configurations, which vary in terms of the number and type of server instances the new domain will contain The alternative configurations are:

- Single Server (Standalone Server)—This option creates a new domain with a single server that acts as the Administration Server and also hosts applications.

  This type of configuration is recommended for development or testing, not for a production environment. See "Create a Domain with a Single Server Instance" on page 2-15 for instructions on creating a domain with a single server.

- Admin Server with Managed Server(s)—This option creates a new domain with an Administration Server and one or more Managed Servers.

  This type of configuration is recommended for production environments, as it provides a dedicated Administration Server, and Managed Servers to host applications. See "Create a Domain with Administration Server and Stand-Alone Managed Servers" on page 2-10 for instructions on creating a domain with multiple servers.

- Admin Server with Clustered Managed Server(s)—This option creates a new domain with an Administration Server and multiple clustered Managed Servers.

  This type of configuration is recommended for production environments that require high availability and reliability. It provides a dedicated Administration Server, and a cluster of Managed Servers to support failover and load balancing for the applications it hosts. See "Create a Domain with Administration Server and Clustered Managed Servers" on page 2-12 for instructions on creating a domain multiple servers arranged as a cluster.

- Managed Server (with owning Admin Server configuration)—This option creates startup scripts and other files required to run an existing Managed Server on a WebLogic Server host machine that is separate from the host on which the Administration Server runs.

See "Configure Support for Remote Managed Servers" on page 2-16 for more information.

Determine the Server Name and Listen Address

The Configuration Wizard prompts you to assign a name and identify the listen address for each server instance you create. The following sections describe important considerations related to setting the server name and listen address.

Server Name Considerations

Each server instance in your WebLogic environment must have a unique name, regardless of the domain or cluster in which it resides, or whether it is an Administration Server or a Managed Server.

Listen Address Considerations

The following table describes important considerations related to setting the listen address value.

If the Listen Address is set to . . .	**Then, the following is true . . .**
IP address or DNS name	■ To connect to the server instance, processes can specify either the IP address or the corresponding DNS name. ■ Processes that specify `localhost` will fail to connect. ■ You must update existing processes that use `localhost` to connect to the server instance. ■ Connections that specify the IP address for the listen address and a secured port for the listen port must disable host name verification. **Note:** To resolve a DNS name to an IP address, Weblogic Server must be able to contact an appropriate DNS server or obtain the IP address mapping locally. Therefore, if you specify a DNS name for the listen address, you must either leave a port open long enough for the WebLogic Server instance to connect to a DNS server and cache its mapping or you must specify the IP address mapping in a local file. If you specify an IP address for ListenAddress and then a client request specifies a DNS name, WebLogic Server will attempt to resolve the DNS name, but if it cannot access DNS name mapping, the request will fail.
`localhost`	■ Processes must specify `localhost` to connect to the server instance. ■ Only processes that reside on the machine that hosts the server instance (local processes) will be able to connect to the server instance. ■ Remote (non-local) processes will not be able to connect to the server instance.

If the Listen Address is set to . . .	Then, the following is true . . .
Undefined or Blank (“”)	■ Processes can specify the IP address, DNS name, or `localhost` to connect to the server instance. ■ Processes that specify `localhost` must reside on the machine that hosts the server instance. ■ If the server instance must be accessible as `localhost` (for instance, if you have administrative scripts that connect to `localhost`), and must also be accessible by remote processes, leave the listen address blank. **Note:** For WebLogic servers running on multi-homed Windows NT machines, you should not leave the listen address value undefined or blank. (Multi-homed machines are configured with multiple IP addresses.) Otherwise, the WebLogic Server reserves and listens on its port for each of the machine IP addresses. This precludes other servers from using the same port on the machine.

Using the Configuration Wizard

The following sections provide instructions for starting and using the Configuration Wizard.

Starting the Configuration Wizard

The WebLogic Server installation program provides the option to automatically start the Configuration Wizard when you select the Custom Install option during the installation. You can also start the Configuration Wizard at any time after installing WebLogic Server using either a GUI or a console (command-line) interface, as described below.

Starting in GUI Mode

Running the Configuration Wizard in GUI mode executes the Configuration Wizard program in a graphics environment that can run on Windows and some Unix systems.

To start the Configuration Wizard in GUI mode on a Windows platform, select the Run Configuration Wizard option from the BEA program group in the Windows Start Menu:

```
Start→Programs→BEA WebLogic Platform 7.0→Domain Configuration
Wizard
```

To start the Configuration Wizard in GUI mode on a UNIX platform (or from a Windows command prompt):

1. Log in to a Windows or UNIX system on which the WebLogic Server software is installed.
2. Open a command-line shell.
3. Go to the following directory: `WL_HOME/common/bin`

   where `WL_HOME` is the directory in which you installed WebLogic Server. For example:

   ```
   cd c:\bea\weblogic700\common\bin
   ```
4. Invoke the `dmwiz.cmd` or `dmwiz.sh` script.

If you try to start the Configuration Wizard on a system that cannot support the graphical display, the wizard automatically starts in console mode.

Starting in Console Mode

Running the Configuration Wizard in Console mode executes the Configuration Wizard program in a text-based environment. To start the Configuration Wizard in Console mode:

1. Log in to the target Windows or UNIX system.
2. Open a command-line shell.
3. Go to the following directory: `WL_HOME/common/bin`

   where `WL_HOME` is the directory in which you installed WebLogic Server. For example:

```
cd ~/bea/weblogic700/common/bin
```

4. Invoke the `dmwiz.cmd` or `dmwiz.sh` script with the `-mode=console` argument. For example, in a `bash` shell on UNIX:

   ```
   . dmwiz.sh -mode=console
   ```

Create a Domain with Administration Server and Stand-Alone Managed Servers

To create a new domain with one or more Managed Servers and a standalone Administration Server:

1. Start the Configuration Wizard using the instructions in "Starting the Configuration Wizard" on page 2-8. The instructions that follow assume that you are running the Configuration Wizard in GUI mode.

   The Configuration Wizard displays the Choose Domain Type and Name screen.

2. Perform the following actions:
   - **Select a Template:** Select WLS Domain, WLS Examples, or WLS Petstore. See "Understanding Configuration Wizard Templates" on page 2-4 for more information.
   - **Name:** Enter a domain name using alphanumeric characters. The Configuration Wizard uses the name you enter to create a domain subdirectory for the new domain.

     This field will not accept spaces. In addition, "Portal" cannot be used as a domain name.

3. Click the Next button to continue to the Choose Server Type screen.

4. Select the Admin Server with Managed Server(s) option and click Next. The Configuration Wizard displays the Choose Domain Location screen.

5. Enter a top-level directory to store your custom domain, or use the Browse button to select a directory. Click Next to move to the Configure Managed Servers in Admin Server screen.

6. To add a new Managed Server to the domain, click Add and fill in the fields of the Add Server dialog box as follows:

- Server Name—Enter a server name using alphanumeric characters. This field will not accept spaces.

  See "Server Name Considerations" on page 2-6.

- Listen Address—Enter an IP address for this server.

  See "Listen Address Considerations" on page 2-6.

- Listen Port—Enter a numeric value for the listen port. The range is 1 to 65535.

7. Click Add on the Add Server dialog box to add the new Managed Server and return to the Configure Managed Servers in Admin Server screen.

   Note: If you make a mistake and want to edit or delete a server you added, select the server name and click the Edit or Delete button.

8. Repeat step 6 to add additional Managed Servers, or click Next to move to the Configure Admin Server screen.

9. Fill in the fields of the Configure Admin Server screen as follows:

   - **Server Name:** Enter a server name using alphanumeric characters. This field will not accept spaces.

     See "Server Name Considerations" on page 2-6.

   - **Server Listen Address:** Enter an IP address for this server.

     See "Listen Address Considerations" on page 2-6.

   - **Server Listen Port:** Enter a numeric value for the listen port. The range is 1 to 65535. The default port is 7001.

   - **Server SSL Listen Port:** Enter a numeric value for SSL listening port for your security configuration. The range is 1 to 65535. The default port is 7002.

10. Click Next to move to the Create System User and Password screen.

11. Enter the user name and password required to boot and connect to the Administration Server you configured. Click Next.

12. For Windows systems, the Configuration Wizard prompts you to install the Administration Server as a Windows service. Select Yes if you want to run the server as a Windows service, or No to start the server from the command-line or windows Start menu. Click Next to continue.

13. For Windows systems, the Configuration Wizard prompts you to install the domain in the Windows Start menu. Select Yes if you want to install the server start scripts under the start menu. Click Next to move to the Configuration Summary screen.

14. Verify the new domain and server configuration in the Configuration Summary screen. If you want to edit your selections, click the back button to return to the correct screen. Otherwise, click Create to create the new domain with the servers you specified.

Create a Domain with Administration Server and Clustered Managed Servers

For guidelines on supplying addressing information for a cluster and its members, see "Identify Names and Addresses" in *Using WebLogic Server Clusters*.

To create a new domain with a cluster of Managed Servers and a standalone Administration Server:

1. Start the Configuration Wizard using the instructions in "Starting the Configuration Wizard" on page 2-8. The instructions that follow assume that you are running the Configuration Wizard in GUI mode.

   The Configuration Wizard displays the Choose Domain Type and Name screen.

2. Perform the following actions:

   - **Select a Template:** Select WLS Domain, WLS Examples, or WLS Petstore. See "Understanding Configuration Wizard Templates" on page 2-4 for more information.
   - **Name:** Enter a domain name using alphanumeric characters. The Configuration Wizard uses the name you enter to create a domain subdirectory for the new domain.

     This field will not accept spaces. In addition, "Portal" cannot be used as a domain name.

3. Click the Next button to continue to the Choose Server Type screen.

4. Select the Admin Server with Clustered Managed Server(s) option and click Next. The Configuration Wizard displays the Choose Domain Location screen.

5. Enter a top-level directory to store your custom domain, or use the Browse button to select a directory. Click Next to move to the Configure Clustered Servers screen.

6. To add a new Managed Server to the domain, click Add and fill in the fields of the Add Server dialog box as follows:

   Server Name: Enter a server name using alphanumeric characters. This field will not accept spaces.

   See "Server Name Considerations" on page 2-6.

   - **Listen Address:** Enter an IP address for this server.

     See "Listen Address Considerations" on page 2-6.

   - **Listen Port:** Enter a numeric value for the listen port. The range is 1 to 65535.

7. Click Add on the Add Server dialog box to add the new Managed Server and return to the Configure Managed Servers in Admin Server screen.

   Note: If you make a mistake and want to edit or delete a server you added, select the server name and click the Edit or Delete button.

8. Repeat step 6 to add additional Managed Servers, or click Next to move to the Configure Cluster screen.

9. Enter the following information in the Configure Cluster screen:

   - **Cluster Name:** Enter a cluster name using alphanumeric characters. This field will not accept spaces. The default is mycluster.

     The name of the cluster must be unique among all configuration component names within the domain.

   - **Cluster MultiCast Address:** Enter a multicast address for the cluster. A multicast address is an IP address in the range from 224.0.0.0 to 239.255.255.255.

   - **Cluster MultiCast Port:** Enter a numeric value for the multicast port. The range of values is up to 65535.

   - **Cluster Address:** Enter the address that clients use to connect to this cluster. For production use, enter a DNS name that maps to the individual IP addresses of the Managed Servers in the cluster. For testing or development purposes, use a comma-separated list of the IP addresses and ports assigned to the Managed Servers (this is the default entry).

10. Click Next to move to the Configure Admin Server (with Cluster) screen.
11. Fill in the fields of the Configure Admin Server (with Cluster) screen as follows:
    - **Server Name:** Enter a server name using alphanumeric characters. This field will not accept spaces.

      See "Server Name Considerations" on page 2-6.
    - **Server Listen Address:** Enter an IP address for this server.

      See "Listen Address Considerations" on page 2-6.
    - **Server Listen Port:** Enter a numeric value for the listen port. The range is 1 to 65535. The default port is 7001.
    - **Server SSL Listen Port:** Enter a numeric value for SSL listening port for your security configuration. The range is 1 to 65535. The default port is 7002.
12. Click Next to move to the Create System User and Password screen.
13. Enter the user name and password required to boot and connect to the Administration Server you configured. Click Next.
14. For Windows systems, the Configuration Wizard prompts you to install the Administration Server as a Windows service. Select Yes if you want to run the server as a Windows service, or No to start the server from the command-line or windows Start menu. Click Next to continue.
15. For Windows systems, the Configuration Wizard prompts you to install the domain in the Windows Start menu. Select Yes if you want to install the server start scripts under the start menu. Click Next to move to the Configuration Summary screen.
16. Verify the new domain and server configuration in the Configuration Summary screen. If you want to edit your selections, click the back button to return to the correct screen. Otherwise, click Create to create the new domain with the servers you specified.

Create a Domain with a Single Server Instance

To create a new domain with a single WebLogic Server instance that acts as both the Administration Server and application host server:

1. Start the Configuration Wizard using the instructions in "Starting the Configuration Wizard" on page 2-8. The instructions that follow assume that you are running the Configuration Wizard in GUI mode.

   The Configuration Wizard displays the Choose Domain Type and Name screen.

2. Perform the following actions:

   - **Select a Template:** Select WLS Domain, WLS Examples, or WLS Petstore. See "Understanding Configuration Wizard Templates" on page 2-4 for more information.
   - **Name:** Enter a domain name using alphanumeric characters. The Configuration Wizard uses the name you enter to create a domain subdirectory for the new domain.

     This field will not accept spaces. In addition, "Portal" cannot be used as a domain name.

3. Click the Next button to continue to the Choose Server Type screen.

4. Select the Single Server (Standalone Server) option and click Next. The Configuration Wizard displays the Choose Domain Location screen.

5. Enter a top-level directory to store your custom domain, or use the Browse button to select a directory. Click Next to move to the Configure Single Server screen.

6. Fill in the fields of the Configure Single Server screen as follows:

   - **Server Name:** Enter a server name using alphanumeric characters. This field will not accept spaces.

     See "Server Name Considerations" on page 2-6.
   - **Server Listen Address:** Enter an IP address for this Server.

     See "Listen Address Considerations" on page 2-6.
   - **Server Listen Port:** Enter a numeric value for the listen port. The range of values is 1 to 65535. The default port is 7001.

- **Server SSL Listen Port:** Enter a numeric value for SSL listen port. The range of values is 1 to 65535. The default port is 7002.

7. Click Next to move to the Create System User and Password screen.
8. Enter the user name and password required to boot and connect to the server you configured. Click Next.
9. For Windows systems, the Configuration Wizard prompts you to install the Administration Server as a Windows service. Select Yes if you want to run the server as a Windows service, or No to start the server from the command-line or windows Start menu. Click Next to continue.
10. For Windows systems, the Configuration Wizard prompts you to install the domain in the Windows Start menu. Select Yes if you want to install the server start script under the Start menu. Click Next to move to the Configuration Summary screen.
11. Verify the new domain and server configuration in the Configuration Summary screen. If you want to edit your selections, click the back button to return to the correct screen. Otherwise, click Create to create the new domain with the standalone server you specified.

Configure Support for Remote Managed Servers

To enhance performance and reliability within a WebLogic domain, you can run WebLogic Server instances on different computers (machines). For example, you can run the Administration Server on a computer named `MachineA`, a Managed Server named `MS1` on `MachineB`, and a Managed Server named `MS2` on `MachineC`. To run a WebLogic Server instance on a machine, you must:

- Install the WebLogic Server software.
- For an Administration Server, use the Configuration Wizard to create a `config.xml` file and startup script.
- For a Managed Server, use the Configuration Wizard to create a startup script that specifies the location of the Administration Server. When the Managed Server starts, it refers to the Administration Server for its configuration data.

For information on configuring a Managed Server to start when the Administration Server is unavailable, refer to "Starting a Managed Server When the Administration Server Is Not Accessible" on page 4-11.

For each Managed Server that you want to run on a WebLogic Server host that is separate from the host on which the domain's Administration Server runs:

1. Make sure the domain's `config.xml` file already specifies configuration data for the Managed Server.

   You can specify this information when you create a domain (see "Create a Domain with Administration Server and Stand-Alone Managed Servers" on page 2-10 or "Create a Domain with Administration Server and Clustered Managed Servers" on page 2-12) or after you have created a domain (see "Adding a Server to the Domain" in the *Administration Console Help*).

2. Log in to the machine that will run the new Managed Server.

   Note that this machine must have access to the WebLogic Server installation files in order to run the Configuration Wizard executable.

3. Start the Configuration Wizard using the instructions in "Starting the Configuration Wizard" on page 2-8. The instructions that follow assume that you are running the Configuration Wizard in GUI mode.

   The Configuration Wizard displays the Choose Domain Type and Name screen.

4. Perform the following actions:

   - **Select a Template:** Select the template that was used to create the domain in which you have already defined the Managed Server.
   - **Name:** Enter the name of the domain in which you have already defined the Managed Server.

     Click the Next button to move to the Choose Server Type screen.

5. Select the Managed Server (with owning Admin Server configuration) option and click Next. The Configuration Wizard displays the Choose Domain Location screen.

6. Enter the name of the directory where you want to store the start scripts and demo security files for the Managed Server, or use the Browse button to select a directory. Click Next to move to the Configure Administrative Server Connection screen.

7. Provide information that the Managed Server uses to connect to the Administration Server:
   - **Admin Server Listen Address:** Enter the DNS name or the IP address of the computer that hosts the Administration Server.
   - **Admin Server Listen Port:** Enter the port number on which you configured the Administration Server to listen for non-SSL requests.
   - **Admin Server SSL Listen Port:** Enter the port number on which you configured the Administration Server to listen for SSL requests.
   - **Managed Server Name:** Enter the name of the Managed Server (this name must match the name that is specified for the Managed Server in the domain's `config.xml` file).
8. Click Next to move to the Configure Standalone/Administrative Server page.
9. Ignore all values on the Configure Standalone/Administrative Server page. This page is not used to configure support for remote Managed Servers.
10. Click Next to move to the Create System User and Password screen.
11. Enter the user name and password required to boot and connect to the server you configured. Click Next.
12. For Windows systems, depending on the template that you chose, the Configuration Wizard prompts you to install the new server as a Windows service. Select Yes if you want to run the server as a Windows service, or No to start the server from the command-line or windows Start menu. Click Next to continue.
13. For Windows systems, depending on the template that you chose, the Configuration Wizard prompts you to install the domain in the Windows Start menu. Select Yes if you want to install the server start script under the Start menu. Click Next to move to the Configuration Summary screen.
14. Verify the new domain and server configuration in the Configuration Summary screen. If you want to edit your selections, click the back button to return to the correct screen. Otherwise, click Create to create the startup scripts.

To start the Managed Server instance:

1. Start the domain's Administration Server on the WebLogic Server host that you specified in the **Admin Server Name or IP** field of the wizard (step 7. in "Configure Support for Remote Managed Servers").

2. Log in to the remote host and invoke the following script:

   ```
   domain-name/startManagedWebLogic.cmd (Windows)
   domain-name/startManagedWebLogic.sh (UNIX)
   ```

   where `domain-name` is the directory you specified in step 6. in “Configure Support for Remote Managed Servers”.

CHAPTER

3 Configuring Network Resources

The following sections describe how to configure WebLogic Server network resources in a domain:

Overview of Network Configuration

In BEA WebLogic Server 7.0 and later, you can use multiple Network Interface Cards (NICs) and/or multiple port numbers in your domain to improve performance and solve common networking problems. These capabilities allow you to:

- Separate administration traffic from application traffic in a domain.
- Improve network throughput by using multiple NICs with a single WebLogic Server instance.
- Designate specific NICs or multiple port numbers on a single NIC for use with specific WebLogic Server instances.
- Physically separate external, client-based traffic from internal, server-based traffic in a domain.
- Prioritize network connections that servers use to connect to other servers in a domain.

The instructions in this chapter describe how to configure domain network settings using the Administration Console. You can also configure these settings programmatically by using the specific MBeans described in each section.

New Network Configuration Features in WebLogic Server 7.0

Prior to WebLogic Server 7.0, a WebLogic Server instance could accept connections only from a single NIC, and the TCP port numbers that a given server instance used were restricted. The following table compares earlier TCP port restrictions with the network configuration features available in WebLogic Server 7.0.

Table 3-1 Network Feature Comparison

Version 6.x Restriction	Version 7.0 Capability
Each WebLogic Server instance listened on a single IP address.	A WebLogic Server instance can listen on multiple IP addresses.
A server can use a maximum of three distinct port numbers: ■ A standard port reserved for non-secure HTTP, IIOP, and T3 traffic (7001 by default). ■ A secure port reserved for HTTPS, IIOPS, and T3S traffic (7002 by default). ■ An optional port used to isolate administration traffic.	You can assign multiple port numbers to a server by creating and assigning multiple network channels. For more information, see "Network Channels" on page 3-10.
Administration Console traffic could take place on any port available in the default network configuration.	After you configure and enable a separate, SSL port for administration traffic, all WebLogic Server instances in the domain must use the administration port for all Administration Console network traffic.
Different quality of service levels could not be mixed among the available port numbers. For example, you could not support HTTPS traffic on one port and IIOPS traffic on another, because all secure traffic was restricted to the same port (7002 by default).	Protocol support can be tailored to individual TCP ports using network channels. For more information, see "Network Channels" on page 3-10.
Many network configuration fields, such as login timeout and backlog configuration applied to the server itself (the server's listen address), rather than the port. Those configuration settings could not vary by port.	Each of the configured network channels can use different protocol configurations for their TCP ports. In addition, servers that use the channels can override many protocol settings using Network Access Points. For more information, see "Network Channels" on page 3-10 and "Network Access Points (NAPs)" on page 3-12.
In a cluster, the multicast port number was copied from each server's listen port setting. Because all members of a cluster must use the same multicast address and port number, the copied port number required all servers in a cluster to use the same listen port.	A cluster's multicast configuration is no longer tied to individual servers' network configuration. Instead, you configure the cluster multicast port number independently of the port numbers used by cluster members. You can also identify which NIC each clustered server should use for multicast communication.

Understanding the Default Network Configuration

WebLogic Server 7.0's network configuration capabilities give you increased control over the characteristics of network connections used for client and server-to-server network traffic. Based on your requirements, you can configure network channels and Network Access Points to manage performance.

You must configure your domain to take advantage of these capabilities. If you do not configure channels and NAPs, your domain's network configuration will be similar to the that supported by previous versions of WebLogic Server, as summarized in the "Version 6.x Restriction" column of Table 3-1, "Network Feature Comparison," on page 3-3. This simple configuration, which does not utilize the new network configuration capabilities of WebLogic Server 7.0, is referred to as the default network configuration. For each server instance, the default network configuration enables a single listen address, one port for HTTP communication (7001 by default), and one port for HTTPS communication (7002 by default). You can configure the listen address and port assignments using the Configuration->General tab in the Administration Console; the values you assign are stored in attributes of the `ServerMBean` and `SSLMBean`, as in previous WebLogic Server versions.

The default configuration may meet your needs if:

- You are installing in a test environment that has simple network requirements.
- Your server uses a single NIC, and the default port numbers provide enough flexibility for segmenting network traffic in your domain.

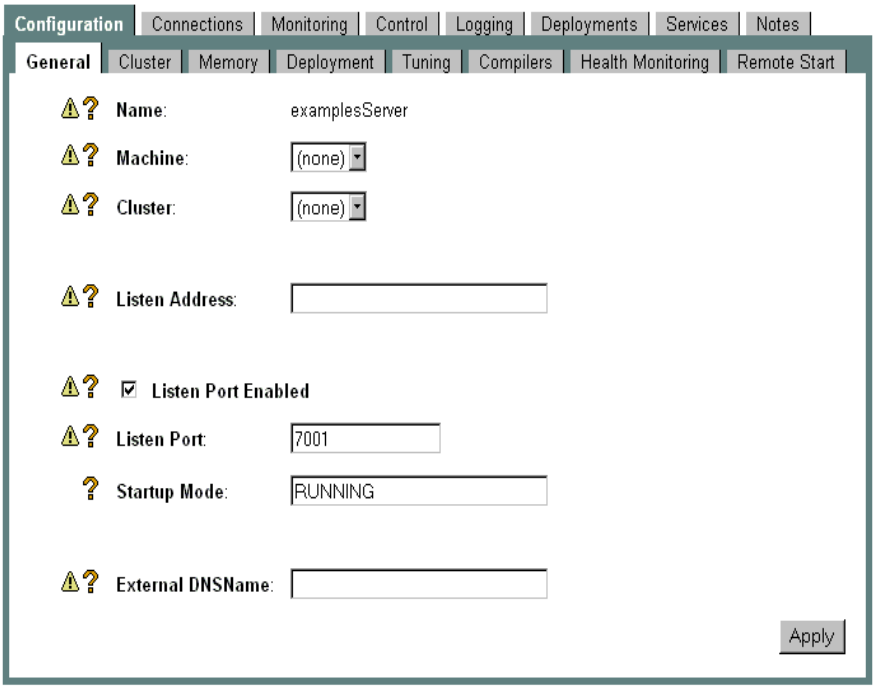
Using the default configuration ensures that third-party administration tools remain compatible with the new installation, because network configuration attributes remain stored in `ServerMBean` and `SSLMBean`.

Regardless of the configuration you perform, the settings associated with the default network configuration remain stored in `ServerMBean` and `SSLMBean`, and are used if if necessary to provide connections to a server instance.

Tailoring the Default Network Configuration

A server instance's default listen address, listen port, and network settings are defined in the its `ServerMBean`. You can change the default listen address, listen port, and default listen port connection properties using the following instructions:

1. Start the Administration Server for the domain if it is not already running.
2. Login to the Administration Console for the domain.
3. Select the Servers node in the left pane to display the server instances configured in the domain.
4. Select the name of the server instance you want to configure in the left pane.
5. Click the Configuration->General tab in the right pane to display the server instance's current default network settings:



Name: examplesServer

Machine: (none)

Cluster: (none)

Listen Address:

☑ Listen Port Enabled

Listen Port: 7001

Startup Mode: RUNNING

External DNSName:

6. Enter values for the default listen address and port as follows:

- Listen Address—Enter the default IP address or DNS name the server instance uses to listen for incoming connections.

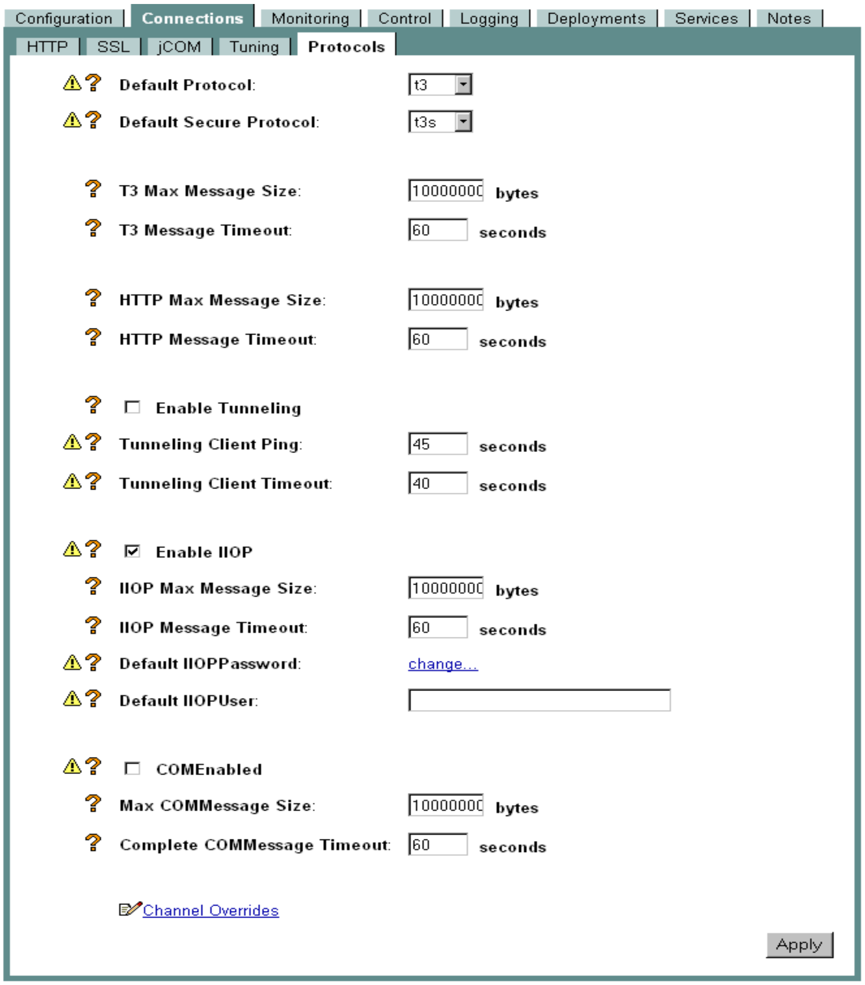
Notes: If you identify the server instance's Listen Address as localhost, non-local processes will not be able to connect to the server instance. Only processes on the machine that hosts the server instance will be able to connect to the server instance. If the server instance must be accessible as localhost (for instance, if you have administrative scripts that connect to localhost), and must also be accessible by remote processes leave the Listen Address blank. The server instance will determine the address of the machine and listen on it.

If the server instance resides on a multi-homed machine, do not leave the Listen Address field blank or specify the localhost address. If the server uses a localhost address, it will bind the Listen Port and SSL Listen Port to all available IP addresses on the multi-homed machine.

- Listen Port—Enter the default TCP port the server uses to listen for incoming connections.

7. Click Apply to apply your changes.

8. Click the Connections->Protocols tab in the right pane to define the default connection properties for the server:



Default Protocol:	t3	
Default Secure Protocol:	t3s	
T3 Max Message Size:	10000000	bytes
T3 Message Timeout:	60	seconds
HTTP Max Message Size:	10000000	bytes
HTTP Message Timeout:	60	seconds
☐ Enable Tunneling		
Tunneling Client Ping:	45	seconds
Tunneling Client Timeout:	40	seconds
☑ Enable IIOP		
IIOP Max Message Size:	10000000	bytes
IIOP Message Timeout:	60	seconds
Default IIOPPassword:	change...	
Default IIOPUser:		
☐ COMEnabled		
Max COMMessage Size:	10000000	bytes
Complete COMMessage Timeout:	60	seconds

9. Edit the default connection attributes on this page and click Apply to apply your changes. For information about individual attributes, see "Server --> Connections --> Protocols" in *Administration Console Online Help*.

10. Restart the server to use the new default network configuration.

Viewing the Default Configuration at Server Startup

As described in "Understanding the Default Network Configuration" on page 3-4, unless you configure the network configuration features new in WebLogic Server 7.0, your server instances will have a simple network configuration that adheres to the restrictions that applied in previous versions of WebLogic Server. As in previous versions of WebLogic Server, this basic network configuration information is stored using `ServerMBean` and `SSLMBean`.

One of the new network configuration capabilities of WebLogic Server 7.0 is the ability to configure network channels. A network channel allows you to assign multiple port numbers to a server instance. Network channels and their capabilities are described in "Network Channels" on page 3-10

At startup, WebLogic Server automatically generates a "default" network channel using the listen address and port attributes defined in the `ServerMBean`.

A server's default network configuration is shown in its log file following the line:

```
Network Channel:         Default
```

as shown in this log file excerpt:

```
####<Apr 22, 2002 10:49:36 AM PDT> <Info> <RJVM> <myhostname>
<examplesServer> <main> <kernel identity> <> <000520> <Network
Configuration
Cluster Participant:              false
Native Socket IO Enabled:         true
Reverse DNS Allowed:              false
Network Channel:         Default
Listen Address:          not configured
Listen Port:             7001
SSL Listen Port:         7002
External DNS Name:       not configured
Cluster Address:         not configured
Protocol(s):             T3,T3S,HTTP,HTTPS,IIOP,IIOPS,COM
Tunneling Enabled:       false
Outgoing Enabled:        true
Admin Traffic Only:      false
Admin Traffic OK:        true
Channel Weight:          50
Accept Backlog:          50
Login Timeout:           5000 ms
Login Timeout SSL:       25000 ms
```

```
Message Timeout HTTP:    60000 ms
Message Timeout T3:      60000 ms
Message Timeout COM:     60000 ms
Message Timeout IIOP:    60000 ms
Idle Timeout IIOP:       60000 ms
Max Message Size HTTP:   10000000
Max Message Size T3:     10000000
Max Message Size COM:    10000000
Max Message Size IIOP:   10000000
>
```

Later in the log file, you can see the actual listen addresses and ports to which the server binds:

```
####<Apr 22, 2002 10:58:52 AM PDT> <Notice> <WebLogicServer>
<myhost> <examplesServer> <SSLListenThread.Default> <kernel
identity> <> <000354> <Thread "SSLListenThread.Default" listening
on port 7002>

####<Apr 22, 2002 10:58:52 AM PDT> <Info> <WebLogicServer> <myhost>
<examplesServer> <ListenThread.Default> <kernel identity> <>
<000213> <Adding address: myhost/192.168.1.11 to licensed client
list>
```

Using Network Channels and NAPs

WebLogic Server 7.0 introduces two new configurable network resources that you can administer using either the Administration Console or WebLogic Server MBeans: network channels and Network Access Points (NAPs). These resources are stored in two new MBeans: `NetworkChannelMBean` and `NetworkAccessPointMBean`.

The listen address and port number attributes from `ServerMBean` are also used in certain circumstances. For example,

- if you do not define a Listen Address using a NAP, the Listen Address defined in `ServerMBean` is used.
- if you configure a domain-wide administration port, and for some reason a Managed Server cannot bind to its configured administration port, the default listen port from `ServerMBean` or `SSLMBean` is used.

The following sections describe the features provided by network channels and Network Access Points.

Network Channels

Network Channels enable you to configure additional port numbers and protocol settings for use with one or more WebLogic Server instances.

These port numbers are in addition to the default port numbers associated with `ServerMBean` and `SSLMBean` (described in "Understanding the Default Network Configuration" on page 3-4).

A network channel defines the basic attributes of a network connection to WebLogic Server including:

- The protocols the connection supports (HTTP, HTTPS, T3, T3S, COM).
- Default listen ports to use for secure and non-secure communication.
- Default properties for the connection such as the login timeout value and maximum message sizes.
- Whether or not the connection supports tunneling.
- Whether the connection can be used to communicate with other WebLogic Server instances in the domain, or used only for communication with clients.

You configure network channels as distinct entities in the Administration Console, and then assign one or more channels to servers in a domain. The server instances to which you assign a channel use the port numbers and protocol configuration associated with the channel, instead of the default network configuration.

Note: Messages sent via the T3 can contain DNS information about the hosts they originate on or are destined to. If a T3 connection is established across a firewall that has network address translation (NAT) enabled, it is possible that some information about the network configuration behind the firewall will be revealed. Using the firewall to prevent T3 connections through the firewall will prevent this problem.

Configuring Outgoing Connections

In addition to defining the connection attributes, channels allow you to separate incoming client traffic from internal, server-to-server traffic in the domain.

Each channel definition specifies whether or not the channel supports outgoing connections (supported by default). By assigning two channels to a server instance—one with support for outgoing connections and one without—you can independently configure network traffic for client connections and server connections. By combining the channels with multiple NAPs (see "Network Access Points (NAPs)" on page 3-12), you can also physically separate client and server network traffic onto different IP addresses or port numbers.

Channels also enable you to prioritize the connections that are used for outbound network traffic. If a server instance has several outbound-capable channels assigned, you can prioritize each channel with a weighted value. When the server instance initiates an outgoing connection, the NAPs assigned to channels with a higher-weighted value are used before those with lower-weighted channels. You can use this functionality to ensure that all server-to-server traffic has a guaranteed level of throughput, by assigning the highest weighting to an outbound channel/NAP combination that utilizes a fast NIC.

Note: Network channel weights apply only to internal connections made for remote references, such as a remote EJB reference or a resource located via JNDI. Channel weights are not used for connections initiated directly via a URL.

Common WebLogic Server Channels

Channels can be used to accomplish a variety of network configuration goals. Most WebLogic Server installations use one or more of the following common types of channels:

- "Default" Channel—WebLogic Server automatically generates a default channel to describe the listen address and listen port settings associated with the `ServerMBean`. You can view the default channel configuration during server startup, as described in "Viewing the Default Configuration at Server Startup" on page 3-8.
- Administration Channel—You can define an optional administration port to use for separating administration traffic from application traffic in your domain. When you enable the administration port, WebLogic Server automatically generates an Administration Channel based on the port settings. See

"Configuring a Domain-Wide Administration Port" on page 3-13 for more information.

- Custom Channels—A custom channel is a channel that you define and apply in a domain (rather than a channel that WebLogic Server automatically generates). See "Common Uses for Network Channels and NAPs" on page 3-13 for examples of how to use custom channels in your domain.

Network Access Points (NAPs)

A Network Access Point (NAP) is a resource you can configure to assign the port numbers, protocol configuration, and optionally, IP address to be used with a network channel on a server. A NAP is only used in conjunction with a network channel, and only one NAP can be assigned to each channel on a server instance.

You can use NAPs to override certain network channel attributes on a specific WebLogic Server instance, or to associate a server's network channel with a specific IP address or NIC.

A NAP is optional. If you configure a server instance to use a Network Channel but do not configure a NAP, the server uses the network configuration associated with the channel. Because the channel itself does not specify a listen address, the new connection uses the listen address associated with the server's default network configuration (the listen address defined in `ServerMBean`).

If you do specify a NAP with a network channel on a server, the channel uses the listen address identified in the NAP (if one is defined) to generate a new network connection. The NAP may also override certain protocol and network configuration settings of the underlying network channel to tailor the connection on this server. "Network Channel and NAP Attributes" on page 3-31 lists the attributes of Network Channels and NAPs, and explains which channel attributes a NAP can override.

Warning: If you use a network channel with a server that resides on a multi-homed machine, you must enter a valid listen address either in `ServerMBean` or in a NAP associated with the server. If the NAP and `ServerMBean` listen address are blank or specify the localhost address (IP address 0.0.0.0 or 127.*.*.*), the server will bind the network channel listen port and SSL listen ports to all available IP addresses on the multi-homed machine. See "Understanding the Default Network Configuration" on page 3-4 for information on setting the listen address in `ServerMBean`.

Common Uses for Network Channels and NAPs

WebLogic Server uses an automatically-generated channel when you configure an Administration Port in a domain. See "Configuring a Domain-Wide Administration Port" on page 3-13 for more information.

Using a single custom channel with multiple servers simplifies network configuration for a domain—changing a channel configuration automatically changes the connection attributes of all servers that use the channel. See "Using a Custom Channel to Simplify Domain Administration" on page 3-18 for instructions on configuring and applying a channel.

You can also create and assign multiple channels to a single server. Using multiple channels helps you segment network traffic by protocol, listen ports, or any other channel configuration property. For example, you can use two channels with a single server to tailor the default connection properties for secure vs. non-secure traffic. You can also use multiple channels to separate external, client traffic from internal, server-to-server traffic. See "Segmenting Network Traffic by Port Number" on page 3-25 for a simple example of using multiple channels. See "Separating Internal and External Network Traffic" on page 3-27 for an example of using channels to configure outgoing connections.

You can also override many channel properties on a per-server basis by configuring an associated Network Access Point. See "Using NAPs to Configure Multiple NICs with a Server" on page 3-20 for an example of using a NAP.

Configuring a Domain-Wide Administration Port

In WebLogic Server 7.0, you can enable an administration port for Managed Servers in a domain. An administration port is a port that a Managed Server uses only for communications with the domain's Administration Server.The administration port is optional, but it provides two important capabilities:

- It enables you to start a server in standby state. While in the standby state, the administration port remains available to activate or administer the Managed

Server, although the server's other network connections are unavailable to accept client connections. See "Starting and Stopping WebLogic Server" in the *Administration Guide* for more information on the standby state.

- It enables you to separate administration traffic from application traffic in your domain. In production environments, separating the two forms of traffic ensures that critical administration operations (starting and stopping servers, changing a server's configuration, and deploying applications) do not compete with high-volume application traffic on the same network connection.

WebLogic Server implements the administration port by generating an administration channel when the Managed Server starts up.

Administration Port Restrictions

The administration port accepts only secure, SSL traffic, and that all connections via the port require authentication. Because of these features, enabling the administration port imposes the following restrictions on your domain:

- The Administration Server and all Managed Servers in your domain must be configured with support for the SSL protocol. Managed Servers that do not support SSL will be unable to connect with the Administration Server during startup—you will have to disable the administration port in order to configure those servers.
- Because all server instances in the domain must enable or disable the administration port at the same time, you configure the administration port at the domain level. You can change an individual Managed Server's administration port number, but you cannot enable or disable the administration port for an individual Managed Server.
- After you enable the administration port, you must establish an SSL connection to the Administration Server in order to start any Managed Server in the domain. This applies whether you start Managed Servers manually, at the command line, or using Node Manager. To establish the SSL connection, use the following startup arguments:

  ```
  -Dweblogic.management.server=https://host:admin_port
  ```

  Note: You cannot simply supply a hostname and port combination as with previous server releases; the HTTPS protocol is required.

```
-Dweblogic.security.SSL.trustedCAKeyStore=path_to_keystore
-Dweblogic.security.SSL.ignoreHostnameVerification=true
```

- After enabling the administration port, all Administration Console traffic *must* connect via the administration port. (Prior to WebLogic Server 7.0, Administration Console traffic could connect using any available server port.)
- If multiple server instances run on the same computer in a domain that uses a domain-wide administration port, you must either:
  - Host the server instances on a multi-homed machine and assign each server instance a unique listen address, or
  - Override the domain-wide port on all but one of one of the servers instances on the machine. Override the port using the Local Administration Port Override option on the Advanced Attributes portion of the Server->Connections->SSL Ports page in the Administration Console.

Administration Port Configuration and Startup

Before you enable the administration port,

- Enable the SSL protocol on each server instance in your domain, including the Administration Server and all Managed Servers, using the Connections->SSL Ports tab in the Administration Console. Follow the steps in "Configuring the SSL Protocol" in *Managing WebLogic Security* to obtain SSL security components and configure Administration Server to use SSL, or use the demo certificate files installed by default with new servers.
- Make sure that each server in the domain has a configured default listen port or default SSL listen port. (See "Tailoring the Default Network Configuration" on page 3-5 for more information.) A default port is required in the event that the server cannot bind to its configured administration port. If an additional default port is available, the server will continue to boot and you can change the administration port to an acceptable value.

After performing these prerequisites, follow these steps to enable the administration port:

1. Start the Administration Server for the domain if you have not already done so.
2. Login to the Administration Console.

3. Click the name of your domain in the left pane to display the domain's configuration properties.

4. Click the Configuration->General tab in the right pane.

5. Select the `Enable Domain Wide Administration Port (Please configure SSL)` checkbox in the right pane.

6. Enter a value in the `Domain Wide Administration Port` attribute field to specify the default administration port all servers will use. By default, the domain-wide administration port is set to 9002.

7. Click Apply to apply your changes to the domain.

8. If you want to use the same administration port with all servers in the domain, skip to step 13 now.

9. To change the administration port that an individual server in the domain uses, select the Servers node in the left pane, then select the name of the server you want to configure.

10. Click the Connections->SSL Ports tab in the right pane to display the server's current SSL configuration.

11. Enter a value in the `Local Administration Port Override: (0: no override)` attribute field to specify the administration port this server uses to communicate with the administration console. A value of zero indicates that the server uses the domain wide administration port value you specified in step 6.

    Note: Choose a well-known, unused port number for the administration port on the Administration Server. All Managed Servers in the domain need to specify this port in order to start up in the domain.

12. Click Apply to apply your changes.

13. Reboot the Administration Server and all Managed Servers to use the new administration port.

    When rebooting the Managed Servers in the domain, you must specify the following options at the command line (or in the server start script) to connect to the Administration Server's administration port:

    ```
    -Dweblogic.management.server=https://host:admin_port
    -Dweblogic.security.SSL.trustedCAKeystore=path_to_keystore
    -Dweblogic.security.SSL.ignoreHostnameVerification=true
    ```

Viewing the Administration Channel at Server Startup

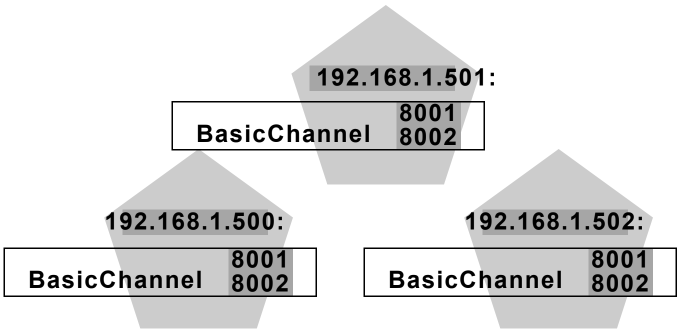
If you enable the Administration Port in a server's default network configuration, the server uses the listen address setting from the `ServerMBean` and SSL configuration settings from the `SSLMBean` to generate an "administration channel" for use with the server. The administration channel settings are similar to those of the default channel, except that the non-secure listen port settings are absent. The following log file excerpt shows default administration port setup:

```
Network Channel:         Administrator
Listen Address:          172.17.10.55
Listen Port:             none
SSL Listen Port:         9002
External DNS Name:       not configured
Cluster Address:         not configured
Protocol(s):             T3S,HTTPS
Tunneling Enabled:       false
Outgoing Enabled:        true
Admin Traffic Only:      true
Admin Traffic OK:        true
Channel Weight:          50
Accept Backlog:          50
Login Timeout:           5000 ms
Login Timeout SSL:       25000 ms
Message Timeout HTTP:    60000 ms
Message Timeout T3:      60000 ms
Max Message Size HTTP:   10000000
Max Message Size T3:     10000000
>

...

####<Apr 22, 2002 3:14:34 PM PDT> <Notice> <WebLogicServer>
<myhost> <adminserver> <SSLListenThread.Administrator> <kernel
identity> <> <000355> <Thread "SSLListenThread.Administrator"
listening on port 9002, ip address 192.168.1.11>
```

Using a Custom Channel to Simplify Domain Administration

Network channels can simplify domain network administration by serving as a template for WebLogic Server connection properties. For example, the figure below shows a domain that utilizes a single custom channel to define the connection characteristics for all network traffic for all servers.

Figure 3-1 Using a Custom Network Channel



In this example, a network channel named BasicChannel is configured as follows:

- ListenPort is set to the default of 8001
- SSLListenPort is set to the default of 8002
- All network protocols are enabled for this channel.

The administrator has added BasicChannel to every server instance in the domain.

This sample domain does not use NAPs. Instead, the listen address for each server is obtained from the ListenAddress attribute in each server's `ServerMBean`. The channel specify default port numbers for secure and non-secure traffic, which open new ports in addition to those specified the servers' `ServerMBean` attributes.

During the testing phase for applications in this domain, the administrator can set default values for all protocols supported for this channel. As the domain is opened up to additional beta testers, the administrator can fine tune connection timeouts and maximum message sizes as needed by making a single change to BasicChannel.

Configuring a Custom Network Channel

To configure a network channel, refer to "Network Channel and NAP Attributes" on page 3-31 and follow these steps:

1. Start the Administration Console for the domain that contains the server you want to configure.
2. Select the Network Channels node in the left pane of the Administration Console.
3. Click Configure a new Network Channel... in the right pane.
4. Enter the attribute values for the new network channel and click Create to create the new channel definition.

   Note: WebLogic Server uses the internal channel names `.WLDefaultChannel` and `.WLDefaultAdminChannel` and reserves the `.WL` prefix for channel names. You cannot create a custom channel that begins with `.WL`.
5. Select the Configuration->Tuning tab to change the backlog and timeout attributes for the new channel. Click Apply to apply your changes to this tab.
6. Select the Configuration->Protocols tab to enable, disable, or configure protocol support for the new channel. Click Apply to apply your changes to this tab.
7. After you configure the new network channel properties, select the Targets->Servers or Targets->Clusters tab to select the servers or clusters in the domain that will use the new channel. Select a server or cluster in the Available column, and use the arrow button to place the server or cluster in the Chosen column.
8. Click Apply to assign the network channel to the chosen servers or clusters.
9. To use the new channel port designations, you must reboot any servers that you assigned as targets.

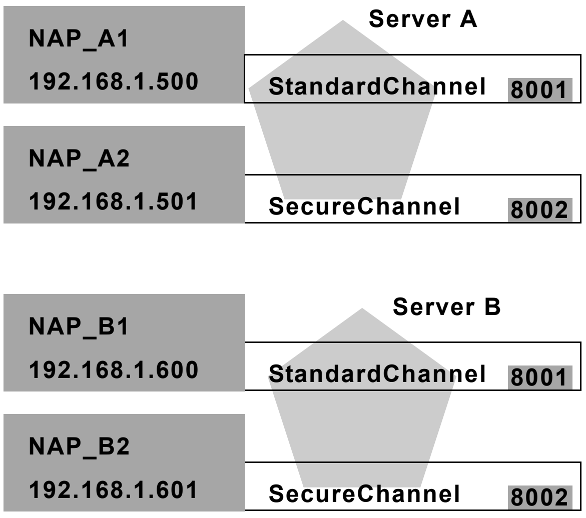
Using NAPs to Configure Multiple NICs with a Server

By default, a newly configured WebLogic Server instance uses only the single `ListenAddress`, `ListenPort`, and `SSLListenPort` attribute associated with its `ServerMBean` and `SSLMBean`. However, in domains where application performance is network-bound, rather than server-bound, it may be desirable to configure a single WebLogic Server instance with multiple NICs (or with the multiple IP addresses provided by multihomed hardware).

To utilize multiple NICs or multihomed hardware with a single server, you must override the default `ServerMBean` configuration by setting up multiple network channels and multiple associated NAPs. In such a configuration, NAPs segment incoming network traffic by identifying individual NICs to use with the server. The channels provide the baseline network configuration for each NAP.

For example, in the sample domain shown in the following figure, each server instance listen on two separate NICs. On each server instance, one NIC is reserved for standard network traffic, and the other is used for secure traffic

Figure 3-2 Using Network Access Points



To create this configuration, the administrator configured two channels—StandardChannel and SecureChannel—and assigned both channels to both server instances. To segment secure and non-secure traffic, the two channels were configured as follows:

- StandardChannel enables the HTTP and T3 protocols, but disables HTTPS and T3S. This channel uses the ListenPort and SSListenPort values of 8001 and 8002, although the SSL port is unused.
- SecureChannel enables the HTTPS and T3S protocols, but disables HTTP and T3. This channel uses the ListenPort and SSListenPort values of 8001 and 8002, although the standard ListenPort is unused.

Four NAPs were configured—one for each NIC in the domain. For WebLogic Server A, NAP_A1 was configured with listen address 192.168.1.500, and assigned to StandardChannel. NAP_A2 was created with listen address 192.168.1.501, and assigned to SecureChannel.

For WebLogic Server B, NAP_B1 and NAP_B2 were configured in a similar manner, but using IP addresses 192.168.1.600 and 192.168.1.601.

For both servers, the administrator accepted the default `ListenPort` and `SSLListenPort` values associated with the available channels. Note, however, that the port numbers could vary by NIC, if unique port numbers were defined in the associated NAPs.

Configuring a Network Access Point

You can configure a network access point for a server only after you have configured a network channel and assigned it to the server, using the instructions in "Configuring a Custom Network Channel" on page 3-19.

Follow these steps to configure a server's network access points. See "NAP Attributes" on page 3-34 for more information about individual attributes.

1. Start the Administration Console for the domain, if it is not already running.
2. In the left pane, select the Servers node, then select the server name you want to configure.
3. In the right pane, select the Configuration->Tuning tab to display the current backlog and tunneling settings for the server's default network configuration.
4. At the bottom of the right pane, click Configure Channel Fine Tunings... to display a table listing all network channels currently assigned to the server.
5. Click the name of the network channel for which you want to create a network access point. This displays a Configuration tab that allows you to override attributes of the selected channel.
6. Select the Configuration->General tab to override channel attributes such as the listen address or listen ports, then click Apply.
7. Select the Configuration->Tuning tab to override channel attributes for backlog and timeout settings, then click Apply.
8. Select the Configuration->Protocols tab to override channel configurations for supported network protocols, then click Apply.

   Note: You cannot enable or disable a network protocol from within the Network Access Point—only the network channel can enable or disable protocol support.

Configuring Network Channels with a Cluster

To use one or more custom channels with a WebLogic Server cluster, follow the guidelines described in the sections that follow.

If you do not intend to use custom channels with a WebLogic Server cluster, follow the instructions in "Setting up WebLogic Clusters" in *Using WebLogic Server Clusters* to set up your cluster.

Create Managed Servers

Use the Administration Console to create all Managed Servers that will participate in the cluster. Configure each Managed Server's listen address and listen ports as described in "Tailoring the Default Network Configuration" on page 3-5.

Managed Servers in a cluster require a default listen address and listen port to generate a default channel for the server. The custom channel you will add to the cluster is used in addition to the default channel, and its properties are applied to all members of the cluster.

Create the Cluster

Use the Administration Console to create a new cluster in the domain, as described in the Administration Console online help. When creating the cluster:

- Select an unused multicast address and enter that address in the Configuration->Multicast tab for the new cluster.

  For information about how Managed Servers in a cluster use multicast communications, and guidelines for setting the multicast address, see "One-to-Many Communication Using IP Multicast" in *Using WebLogic Server Clusters*.

- You do not need to enter a cluster address in the Configuration->General tab for the new cluster, if you will later apply a network channel to the cluster. The network channel has a Cluster Address attribute for specifying the address.
- Use the Servers tab to add Managed Servers to the new cluster. In a production environment, the Administration Server should not participate in a cluster.

Create and Assign the Network Channel

Use the instructions in "Configuring a Custom Network Channel" on page 3-19 to create a new network channel for use with the cluster. When creating the new channel:

- Make sure that the new channel's listen port and SSL listen port do not use the same values as the clustered servers' default ports. If the network channel specifies the same port as a Managed Server's default port, the custom network channel and the Managed Server's default channel will each try to bind to the same port, and you will be unable to start the Managed Server.
- Enter a Cluster Address value on the new channel's Configuration->General tab. The cluster address should be a publicly-available address that identifies all managed servers in the cluster (for example, a DNS name that resolves to all IP addresses participating in the cluster). The network channel uses this value to generate EJB handles and failover addresses for use with the cluster.

  If you do not specify a cluster address in the network channel, the cluster will use the Cluster Address value specified in the cluster definition, if available.
- After creating the new channel, use the channel's Targets->Cluster tab to choose the cluster you created in "Create the Cluster" on page 3-23.

Define Multicast Address for each Server Instance

You can optionally designate which NIC each server in the cluster uses for multicast traffic. If you do not specify a multicast address for a server instance, it will use the multicast address you defined for the cluster in "Create the Cluster" on page 3-23

Follow these steps:

1. Start the Administration Console for the domain, if it is not already running.

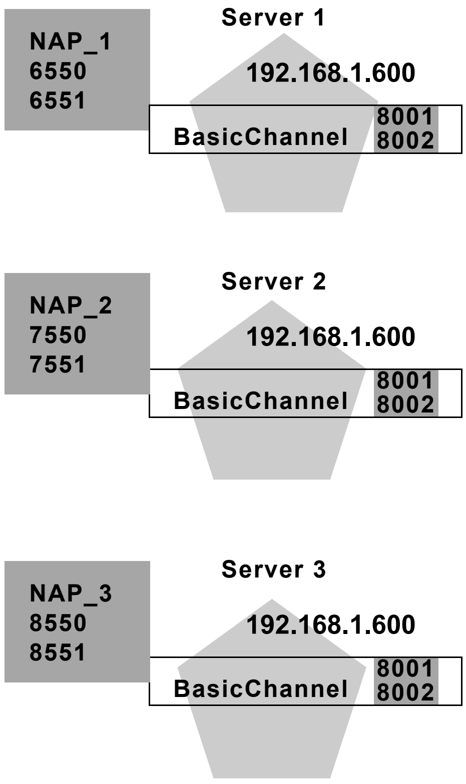
2. In the left pane, select the Servers node, then select the server name you want to configure.
3. In the right pane, select the Configuration->Cluster tab.
4. In the Interface Address attribute field, enter the IP address of the NIC the server should use for multicast traffic.
5. Click Apply to apply your changes.

Segmenting Network Traffic by Port Number

You can use NAPs and channels to segment network traffic by port numbers. For example, if a WebLogic Server in your domain is not network bound, you may choose to use a single NIC with the server but configure multiple port numbers to direct connections to multiple physical servers in the domain. Using NAPs allows you to configure more port numbers than the standard three port numbers available in the `ServerMBean`.

In the sample domain below, the administrator has configured multiple NAPs to utilize a single NIC with three different server instances:

Figure 3-3 Segmenting Network Traffic



In this example, a single NIC is used to segment network connections to multiple servers. BasicChannel is used in conjunction with multiple NAPs to handle connections over unique port numbers.

All servers in the sample domain use the default NIC IP address, 192.168.1.600. However, each server uses a NAP to listen on different port numbers of the same IP address. Notice that while the BasicChannel configuration uses the port numbers 8001 and 8002, the NAP on each server overrides those port assignments.

You can use similar network configurations, in conjunction with load balancers, to distribute connection requests to multiple WebLogic Server instances in a cluster.

Separating Internal and External Network Traffic

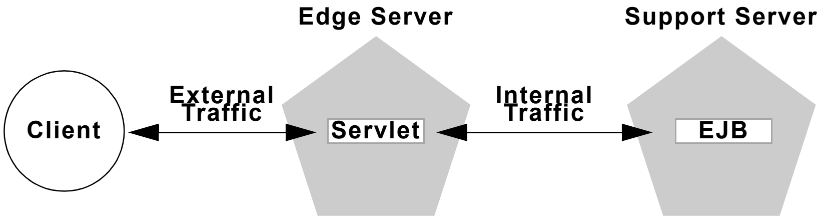
A more specialized use of network channels and NAPs involves separating client-oriented, external network traffic from server-oriented, internal traffic. You may want to separate internal and external traffic in certain firewall configurations, or to guarantee different levels of throughput for servers and clients.

Configuring Edge Servers

An "edge" server is a WebLogic Server instance that external clients use to initiate contact with a Web application. Although components of the Web application may reside on other WebLogic Servers in the domain, direct client access is restricted to the edge server.

For example, you may deploy a servlet or JSP on a single WebLogic Server in your domain, and make that server's IP address available to external clients. The servlet or JSP may interact with a second server in the domain to obtain EJBs or other services. However, the WebLogic Server that hosts the EJBs is not made available to external clients. Instead, clients interact with EJBs only via the edge server itself. The following figure depicts a simple edge server configuration.

Figure 3-4 Edge Servers

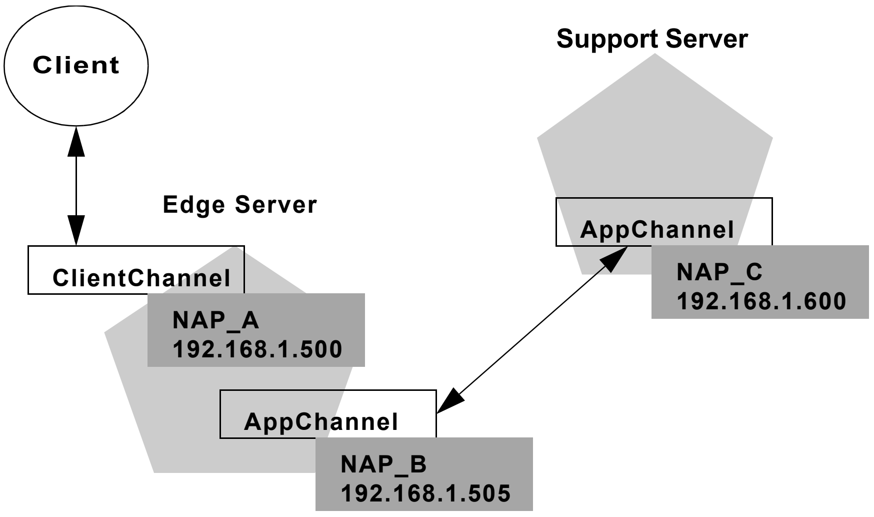


Note: A WebLogic Server domain can contain any number of edge servers. Edge servers can be stand-alone Managed Servers, or be configured as cluster. See "Recommended Multi-Tier Architecture" in *Using WebLogic Server Clusters* for information about configuring a cluster of edge servers.

Network channels help you separate external, client-based network connections from internal, server-based connections with edge server configurations. The separation of network traffic can be either logical (separated by port numbers of a single NIC), or physical (separated by different NICs and IP addresses).

The following figure shows a simple edge server configuration that uses one NIC for internal traffic and another for external network traffic.

Figure 3-5 Simple Edge Server Configuration



In this example, the two servers uses the same channel, AppChannel, to communicate with each other. The `OutgoingEnabled` attribute is enabled in AppChannel to provide two-way communication between the servers.

The EdgeServer in this domain also uses a separate channel, ClientChannel, to handle network connections from Web application clients. The OutgoingEnabled attribute is disabled in ClientChannel; this forces EdgeServer to use the AppChannel as well as its associated NIC to initiate connections to SupportServer.

Tracing the network traffic through this configuration:

1. A client connects to EdgeServer over the public IP address, 192.168.1.500, defined in the associated NAP on EdgeServer. This NAP uses the connection properties in ClientChannel.
2. The client accesses the servlet on EdgeServer, which in turn looks up an EJB that resides in SupportServer.
3. To initiate the network connection and access the EJB, EdgeServer requires a channel that support outgoing connections. In this configuration only AppChannel supports outgoing connections, so it is used for communication with SupportServer.
4. Because AppChannel has an associated NAP on EdgeServer, the NAP's IP address of 192.168.1.505 is used for communication with SupportServer. This physically separates the internal network connection from the connection the client used to access the Web application.

Network configuration at the WebLogic Server domain level can be combined with firewall hardware or software to block access to supporting servers.

Prioritizing Outgoing Connections

If a WebLogic Server has several channels capable of initiating outgoing connections, the server must choose which channel to use when connecting to another server. WebLogic Server first selects channels based on the protocol required for the connection. If multiple channels have the same protocol support, you can prioritize those channels by assigning a different weight to each.

A channel weight is a simple numerical value that can be applied to the `NetworkChannelMBean`. Channel weights are considered only when multiple channels with the same service level could be used to initiate an outgoing connection. (If a channel with a higher service level is currently active, it is used regardless of channel weights). Higher-valued weights are selected over lower-weighted channels to choose a NAP for outgoing connections.

In a multihomed system, channel weights allow you to prioritize equivalent channels based on the known capacity of available network cards.

Note: The default channel and administration channel, derived from values in the `ServerMBean` and `SSLMBean`, are always considered for outgoing connections, and use a default weight of 50.

Handling Channel Failures

Although WebLogic Server always attempts to use the highest-weighted channels before lower-weighted ones, a network failure may render the selected channel unavailable. To handle potential failures, WebLogic Server selects outgoing channels using the following algorithm:

1. WebLogic Server first tries the highest-weighted channel having the required quality of service.
2. If a connection cannot be made using the highest-weighted channel, WebLogic Server tries the next-highest weighted channel with the required quality of service.
3. If the connection request fails again, the server continues the connection attempt using lower-weighted channels, until all channels have been attempted.
4. If the server cannot connect using any available channel, a failure message is returned to the calling user.

The above procedure ensures that users receive a connection error message only when all channels of the required quality of service level have been exhausted. If all channel combinations are exhausted and another user attempts to initiate an outgoing connection (or a connection is retried after a failure), WebLogic Server restarts the above algorithm beginning with the highest-weighted channel.

Upgrading Quality of Service Levels for RMI

For RMI lookups only, WebLogic Server may upgrade the service level of an outgoing connection. For example, if a T3 connection is required to perform an RMI lookup, but an existing channel supports only T3S, the lookup is performed using the T3S channel.

This upgrade behavior does not apply to server requests that use URLs, since URLs embed the protocol itself. For example, the server cannot send a URL request beginning with `http://` over a channel that supports only `https://`.

Network Channel and NAP Attributes

The following sections list the attributes available in a NetworkChannelMBean and NetworkAccessPointMBean, and explain which NAP attributes can override their channel counterparts.

Channel Attributes

A channel is represented in a WebLogic Server domain as a `NetworkChannelMBean`. This MBean consists of the configuration attributes described in the following table. Values of highlighted attributes can be overridden by specifying a complementary value in the channel's associated `NetworkAccessPointMBean` using the Administration Console. See "NAP Attributes" on page 3-34 for more information about overriding channel properties.

Table 3-2 Network Channel Configuration Attributes

MBean Attribute Name	Default Value	Possible Values	Description
Name*	None	String	The name used to identify this channel in the console. WebLogic Server uses the internal channel names `.WLDefaultChannel` and `.WLDefaultAdminChannel` and reserves the `.WL` prefix for channel names. You cannot create a custom channel that begins with `.WL`.
Description	None	String	Optional text notes describing this channel.
ChannelWeight*	50	1-100	The relative weight to assign to this channel when selecting channels for outgoing connections. See "Prioritizing Outgoing Connections" on page 3-29 for more information.
ListenPortEnabled	False	True, False	Specifies whether this channel provides a plain text listen port.

MBean Attribute Name	Default Value	Possible Values	Description
ListenPort*	8001	1-65534	The default listen port to use for non-secure network protocols.
SSLListenPortEnabled	False	True, False	Specifies whether this channel provides an SSL listen port.
SSLListenPort*	8002	1-65534	The default listen port to use for secure network protocols.
COMEnabled	False	True, False	Specifies whether this channel provides a plain text (non-SSL) port for COM traffic.
ClusterAddress	None	String	The address that this channel uses to generate EJB handles and failover addresses for use in a cluster
TunnelingEnabled*	False	True, False	Specifies whether this network channel supports tunneling. To enable tunneling for a network channel, you must also enable the HTTP and T3 protocols for the channel by setting the attributes: ■ HTTP(S) Enabled=true ■ T3(S) Enabled=true Although T3 is required for enabling tunneling, the T3 protocol is hidden from clients that use a tunneling connection. For general information about tunneling, see Setting Up WebLogic Server for HTTP Tunneling in the *Administration Guide*.
T3 Enabled*	False	True, False	Enables or disables the protocol.
T3S Enabled*	False	True, False	Enables or disables the protocol.
HTTP Enabled*	False	True, False	Enables or disables the protocol.
HTTPS Enabled*	False	True, False	Enables or disables the protocol.
OutgoingEnabled*	True	True, False	Specifies whether this channel can initiate outbound connections to other WebLogic Servers in the domain.

MBean Attribute Name	Default Value	Possible Values	Description
LoginTimeoutMillis	5000	0 (Disabled), 1-100000	Sets the number of milliseconds that WebLogic Server should wait for a connection before timing out.
LoginTimeoutMillisSSL	25000	0 (Disabled), 1-2147483647	Sets the number of milliseconds that WebLogic Server should wait for an SSL connection before timing out.
AcceptBacklog*	50	0-2147483647	Sets the number of connections available for backlog. To increase the number of connections to be processed, increase this number.
TunnelingClientPingSecs*	45	0-2147483647	Sets the time, in seconds, that the server will wait before pinging the client.
TunnelingClientTimeoutSecs*	40	0 (Disabled) 1-2147483647	Sets the time, in seconds, that the server will wait before timing out.
MaxT3MessageSize	10000000	4096-2000000000	Sets the size, in bytes, of the maximum T3 message.
MaxHTTPMessageSize	10000000	4096-2000000000	Sets the size, in bytes, of the maximum HTTP message.
MaxCOMMessageSize	10000000	4096-2000000000	Sets the size, in bytes, of the maximum COM message.
CompleteT3MEssageTimeout	60	0 (Disabled), 1-480	The amount of time, in seconds, before the system times out while waiting to receive a T3 message.
CompleteHTTPMessageTimeout	60	0 (Disabled), 1-480	The amount of time, in seconds, before the system times out while waiting to receive an HTTP message.
CompleteCOMMessageTimeout	60	0 (Disabled), 1-480	The amount of time, in seconds, before the system times out while waiting to receive a COM message.

* Indicates the attribute is not dynamically configurable (requires server restart for changes to take effect).

NAP Attributes

A NAP is represented in a WebLogic Server domain as a `NetworkAccessPointMBean`. This MBean consists of the configuration attributes described in the following table. Note that many of the MBean attributes have counterparts in the `NetworkChannelMBean` described in "Network Channel and NAP Attributes" on page 3-31. When a NAP is assigned to a channel, it can potentially override the channel attribute values, as described below.

Table 3-3 Network Access Point (NAP) Configuration Attributes

MBean Attribute Name	Default Value	Possible Values	Description
ListenAddress*	Null	String host name or IP address	The IP address or DNS name this NAP uses to listen for incoming connections. If this attribute is null, the NAP uses the value of the `ListenAddress` attribute specified in the associated server's `ServerMBean`. (If no value is specified in `ServerMBean`, WebLogic Server listens to the localhost address.) **Note:** To resolve a DNS name to an IP address, Weblogic Server must be able to contact an appropriate DNS server or obtain the IP address mapping locally. Therefore, if you specify a DNS name for the listen address, you must either leave a port open long enough for the WebLogic Server instance to connect to a DNS server and cache its mapping or you must specify the IP address mapping in a local file. If you specify an IP address for ListenAddress and then a client request specifies a DNS name, WebLogic Server will attempt to resolve the DNS name, but if it cannot access DNS name mapping, the request will fail.

MBean Attribute Name	Default Value	Possible Values	Description
ListenPort*	-1	-1 1-65534	-1 forces the NAP to use the value defined in the associated `NetworkChannelMBean`. Setting any other value overrides the value defined in the associated `NetworkChannelMBean`. See the `ListenPort` channel attribute description for more information.
SSLListenPort*	-1	-1 1-65534	-1 forces the NAP to use the value defined in the associated `NetworkChannelMBean`. Setting any other value overrides the value defined in the associated `NetworkChannelMBean`. See the `SSLListenPort` channel attribute description for more information.
LoginTimeoutMillis	-1	-1 0 1-100000	-1 forces the NAP to use the attribute value defined in the NAP's `NetworkChannelMBean`. Setting any other value overrides the attribute value defined in the NAP's `NetworkChannelMBean`; 0 disables timeouts. See "Network Channel and NAP Attributes" on page 3-31 for more information about this attribute.
LoginTimeoutMillis SSL	-1	-1 0 1- 2147483647	-1 forces the NAP to use the attribute value defined in the NAP's `NetworkChannelMBean`. Setting any other value overrides the attribute value defined in the NAP's `NetworkChannelMBean`; 0 disables timeouts. See "Network Channel and NAP Attributes" on page 3-31 for more information about this attribute.
AcceptBacklog*	-1	-1 0 1-2147483647	-1 forces the NAP to use the attribute value defined in the NAP's `NetworkChannelMBean`. Setting any other value overrides the attribute value defined in the NAP's `NetworkChannelMBean`; 0 disables backlog. See "Network Channel and NAP Attributes" on page 3-31 for more information about this attribute.

MBean Attribute Name	Default Value	Possible Values	Description
TunnelingClient PingSecs*	-1	-1 0 1-2147483647	-1 forces the NAP to use the attribute value defined in the NAP's `NetworkChannelMBean`. Setting any other value overrides the attribute value defined in the NAP's `NetworkChannelMBean`; 0 disables client pings. See "Network Channel and NAP Attributes" on page 3-31 for more information about this attribute.
TunnelingClient TimeoutSecs*	-1	-1 0 1-2147483647	-1 forces the NAP to use the attribute value defined in the NAP's `NetworkChannelMBean`. Setting any other value overrides the attribute value defined in the NAP's `NetworkChannelMBean`; 0 disables the timeout. See "Network Channel and NAP Attributes" on page 3-31 for more information about this attribute.
CompleteT3 MessageTimeout	-1	-1 0 1-480	-1 forces the NAP to use the attribute value defined in the NAP's `NetworkChannelMBean`. Setting any other value overrides the attribute value defined in the NAP's `NetworkChannelMBean`. See "Network Channel and NAP Attributes" on page 3-31 for more information about this attribute.
CompleteHTTP MessageTimeout	-1	-1 0 1-480	-1 forces the NAP to use the attribute value defined in the NAP's `NetworkChannelMBean`. Setting any other value overrides the attribute value defined in the NAP's `NetworkChannelMBean`; 0 disables the timeout. See "Network Channel and NAP Attributes" on page 3-31 for more information about this attribute.
CompleteCOM MessageTimeout	-1	-1 0 1-480	-1 forces the NAP to use the attribute value defined in the NAP's `NetworkChannelMBean`. Setting any other value overrides the attribute value defined in the NAP's `NetworkChannelMBean`; 0 disables the timeout. See "Network Channel and NAP Attributes" on page 3-31 for more information about this attribute.

* Indicates the attribute is not dynamically configurable (requires server restart for changes to take effect).

CHAPTER

4 Recovering Failed Servers

You can use several strategies to recover failed WebLogic Server instances in a domain. All of the strategies require that you back up a domain's configuration and security data.

This topic describes the following tasks:

- Backing Up Configuration Data
- Backing Up Security Data
- Restarting an Administration Server When Managed Servers are Running
- Starting a Managed Server When the Administration Server Is Not Accessible
- Self-Health Monitoring and Restart for Managed Servers

Backing Up Configuration Data

Because the Administration Server uses its configuration files to administer the domain, we recommend that you keep an archived copy in case a failure of the Administration Server causes them to become unavailable. You can use any of the common methods of archiving, such as periodic back-ups, fault tolerant disks, and manually copying files whenever they are changed. If an Administration Server fails, you can copy your backup files onto a new machine and restart and Administration Server on the new machine.

By default, an Administration Server stores a domain's configuration data in a file called `domain_name\config.xml`, where *`domain_name`* is the root directory of the domain. Many of the configuration changes that you make to servers within the domain, either by using the Administration Console, the `weblogic.Admin` command, or the JMX API, are persisted in the `config.xml` file. For example, the MBeans that configure a WebLogic Server instance persist their data in `config.xml`.

Note: When you start a server, you can configure it to use a different configuration file. For more information, refer to Using the weblogic.Server Command in the *WebLogic Server Administration Guide*.

After an Administration Server successfully completes its startup sequence and is ready to process requests, it saves a copy of its configuration file in the following file:

```
domain_name/config.xml.booted
```

We recommend that you make a backup copy of `config.xml` and `config.xml.booted`. If you need to back out of any changes that you make to the configuration during a given session, you can revert to the configuration as defined in `config.xml.booted`.

In addition, you can configure one or more Managed Servers to replicate some of a domain's configuration data. A subsequent section in this topic, "Replicating the Domain's Configuration Files" on page 4-15, describes setting up this type of replication.

Backing Up Security Data

The Administration Server is responsible for maintaining security data for a domain. We recommend that you archive security data on the Administration Server in case it fails. You cannot modify security data unless an Administration Server is available.

This section describes the following tasks:

- Backing Up Security Configuration Data
- Backing Up the WebLogic LDAP Repository
- Backing Up SerializedSystemIni.dat and Security Certificates

Backing Up Security Configuration Data

For each security provider that you use, the WebLogic Security framework instantiates a managed bean (MBean) to maintain the provider's configuration. To persist the configuration across server sessions, a domain creates an MBean repository in a directory named `*domain_name*\userConfig\Security`. When you start a server, it uses the data from the repository to create a runtime cache. The Administration Server periodically updates the `userConfig\Security` directory based on data in the cache. When you shut down a sever, it also updates the MBean repository.

You can use any of the following strategies to back up your security MBean repository, which is in a binary format:

- Back up the `*domain_name*\userConfig` directory after you make any significant changes to your security data. For example, after you complete your initial configuration of the security providers, shut down the server and back up the `userConfig` directory.

  If a domain's Administration Server fails, you can copy the `userConfig` backup to the root directory of a new Administration Server.

- Dump the data into an XML format and back up the XML file. If the Administration Server fails, you can load the XML file into a new Administration Server's configuration directory.

  Because this strategy dumps data into XML, you can also use it to debug and repair errant configurations.

The following sections describe how to back up and debug security data using XML files:

- Dumping Security Configuration Data into an XML File
- Modifying the Dumped XML File
- Loading XML Data into an MBean Repository
- Creating a Default Security Data Repository

Dumping Security Configuration Data into an XML File

The `WebLogicMBeanDumper` utility reads the data in *`domain_name`*`\userConfig` and outputs an XML file. Because the utility works with data in the MBean repository, you do not need to start a server to use `WebLogicMBeanDumper`.

To dump data for all security-provider MBeans, do the following:

1. Shut down the Administration Server to prevent the server's periodic updates to the MBean repository.
2. Open a command shell.
3. Navigate to the domain's root directory.
4. To set the Java classpath, enter the following command:

   `WL_HOME``\server\bin\setWLSEnv.cmd` (on Windows)
   `WL_HOME``/server/bin/setWLSEnv.sh` (on UNIX)

   For more information, refer to Setting the Classpath in the *WebLogic Server Administration Guide*.
5. Enter the following command:

   ```
   java weblogic.management.commo.WebLogicMBeanDumper
       -includeDefaults -name Security:* output-file
   ```

Listing 4-1 shows an excerpt of an output file that contains security MBean data. The XML format is as follows:

- Each `<SetMBean>` element represents one security-provider MBean. Within this element:
  - `DisplayName` is the name that the Administration Console displays.
  - `ObjectName` is the programmatic name of the MBean instance. The naming convention for Security MBeans that you create from the Administration Console is as follows:
    `Security:Name=`*`securityRealmSecurityProvider`*
  - `TypeName` is the name of the MBean type. The MBean type specifies the available attributes and default values for an MBean instance. For more information, refer to MBean Definition File (MDF) Element Syntax in the *Developing Security Services for WebLogic Server* Guide.
- The `<Attributes>` subelement contains a name-value list of MBean attributes.

- The `<Defaulted AttributeNames>` subelement contains the list of MBean attributes for which the value is the default value specified by the MBean type.

  To see the default values, refer to the Administration Console, or do one of the following:

  - To see defaults of a type that you created, look in the MDF that you used to create the MBean type or look in the Administration Console under Security→Realms→*yourRealm*.
  - To see the default values of a type that WebLogic Server installs, refer to the WebLogic Server Javadoc for the type. For example, to see the default values for the `DefaultAuthenticator`, refer to the `weblogic.security.providers.authentication.DefaultAuthenticatorMBean` JavaDoc.

Listing 4-1 Security MBean Instances Formatted in XML

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

<MBeans>

  <SetMBean
      DisplayName="DefaultRoleMapper"
      ObjectName="Security:Name=myrealmDefaultRoleMapper"
      Type="weblogic.management.security.providers.authorization.DefaultRoleMap
      per"
  >

      <Attributes Realm="Security:Name=myrealm">

            <Defaulted AttributeNames="RoleDeploymentEnabled"/>

      </Attributes>

  </SetMBean>

...

</MBeans>
```

Modifying the Dumped XML File

You can make any of the following changes to the dumped XML file:

- Remove an MBean instance from the repository by deleting the corresponding `<SetMBean>` element. Make sure that you remove everything between and including the `<SetMBean>` and `</SetMBean>` tags.
- Restore an MBean to its default configuration by removing it from the repository as described above. Then, after you load the XML file and start the server, use the Administration Console to create a new instance.
- Create a new MBean instance by copying an existing one and modifying `DisplayName`, `ObjectName`, and other values. `Type` must refer to an existing MBean type. This documentation assumes that all `ObjectName`s follow the existing naming convention: `Security:Name=`*`unique-name`*.
- Replace a value for one of the name-value pairs in the `<Attributes>` element.

By default, the values of password attributes are not printed in clear text in the XML file but instead are masked as "*****". This avoids the security risk of printing passwords in clear text in the file system, but makes these files non-portable to different domains since the ***** mask strings cannot be intrepreted meaningfully. To make the XML file portable, you need to print clear text values to it, or else encrypted values that the new domain can decrypt. Do this using the `-Dweblogic.management.commo.dumpFormat=<encrypted | cleartext>` argument, which causes the following behavior:

- `encrypted` prints the attribute values in 3DES encrypted form. This should only be used when using the same Encryption salt in the new domain.
- `cleartext` prints the attribute values in clear text.
- Omitting this argument or using any other value obscures the attribute values, printing *****.

When you load the modified XML file as described in the next section, the `WebLogicMBeanLoader` utility regenerates the repository based on the data in the file.

Loading XML Data into an MBean Repository

To replace the data in a domain's MBean repository with an XML file generated by `WebLogicMBeanDumper`, do the following:

1. Shut down the Administration Server to prevent the server's periodic updates to the MBean repository.

2. Under *domain_name*/userConfig, rename the Security directory and move it outside of the userConfig directory tree.

   Note: Make sure that you **rename** and move the directory instead of just making a copy. These steps assume that you want to replace the entire security MBean repository with the data in your dumped XML file. If a *domain_name*/userConfig/Security directory exists before you go to the next step, modifications to existing MBeans will succeed, any MBean additions will succeed, but no MBeans will be removed.

3. To set the Java classpath, enter the following command:

   WL_HOME\server\bin\setWLSEnv.cmd (on Windows)
   WL_HOME/server/bin/setWLSEnv.sh (on UNIX)

   For more information, refer to Setting the Classpath in the *WebLogic Server Administration Guide*.

4. From the domain's root directory, enter the following command:

   ```
   java weblogic.management.commo.WebLogicMBeanLoader XML-file
   ```

Creating a Default Security Data Repository

If you want to regenerate all security MBeans in a domain and return to the configuration as installed by WebLogic Server, do the following:

1. Shut down the Administration Server to prevent the server's periodic updates to the MBean repository.
2. Under *domain_name*/userConfig, rename the Security directory and move it outside of the userConfig directory tree.
3. Start the Administration Server.

The Administration Server generates a new security MBean repository that uses the installed security providers with default values. To log on to the server with this installed configuration, you must provide the administrative user name that you specified when you created the domain. For more information, refer to Specifying an Initial Administrative User in the *WebLogic Server Administration Guide*.

Backing Up the WebLogic LDAP Repository

The default Authentication, Authorization, Role Mapper, and Credential Mapper providers that are installed with WebLogic Server store their data in an LDAP server. Each WebLogic Server contains an embedded LDAP server. The Administration Server contains the master LDAP server which is replicated on all Managed Servers. If any of your security realms use these installed providers, we recommend that you back up the following directory tree:

```
domain_name\adminServer\ldap
```

where `domain_name` is the domain's root directory and `adminServer` is the directory that the Administration Server generates to store runtime, security, and other data. Each WebLogic Server generates such a directory, but you only need to back up the LDAP data on the Administration Server.

For example, your security realm uses the default Authentication provider that is installed with WebLogic Server. If your Administration Server is named `myAdminServer` and your domain is named `myDomain`, back up the following directory tree:

```
myDomain\myAdminServer\ldap
```

Under the `ldap` directory, the `ldapfiles` subdirectory contains the data files for the LDAP server. Data files in this directory store user, group, group membership, policies, and role information. Other subdirectories of the `ldap` directory store such information as message logs for the LDAP server and data about replicated LDAP servers.

If you or someone else is modifying data for one of the security providers while you are backing up the `ldap` directory tree, it is possible that your backup of the files in the `ldapfiles` subdirectory will be in an inconsistent state. For example, if someone is using the installed default Authorization provider to add a user, the backup might start after the add has started, but before the add has completed.

Once a day, a server suspends write operations and creates its own backup of the LDAP data. It archives this backup in a `ZIP` file below the `ldap\backup` directory and then resumes write operations. This backup is guaranteed to be consistent, but it might not contain the latest security data.

For information about configuring this LDAP backup, refer to Configuring Backups for the Embedded LDAP Server in the Administration Console Online Help.

You do not need to back up the LDAP data on a Managed Server because the master LDAP server replicates the LDAP on each Managed Server as updates are made to the master server. If a domain's Administration Server is unavailable, the WebLogic security providers cannot modify security data. (The LDAP repositories on Managed Servers are replicas and therefore cannot be modified.)

Backing Up SerializedSystemIni.dat and Security Certificates

All servers create a file named `SerializedSystemIni.dat` and locate it in the server's root directory. This file contains encrypted security data that must be present to boot the server. You must back up this file.

If you configured a server to use SSL, you must also back up the security certificates and keys. The location of these files is user-configurable.

Restarting an Administration Server When Managed Servers are Running

If the Administration Server shuts down while Managed Servers continue to run, you do not need to restart the Managed Servers that are already running in order to recover management of the domain. The procedure for recovering management of an active domain depends upon whether you can restart the Administration Server on the same machine it was running on when the domain was started.

This section describes the following tasks:

- Restarting an Administration Server on the Same Machine
- Restarting an Administration Server on Another Machine

Restarting an Administration Server on the Same Machine

If you restart the WebLogic Administration Server while Managed Servers continue to run, by default the Administration Server can discover the presence of the running Managed Servers.

Note: Make sure that the startup command or startup script does not include `-Dweblogic.management.discover=false`, which disables an Administration Server from discovering its running Managed Servers. For more information about `-Dweblogic.management.discover`, refer to Frequently Used Optional Arguments in the *WebLogic Server Administration Guide*.

The root directory for the domain contains a file `running-managed-servers.xml` which is a list of the Managed Servers that the Administration Server knows about. When the Administration Server starts, it uses this list to check for the presence of running Managed Servers.

Restarting the Administration Server does not cause Managed Servers to update the configuration of static attributes. *Static attributes* are those that a server refers to only during its startup process. WebLogic Servers must be restarted to take account of changes to static configuration attributes. Discovery of the Managed Servers only enables the Administration Server to monitor the Managed Servers or make runtime changes in attributes that can be configured while a server is running (dynamic attributes).

Restarting an Administration Server on Another Machine

If a machine crash prevents you from restarting the Administration Server on the same machine, you can recover management of the running Managed Servers as follows:

1. Install the WebLogic Server software on the new administration machine (if this has not already been done).

2. Make your application files available to the new Administration Server by copying them from backups or by using a shared disk. Your application files should be available in the same relative location on the new file system as on the file system of the original Administration Server.

3. Make your configuration and security data available to the new administration machine by copying them from backups or by using a shared disk. For more information, refer to "Backing Up Configuration Data" on page 4-1 and "Backing Up Security Data" on page 4-2.

4. Restart the Administration Server on the new machine.

Note: Make sure that the startup command or startup script does not include `-Dweblogic.management.discover=false`, which disables an Administration Server from discovering its running Managed Servers. For more information about `-Dweblogic.management.discover`, refer to Frequently Used Optional Arguments in the *WebLogic Server Administration Guide*.

When the Administration Server starts, it communicates with the Managed Servers and informs them that the Administration Server is now running on a different IP address.

Starting a Managed Server When the Administration Server Is Not Accessible

Usually when a Managed Server starts, it contacts the Administration Server to retrieve its configuration information. If a Managed Server is unable to connect to the specified Administration Server during startup, it can retrieve its configuration by reading a configuration file and other files directly.

A Managed Server that starts in this way is running in Managed Server Independence mode. In this mode, a server uses its cached application files to deploy the applications that are targeted to the server. You cannot change a Managed Server's configuration until it is able to restore communication with the Administration Server.

This section contains the following subsections:

- Starting in Managed Server Independence Mode

Starting in Managed Server Independence Mode

If Managed Server Independence Mode is enabled (which is the default setting for a server), and if the Administration Server is unavailable when you start a Managed Server, a Managed Server looks in its root directory for the following files:

- A configuration file (`config.xml` by default)
- `SerializedSystemIni.dat`
- `boot.properties` (This is an optional file that contains an encrypted version of your username and password. For more information, refer to Bypassing the Prompt for Username and Password in the *WebLogic Server Administration Guide*.)

By default, a server assumes that its root directory is the directory from which you issue the server startup command.

If a Managed Server runs in the same domain and on the same machine as the Administration Server, by default it shares its root directory with the Administration Server. If you start such a Managed Server, it will automatically find the configuration files.

If a Managed Server does not share its root directory with the Administration Server, you can do any of the following:

- Copy the configuration files from the Administration Server's root directory (or from a backup) to the Managed Server's root directory. When you start the server, it will use the copied configuration files.
- Use the `-Dweblogic.RootDirectory=`*`path`* startup option to specify a root directory that already contains these files.
- Enable replication of configuration data, as described in "Replicating the Domain's Configuration Files" on page 4-15. This option requires that the

Managed Server has established contact with the Administration Server at least once prior to being started in Managed Server Independence mode.

Note: If you set up SSL for your servers, each server requires its own set of certificate files, key files, and other SSL-related files. Managed Servers do not retrieve SSL-related files from the Administration Server (though the domain's configuration file does store the pathnames to those files for each server). Starting in Managed Server Independence Mode does not require you to copy or move the SSL-related files unless they are located on a machine that is inaccessible.

The Node Manager and Managed Server Independence

You cannot use the Node Manager to start a server in Managed Server Independence mode. The Node Manager requires the presence of the Administration Server. If the Administration Server is unavailable, you must log on to a local host to start a Managed Server. For more information about the Node Manager, refer to "Managing Server Availability with Node Manager" on page 5-1.

MSI Mode and the Managed Servers Root Directory

By default, a server instance assumes that its root directory is the directory from which it was started. For more information about a server's root directory, refer to "A Server's Root Directory".

If you enable replication of configuration data, as described in "Replicating the Domain's Configuration Files" on page 4-15, and if you have started the Managed Server at least once while the Administration Server was running, `msi-config.xml` and `SerializedSystemIni.dat` will already be in the server's root directory. The `boot.properties` file is not replicated. If it is not already in the Managed Server's root directory, you must create one. For more information, see "Bypassing the Prompt for Username and Password" in the *WebLogic Server Administration Guide*.

If `msi-config.xml` and `SerializedSystemIni.dat` are not in the root directory, you can either:

- Copy `config.xml` and `SerializedSystemIni.dat` from the Administration Server's root directory (or from a backup) to the Managed Server's root directory. Then, rename the configuration file to `msi-config.xml`, or
- Use the `-Dweblogic.RootDirectory=`*`path`* startup option to specify a directory that already contains these files.

MSI Mode and the Domain Log File

Each WebLogic Server instance writes log messages to its local log file and a domain-wide log file. The domain log file provides a central location from which to view messages from all servers in a domain.

Usually, a Managed Server forwards messages to the Administration Server, and the Administration Server writes the messages to the domain log file. However, when a Managed Server runs in MSI mode, it assumes the role of writing to the domain log file.

By default, the pathnames for local log files and domain log files are relative to the Manged Server's root directory. With these default settings, if a Managed Server is located in its own root directory (and it does not share its root directory with the Administration Server), when it runs in MSI mode the Managed Server will create its own domain log file in its root directory.

If a Managed Server shares its root directory with the Administration Server, or if you specified an absolute pathname to the domain log, the Managed Server in MSI mode will write to the domain log file that the Administration Server created.

Note: The Managed Server must have permission to write to the existing file. If you run the Administration Server and Managed Servers under different operating system accounts, you must modify the file permissions of the domain log file so that both user accounts have write permission.

Contacting the Security Realm

In addition to the configuration files, a server must also have access to a security realm to complete its startup process.

If you use the security realm that WebLogic Server installs, then the Administration Server maintains an LDAP server to store the domain's security data. All Managed Servers replicate this LDAP server. If the Administration Server fails, Managed Servers running in Managed Server Independence mode can use the replicated LDAP server for security services.

If you use a third party security provider, then the Managed Server must be able to access the security data before it can complete its startup process.

Replicating the Domain's Configuration Files

Managed Server Independence mode includes an option that copies the required configuration files from the Administration Server's root directory into the Managed Server's root directory every 5 minutes. Depending on your backup schemes and the frequency with which you update your domain's configuration, this option might not be worth the performance cost of copying potentially large files across a network.

Before you can use this feature, you must initialize the required environment:

Caution: Do not enable file replication for a server that shares an installation or root directory with another server. Unpredictable errors can occur for both servers.

1. Start the domain's Administration Server.
2. Configure the Managed Server to replicate the domain's configuration files.

   See Replicating a Domain's Configuration Files in the Administration Console Online Help.
3. Leave the Administration Server running for at least 5 minutes.

After the Managed Server contacts the Administration Server and copies the configuration files to its own root directory, the Managed Server can use its copy of these files when starting in Managed Server Independence mode.

This option does not replicate a boot identity file. (For more information about boot identity files, refer to Bypassing the Prompt for Username and Password in the *WebLogic Server Administration Guide*.)

How a Managed Server Restores Communication with an Administration Server

When the Administration Server starts, it can detect the presence of running Managed Servers (if `-Dweblogic.management.discover=true`, which is the default setting for this property). Upon startup, the Administration Server looks at a persisted copy of the file `running-managed-servers.xml` and notifies all the Managed Servers of its presence. If the Managed Server is running in Managed Server Independence mode, it deactivates this self-administering mode and registers itself to the Administration Server for future configuration change notifications.

For information about restarting the Administration Server in this scenario, see "Restarting an Administration Server When Managed Servers are Running" on page 4-9.

Disabling Managed Server Independence

By default, Managed Server Independence mode is enabled. For information about disabling the mode, refer to Disabling Managed Server Independence in the Administration Console online help.

Self-Health Monitoring and Restart for Managed Servers

WebLogic Server 7.0 provides a new self-health monitoring feature to improve the reliability and availability of servers in a domain. Selected subsystems within each WebLogic Server monitor their health status based on criteria specific to the subsystem. (For example, the JMS subsystem monitors the condition of the JMS thread pool while the core server subsystem monitors default and user-defined execute queue statistics.) If an individual subsystem determines that it can no longer operate in a consistent and reliable manner, it registers its health state as "failed" with the host server.

Each WebLogic Server, in turn, checks the health state of all its registered subsystems to determine the overall viability of the server. If the server finds one or more critical subsystems have reached the `FAILED` state, the server marks its own health state as `FAILED` to indicate that the it cannot reliably host an application.

When used in combination with the Node Manager application, server self-health monitoring enables you to automatically reboot servers that have failed. This improves the overall reliability of a domain, and requires no direct intervention from an Administrator. See "Managing Server Availability with Node Manager" on page 5-1 for more information.

CHAPTER

5 Managing Server Availability with Node Manager

The following sections describe how to configure and use Node Manager in a WebLogic Server domain:

Overview of Node Manager

Node Manager is a standalone Java program provided with WebLogic Server that you can use to:

- Start remote Managed Servers.
- Restart Managed Servers that have shut down unexpectedly (for example, due to a system crash, hardware reboot, or server failure).

- Automatically monitor the health of Managed Servers and restart server instances that have reached the “failed” health state.
- Shut down or force the shut down of a Managed Server that has failed to respond to a shutdown request.

Environmental Considerations

The following sections describe the environment in which you run Node Manager.

Node Manager Runs on Machines that Host Managed Servers

To use Node Manager, you must configure and run one Node Manager process on each machine that hosts Managed Servers. You can manage multiple Managed Servers that reside on a single machine with a single Node Manager process. You can also control Managed Servers in different domains with a single Node Manager process.

Node Manager Should Run as a Service

Configure Node Manager to run as a daemon on UNIX machines or as a Windows service on Windows-based machines (see “Starting Node Manager as a Windows Service” on page 5-17). This ensures that Node Manager is available after a machine reboot.

Note: You cannot use Node Manager to start or kill an Administration Server. In a production environment, there is no need run Node Manager on a machine that runs an Administration Server, unless that machine also runs Managed Servers.

Node Manager Uses SSL to Communicate with Administration Servers

Communication between Node Manager and the Administration Server requires the Secure Socket Layer protocol, which provides authentication and encryption. You cannot use Node Manager features with an unsecured communication protocol.

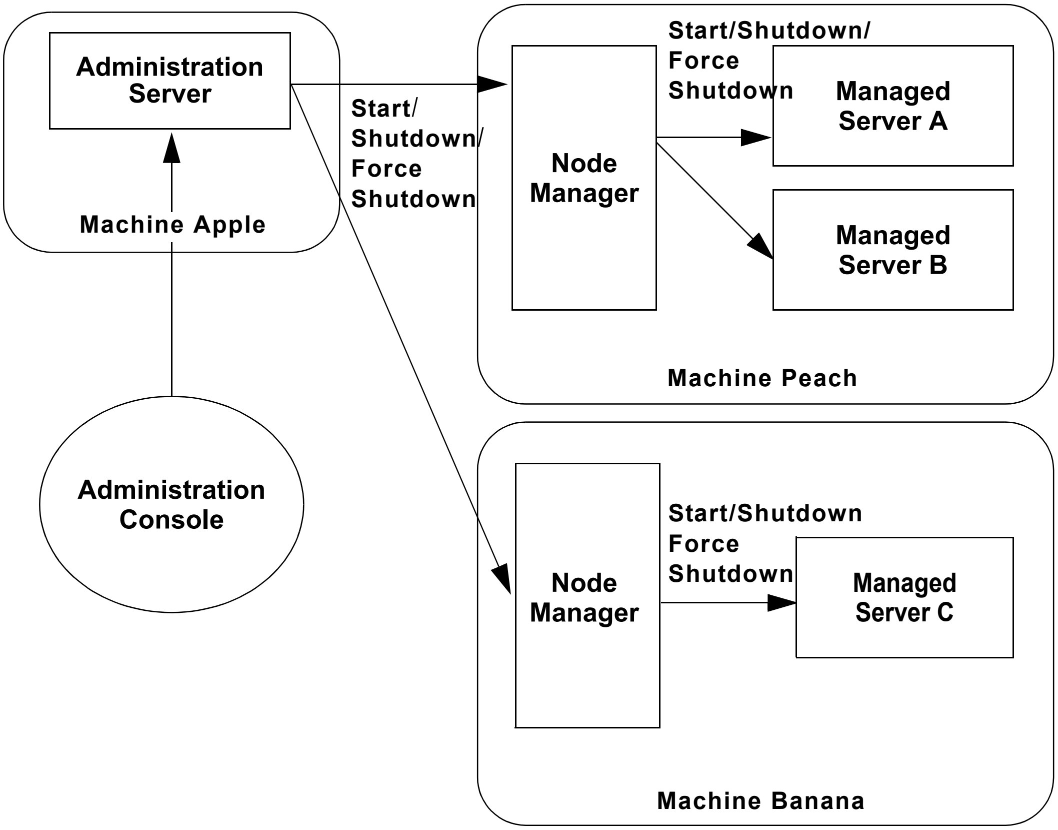
Native Support for Node Manager

BEA provides native Node Manager libraries for Windows, Solaris, and HP UX operating systems. For UNIX operating systems other than Solaris and HP UX, you must disable the `weblogic.nodemanager.nativeVersionEnabled` option at the command line when starting Node Manager. See "Starting Node Manager" on page 5-16 for more information.

Node Manager Architecture

The following figure shows the Node Manager architecture for a single WebLogic Server domain.

Figure 5-1 Node Manager Architecture



Node Manager works with other processes in a WebLogic Server domain to help you start and stop Managed Servers.

Users can issue requests to start, suspend, or shut down a Managed Server using the Administration Console. These requests are dispatched by the domain's Administration Server to the Node Manager process on the machine that hosts the Managed Server.

The Administration Server can also accept programmatic requests to change the state of a Managed Server. These requests, initiated by JMX clients such as the `weblogic.Admin` command-line utility, are dispatched to the Node Manager process on the machine that hosts the Managed Server.

Node Manager verifies that the actions requested by the Administration Server and programmatic clients are feasible, based on locally cached information on the state of Managed Servers it previously started. After verifying incoming requests, Node Manager forwards them to the target Managed Server for execution.

The target Managed Server performs the requested operation—shutdown, suspend, or resume. If the Managed Server fails to respond to a shutdown request from the Node Manager, the Node Manager process itself performs the shutdown.

A Node Manager is not part of a particular domain. You can use a single Node Manager process to start Managed Servers in any domain that it can access.

Node Manager starts a Managed Server in a dedicated process on the target machine, separate from the Node Manager and Administration Server processes. If you start a Managed Server using Node Manager, the associated status messages that are usually printed to `STDOUT` or `STDERROR` are displayed in the Administration Console and written to the Node Manager log file for that Managed Server, as described in "Managed Server Log Files" on page 5-25.

Node Manager Capabilities

The following sections describe key Node Manager capabilities. These capabilities are subject to the requirements and limitations described in "Prerequisites for Node Manager Capabilities" on page 5-7.

Node Manager Starts and Stops Managed Servers

As described in "Node Manager Architecture" on page 5-3, requests initiated by Administration Console users or JMX clients are forwarded by the Administration Server to the Node Manager process on the machine that hosts the target Managed Server. Node Manager verifies the feasibility of each request it receives and dispatches the verified request to the target Managed Server.

Note: Node Manager always starts a Managed Server in its last runtime state, as described in "Node Manager Ignores Server Startup Mode" on page 5-8.

Node Manager Monitors Managed Servers that it Started

WebLogic Server 7.0 subsystems perform self-health monitoring. If one or more of its critical subsystems fail, a Managed Server sets its health state to "failed." Node Manager periodically checks the health state of each Managed Server that it started.

For instructions on how to control how often Node Manager checks the health state of a Managed Server, and the length of time it will wait for a response from the Managed Server, see "Configure Monitoring, Shutdown and Restart for Managed Servers" on page 5-14.

For more information about how a Managed Server monitors its health, see "Server Self-Health Monitoring".

Node Manager Can Kill Unavailable and Failed Managed Servers

Node Manager can optionally shut down a Managed Server that reports its health status as "failed" or that does not respond to a health state query. By default this behavior is disabled. For instructions on how to enable this feature, see "Configure Monitoring, Shutdown and Restart for Managed Servers" on page 5-14.

Node Manager Automatically Restarts Managed Servers

By default, Node Manager attempts to restart a Managed Server whose health state is "failed". You can:

- Disable automatic restarts. For instructions, see "Configure Monitoring, Shutdown and Restart for Managed Servers" on page 5-14.
- Configure the duration of the period during which Node Manager will attempt to restart a Managed Server, and the number of times that Node Manager will attempt to restart a Managed Server. For instructions, see "Configure Monitoring, Shutdown and Restart for Managed Servers" on page 5-14.

Node Manager always starts a Managed Server in its last runtime state, as described in "Node Manager Ignores Server Startup Mode" on page 5-8.

Prerequisites for Node Manager Capabilities

Node Manager's ability to monitor the health of Managed Servers, and to automatically shutdown and restart Managed Servers is subject to these conditions:

- Server instances must be configured in accordance with the guidelines described in "Determine the Server Name and Listen Address" on page 2-6.
- Node Manager can automatically monitor, shutdown, and restart only those Managed Servers that were started with Node Manager. It cannot provide these services for Managed Servers booted directly from the command line.
- Node Manager will restart a Managed Server automatically only if the `AutoRestartEnabled` attribute for the server instance is selected.
- If the event that caused a Manager Server to fail also causes Node Manager to fail, additional prerequisites for automatic restart of the failed Managed Server apply:
  - The Node Manager process on the machine where the Managed Server runs must be restarted, either by the operating system or manually.
  - The Managed Server's `HostsMigratableServices` attribute must be set to "false". Node Manager will not automatically restart a Managed Server whose `HostsMigratableServices` is "true"—the default value for the attribute. Set the value of `HostsMigratableServices` using `serverMBean`.

    If a Managed Server fails, and the failure condition *did not* cause Node Manager to fail as well, Node Manager does not consider the setting of `HostsMigratableServices`.

The following scenarios exemplify the use of `HostsMigratableServices`. In Example 1, the value of `HostsMigratableServices` is not a factor in automatic restart of the Managed Server. In Example 2, the value of `HostsMigratableServices` is a factor in automatic restart of the Managed Server.

- Example 1—A Managed Server, started by Node Manager, crashes, and Node Manager continues to run. In this case, the value of `HostsMigratableServices` is *not* a factor in whether or not Node Manager restarts the Managed Server. If the `AutoRestartEnabled` attribute for the server instance is selected, Node Manager will restart the Managed Server.

- Example 2—A machine with a running Node Manager process and a Managed Server that was started by the Node Manager is re-booted. Node Manager is set up to run as a service on the machine, and starts up automatically at re-boot. Node Manager detects that the Managed Server is not running. In this case, the value of `HostsMigratableServices` *is* a factor in whether or not Node Manager restarts the Managed Server. If the value of `HostsMigratableServices` is false, and the `AutoRestartEnabled` attribute for the server instance is selected, Node Manager will restart the Managed Server.

Node Manager Ignores Server Startup Mode

When Node Manager starts a Managed Server, either automatically or as a result of a user command or script, it does not use the startup mode configured for the server. (A Managed Server's startup mode can be specified with the `-Dweblogic.management.startupMode` command argument, or in the Administration Console on the Server's Configuration->General tab). Instead, Node Manager always starts a Managed Server in its last runtime state.

For example, if a Managed Server configured to start up in `STANDBY` mode fails while in the `RUNNING` state, Node Manager will automatically restart the server instance in `RUNNING` state (assuming Node Manager is configured to monitor the health of Managed Servers, and the prerequisites described in "Prerequisites for Node Manager Capabilities" on page 5-7 are met.)

For more information on server startup mode, see "Starting and Stopping WebLogic Servers" in the *Administration Guide*.

Configuring Node Manager

The following tasks are required to use Node Manager in a WebLogic Server domain

- "Set Up the Node Manager Hosts File" on page 5-9
- "Configure SSL for Node Manager" on page 5-10
- "Configure a Machine to Use Node Manager" on page 5-11

- "Configure Startup Arguments for Managed Servers" on page 5-12

After you perform these tasks, you can use Node Manager, via the Administration Console to explicitly start and stop Managed Servers. By default the following behaviors will be configured:

- Automatic restart of Managed Servers is enabled. Node Manager will try up to two times within a 300 second interval to restart a Managed Server that reports its health status as "failed".
- Automatic shutdown of Managed Servers is disabled. Node Manager will not shutdown Managed Servers that it detects to be in the "failed state".
- Node Manager monitor the Managed Servers that it has started—it will check the self-reported health status of each Managed Server every 180 seconds. It will wait 60 seconds for a response from a Managed Server.

To tailor these configuration settings, follow the instructions in "Configure Monitoring, Shutdown and Restart for Managed Servers" on page 5-14

Set Up the Node Manager Hosts File

Node Manager accepts commands only from Administration Servers that reside on a trusted hosts. The trusted hosts for a Node Manager process are identified by IP address or DNS name in a file, which by default is named `nodemanager.hosts` and installed in the `*WL_HOME*\common\nodemanager\config` directory, where `*WL_HOME*` is the top-level installation directory for the WebLogic Server.

Note: You can specify a different name and location for the trusted hosts file using a Node Manager command-line argument. For more information, see "Node Manager Command-Line Arguments" on page 5-18.

`nodemanager.hosts` contains one line for each trusted host on which an Administration Server runs. By default, `nodemanager.hosts` file contains two entries:

```
localhost
127.0.0.1
```

To add or remove trusted hosts, edit the file with a text editor.

If you identify a trusted host by its DNS name, you must enable reverse DNS lookup when starting Node Manager. Reverse DNS lookup is enabled via the command-line argument:

```
-Dweblogic.nodemanager.reverseDnsEnabled=true
```

By default, reverse DNS lookup is disabled.

In a typical production environment Node Manager might not be running on the same machine as the Administration Server. Edit `nodemanager.hosts` to list only machines that host an Administration Server.

Configure SSL for Node Manager

Node Manager uses 2-way SSL to protect communications with Administration and Managed Servers. The configuration of SSL involves obtaining identity and trust for the Node Manager and each Administration and Managed Server with which the Node Manager will be communicating and then configuring the Node Manager, the Administration Server, and any Managed Servers with the proper identity and trust. In addition, the use of host name verification must be taken into consideration.

In WebLogic Server 7.0 and later, Node Manager uses the same certificate format and public key infrastructure used by WebLogic Server 7.0. (Prior to WebLogic Server 7.0, Node Manager used a single certificate file containing both a password-protected private key and user identity certificate.)

For general information about SSL, see “SSL: An Introduction” in *Managing WebLogic Security*. For instructions to configure SSL for Node Manager, see “Configuring SSL for the Node Manager” in Managing WebLogic Security.

Starting Node Manager with SSL

When you start a Node Manager process to use SSL, you must specify startup arguments that identify the keystore, password, and certificate files to use for SSL communication:

- `-Dweblogic.nodemanager.keyFile`—specifies the path to the encrypted key file.
- `-Dweblogic.nodemanager.keyPassword`—specifies the password to use if the key file is encrypted.

- `-Dweblogic.nodemanager.certificateFile`—specifies the path to the certificate file.
- `-Dweblogic.security.SSL.trustedCAKeyStore`—specifies the path to the key store that holds a private key.
- `-Dweblogic.nodemanager.sslHostNameVerificationEnabled`—enables or disables Administration Server hostname checks against the `nodemanager.hosts` file. Disable these checks only when you are using the demonstration certificates.

For example, to start Node Manager using the SSL configuration provided with the sample SSL certificate and key files:

```
java.exe -Xms32m -Xmx200m  -classpath %CLASSPATH% -Dbea.home=c:\bea
-Dweblogic.nodemanager.keyFile=e:\bea\user_domains\mydomain\demok
ey.pem
-Dweblogic.security.SSL.trustedCAKeyStore=e:\bea\weblogic700\serv
er\lib\cacerts
-Dweblogic.nodemanager.certificateFile=e:\bea\user_domains\mydoma
in\democert.pem
-Djava.security.policy=e:\weblogic700\server\lib\weblogic.policy
-Dweblogic.nodemanager.sslHostNameVerificationEnabled=false
weblogic.nodemanager.NodeManager
```

See "Node Manager Command-Line Arguments" on page 5-18 for information about all Node Manager command-line arguments.

Configure a Machine to Use Node Manager

Before using Node Manager you must create a machine definition for each machine that runs a Node Manager process.

1. With the Administration Server running, invoke the Administration Console (if it isn't already running).
2. Select the Machines node in the left pane to display the machines table.
3. Select Configure a new Machine (or Configure a new UNIX Machine).
4. Enter a name for the new machine and click Create to create the new machine entry.

5. Select the new machine's Node Manager tab. Fill in connection attributes as follows:
   - *Listen Address*: Enter the host name or IP address where Node Manager listens for requests.
   - *Listen Port*: Enter the port number where Node Manager listens for requests.
6. Click Apply to apply your changes.
7. Select the new machine's Servers tab. For each Managed Server that the machine hosts, select the server instance in the Available column, and click the appropriate arrow to move the server instance to the Chosen column.
8. Click Apply to apply your changes.

Configure Startup Arguments for Managed Servers

When Node Manager starts a Managed Server, it uses startup command arguments, just as you do if you start a server instance at the Java command line or in a startup script. See "Starting and Stopping WebLogic Servers" in the *Administration Guide* for more information on individual arguments.

If you do not specify optional startup arguments for a Managed Server, Node Manager uses its own properties as defaults to start the server instance. Although these defaults are sufficient to boot a server, you should enter your own custom startup arguments to ensure that each server boots in a consistent and reliable manner.

To configure the startup arguments that Node Manager will use to start a Managed Server:

1. Invoke the Administration Console, if it isn't already running.
2. In the left pane, select Servers, then select the name of the server instance you want to configure.
3. In the right pane, select Configuration→Remote Start. Edit the startup arguments as follows:
   - Java Home—Enter the full path to the Java runtime environment to use when starting this Managed Server (for example, `c:\bea\jdk131`).

- BEA Home—Enter the full path to the BEA Home directory. This is the root directory under which all BEA products and licenses were installed.
- Root Directory—Enter the root directory for the domain in which this Managed Server resides (for example, `c:\bea\Weblogic7.0\samples\server\config\examples`).

  If you do not specify a Root Directory value, the default Node Manager work directory is used (generally *`WL_HOME`*`\common\nodemanager`).
- Class Path—Enter the full class path required to start the Managed Server. At a minimum you will need to specify the following values for the class path option:

  `WL_HOME``/server/lib/weblogic_sp.jar`

  `WL_HOME``/server/lib/weblogic.jar`

  See "Starting and Stopping WebLogic Servers" in the *Administration Guide* for more information about class path requirements.
- Arguments—Enter any additional arguments to pass to the JVM when starting this server. These are the first arguments appended immediately after `java` portion of the startup command. For example, you can set Java heap memory using the `-Xms32m` and `-Xmx200m` arguments, or you can enter debugging options for a specific application.
- Security Policy File—Enter the full path to the WebLogic security policy file (for example, `c:\bea\weblogic700\server\lib\weblogic.policy`).
- Username (Required)—Enter the name of a user with privileges to boot this Managed Server (for example, `weblogic`).

Note: In WebLogic Server version 7.0, you must enter user name and password to start a Managed Server. Managed Servers do not inherit the user name and password that you supplied to the Administration Server.

- Password (Required)—Click change... to change the password used to start the server (for example, `weblogic`). Type the new password in the New Password and Retype to confirm text boxes, then click Apply to apply the change.

4. Click Apply.

Configure Monitoring, Shutdown and Restart for Managed Servers

Follow these steps in this section to configure Node Manager features for:

- Monitoring Managed Servers. See "Node Manager Monitors Managed Servers that it Started" on page 5-6 for a description of this feature.
- Forcing the shutdown of Managed Servers whose health state is "failed". See "Node Manager Can Kill Unavailable and Failed Managed Servers" on page 5-6f or a description of this feature.
- Restarting Managed Servers. See "Node Manager Automatically Restarts Managed Servers" on page 5-6 for a description of this feature.

  Note: These features are available when the conditions described in "Prerequisites for Node Manager Capabilities" on page 5-7 are met.

1. If you have not yet done so, configure the Managed Server's startup information as described in "Configure Startup Arguments for Managed Servers" on page 5-12.
2. Access the Administration Console for the domain that contains the Managed Server.
3. Select the Servers entry in the left pane, then select the name of the server you want to configure.
4. Select the Configuration->Health Monitoring tab to display the Managed Server's automatic restart and monitoring attributes.
5. To change the server's restart behavior, edit the following attributes:
   - Ensure that the `Auto Restart` check box in the right pane is selected.

     When `Auto Restart` is enabled, Node Manager will try to restart the Managed Server after failure. Deselect the check box if you do not Node Manager to automatically restart the Managed Server after failures.
   - `Restart Interval`: Enter the period of time for which Node Manager should attempt to restart the Managed Server. This attribute is used in conjunction with the `Max Restarts within Interval` attribute to limit attempts to restart this server. Enter a value between 3600 and 2147483647 seconds (default is 300 seconds).

- `Max Restarts within Interval`: Enter the maximum number of times Node Manager can restart this server within the interval specified by `Restart Interval` above. Enter a value between 0 and 2147483647 (default is 2 restarts).

6. To configure automatic shutdown and health monitoring features and edit the following attributes:

   - `Auto Kill If Failed`: Check this box to enable Node Manager to automatically kill the Managed Server when its health state is “failed,” or when Node Manager cannot query the Managed Server’s health state. By default, this option is disabled.
   - `Health Check Interval`: Specify the interval (in seconds) between Node Manager health state queries to the Managed Server. Enter a value between 1 and 2147483647 seconds (default is 180 seconds).
   - `Health Check Timeout`: Enter the number of seconds that Node Manager will await a response to a health state query. If the timeout is reached, Node Manager assumes the Managed Server is “failed”, and kills the process. Enter a value between 1 and 2147483647 seconds (default is 60 seconds).
   - `Restart Delay Seconds`: Enter the number of seconds that Node Manager should wait before trying to restart a Managed Server. After killing a server process, the system may need several seconds to release the TCP port(s) the server was using. If Node Manager attempts to restart the Managed Server while its ports are still active, the startup attempt fails.

7. Click Apply to apply your changes.

8. If you did not start the Managed Server using Node Manager, shut it down, and restart it using Node Manager. Node Manager cannot perform automatic monitoring and monitoring and shutdown on a server that it did not start.

Starting Node Manager

You start the Node Manager process manually, using the `java` command at an operating system prompt, or automate the process using scripts. In a production environment, make sure that Node Manager starts automatically when a machine is booted, by creating a startup script (for UNIX systems) or by setting up Node Manager as a Windows service.

Node Manager relies on the same environment variables and command-line options regardless of how you choose to start the Node Manager process.

Note: Always specify a minimum heap size of 32 megabytes (`-Xms32m`) when starting Node Manager to avoid running out of memory. The Node Manager start script and `installNodeMgrSvc.cmd script` automatically set this Java argument.

Starting Node Manager Using Start Scripts

BEA provides sample start scripts to help you begin using Node Manager. These scripts are installed in the *`WL_HOME`*`\server\bin` directory. The standard Node Manager start script is named `startNodeManager.cmd` on Windows systems and `startNodeManager.sh` on UNIX systems. The scripts set the required Node Manager environment values and start the Node Manager process.

When you first use the Node Manager start script, the script changes to the directory *`WL_HOME`*`/common/nodemanager` where *`WL_HOME`* is the WebLogic Server installation directory. Node Manager uses this directory as a working directory for storing output and log files. If you want to specify a different working directory, edit the start script with a text editor and set the value of the `NODEMGR_HOME` variable to a directory of your choice.

You will also need to edit the sample start script to make sure that the Node Manager startup arguments set the correct listen address and port number for your Node Manager process.

Starting Node Manager as a Windows Service

The directory `WL_HOME\server\bin` (where `WL_HOME` is the top-level directory for the WebLogic platform) contains two scripts for installing and uninstalling the Node Manager as a Windows service: `installNodeMgrSvc.cmd` and `uninstallNodeMgrSvc.cmd`.

To use either script, simply invoke the script name at the command line. `installNodeMgrSvc.cmd` creates a default Windows service for the Node Manager application named `NodeManager_localhost_5555`. As with the Node Manager start scripts described above, you may want to edit `installNodeMgrSvc.cmd` before running the script to change the service name or to use non-default environment variables or startup arguments.

`uninstallNodeMgrSvc.cmd` uninstalls the default node manager service. If you edit `installNodeMgrSvc.cmd` to change the listen port or hostname, make the same change to `uninstallNodeMgrSvc.cmd` so that it removes the correct service name.

Node Manager Environment Variables

Before starting Node Manager, you must set several different environment variables. One way to set all of the required environment variables for a domain is to run the environment setup scripts provided with WebLogic Server, as described in "Starting Node Manager Using Start Scripts" on page 5-16 and "Starting Node Manager as a Windows Service" on page 5-17.

If you are starting Node Manager directly from the command line, use the information in the table below to set required environment variables.

Figure 5-2 Node Manager Environment Variables

Environment Variable	Description
`JAVA_HOME`	Make sure that the `JAVA_HOME` environment variable points to the root directory where you installed the JDK that you are using for Node Manager. For example: `set JAVA_HOME=c:\bea\jdk131` Node Manager has the same JDK version requirements as WebLogic Server.
`WL_HOME`	`WL_HOME` specifies the WebLogic Server installation directory. For example: `set WL_HOME=c:\bea\weblogic700`
`PATH`	The `PATH` environment variable should include the WebLogic Server `bin` directory as well as the path to your Java executable. For example: `set PATH=%WL_HOME%\server\bin;%JAVA_HOME%\bin;%PATH%`
`LD_LIBRARY_PATH` (UNIX only)	For HP UX and Solaris systems, make sure `LD_LIBRARY_PATH` includes the path to the native Node Manager libraries. The following is an example on Solaris: `LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$WL_HOME/server/lib/solaris:$WL_HOME/server/lib/solaris/oci816_8` The following is an example on HP UX: `SHLIB_PATH=$SHLIB_PATH:$WL_HOME/server/lib/hpux11:$WL_HOME/server/lib/hpux11/oci816_8`
`CLASSPATH`	You can set the Node Manager `CLASSPATH` either as an option on the `java` command line used to start Node Manager, or as an environment variable. The following is an example (on Windows NT) of setting the classpath as an environment variable: `set CLASSPATH=.;%WL_HOME%\server\lib\weblogic_sp.jar;%WL_HOME%\server\lib\weblogic.jar`

Node Manager Command-Line Arguments

The basic syntax for starting start Node Manager is:

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

In the command line, a `java_property` indicates a direct argument to the `java` executable, such as `-ms` or `-mx`. If you did not set the `CLASSPATH` environment variable, use the `-classpath` option to identify required Node Manager classes.

Note: Always specify a minimum heap size of 32 megabytes (`-Xms32m`) with Node Manager to avoid running out of memory. The Node Manager start scripts installed with WebLogic Server automatically set this Java argument.

A `nodemanager_property` begins with the prefix `weblogic.property` and directly affects the behavior of the Node Manager process. The table below describes all Node Manager properties.

Table 5-1 Node Manager Properties

Node Manager Property	Description	Default
weblogic.nodemanager.certificateFile	Specifies the path to the certificate file used for SSL authentication.	./config/democert.pem
weblogic.nodemanager.javaHome	Specifies the Java home directory that Node Manager uses to start Managed Servers on this machine.	none
weblogic.nodemanager.keyFile	The path to the private key file to use for SSL communication with the Administration Server.	./config/demokey.pem
weblogic.nodemanager.keyPassword	The password used to access the encrypted private key in the key file.	password
weblogic.nodemanager.listenAddress (Deprecated)	The address on which Node Manager listens for connection requests. Use weblogic.ListenAddress in place of this deprecated argument.	localhost

Node Manager Property	**Description**	**Default**
weblogic.nodemanager. listenPort (Deprecated)	The TCP port number on which Node Manager listens for connection requests. Use weblogic.ListenPort in place of this deprecated argument.	5555
weblogic.ListenAddress	The address on which Node Manager listens for connection requests. This argument deprecates weblogic.nodemanager.listenAddress.	localhost
weblogic.ListenPort	The TCP port number on which Node Manager listens for connection requests. This argument deprecates weblogic.nodemanager.listenPort.	5555
weblogic.nodemanager. nativeVersionEnabled	For UNIX systems other than Solaris or HP-UX, 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 `StartTemplate` property to start Managed Servers.	true
weblogic.nodemanager. reverseDnsEnabled	Specifies whether entries in the trusted hosts file can contain DNS names (instead of IP addresses).	false
weblogic.nodemanager. savedLogsDirectory	Specifies the path in which Node Manager stores log files. Node Manager creates a subdirectory in the `savedLogsDirectory` named `NodeManagerLogs`.	./NodeManagerLogs
weblogic.nodemanager. sslHostNameVerificationEnabled	Determines whether or not Node Manager performs host name verification.	false

Node Manager Property	**Description**	**Default**
weblogic.nodemanager.startTemplate	For UNIX systems only, this property specifies the path of a script file used to start Managed Servers. **Note:** Node Manager will use the specified start script to start Managed Servers only if the `NativeVersionEnabled` property is set to `false.` `StartTemplate` is used only for Unix systems that do not have If you create your own start template script, refer to `nodemenager.sh` as an example.native Node Manager libraries.	./nodemanager.sh
weblogic.nodemanager.trustedHosts	The path to the trusted hosts file that Node Manager uses. See "Set Up the Node Manager Hosts File" on page 5-9.	./nodemanager.hosts
weblogic.nodemanager.weblogicHome	Specifies the root directory of the WebLogic Server installation. This is used as the default value of -Dweblogic.RootDirectory for servers that do not have a configured root directory.	n/a

Node Manager uses *`server_property`* values as default values when starting new Managed Server instances. The table below describes all *`server_property`* values.

Table 5-2 Server Properties Used by Node Manager

Server Property	Description	Default
bea.home	Specifies the BEA home directory that Managed Servers use on this machine.	none
java.security.policy	Specifies the path to the security policy file that Managed Servers use.	none
weblogic.security.SSL.trustedCAKeyStore	The path to the KeyStore that contains certificates of trusted authorities.	java.security.keyStore

Note: If you wish to start Node Manager on a UNIX operating system other than Solaris or HP UX, you cannot have any white space characters in any of the parameters that will be passed to the `java` command line in starting Node Manager. For example, if you try to use the parameter

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

the command fails due to the space character in the name `big iron`.

Starting and Stopping Managed Servers with Node Manager

These sections describe how to use Node Manager to:

- "Start and Stop Managed Servers Manually" on page 5-23
- "Start and Stop All Managed Servers in a Domain or Cluster" on page 5-24
- "Start and Stop Servers Using weblogic.Admin" on page 5-24

Start and Stop Managed Servers Manually

You can start a Managed Server using the Administration Console if Node Manager is running on the machine that hosts the Managed Server.

When start a Managed Server in this fashion, Node Manager starts the server instance in a separate process. Messages that are usually printed to `STDOUT` or `STDERROR` when starting a WebLogic Server are instead displayed in the right pane of the Administration Console. These messages are also written to the Node Manager log file for that server, as described in "Managed Server Log Files" on page 5-25.

Node Manager always starts a Managed Server in its last runtime state, ignoring the server's configured startup mode. See "Node Manager Ignores Server Startup Mode" on page 5-8 for more information.

To start a Managed Server using Node Manager:

1. Invoke the Administration Console (if it isn't already running).
2. Select the Servers node, then select the name of the configured server you want to start or stop.
3. Select the Control->start-stop tab in the right-hand pane.
4. Select one of the following options to start or stop the server using Node Manager:
   - `Start this server...` starts the Managed Server using Node Manager.
   - `Start this server in standby mode...` boots a server to the suspend state.

   Note: Node Manager is not required to use the `Shutdown this server...` or `Force shutdown of this server...` options. However, Node Manager is used to complete these shutdown tasks (if configured) if the server fails to respond to the shutdown request.

   See "Starting and Stopping WebLogic Server" in the *Administration Guide* for more information about server states.

   Note: If you select the Administration Server, rather than a Managed Server, you can only choose the `Shutdown this server...` option.
5. Click Yes or No to verify your selection.

Start and Stop All Managed Servers in a Domain or Cluster

You can start or shutdown all Managed Servers in a domain or cluster using the Administration Console if Node Manager is running on each machine that hosts the Managed Servers in the domain or cluster.

To start all Managed Servers in a domain or cluster:

1. Right click on the name of the active domain in the left panel.
2. Select **Kill this domain...** or **Start this domain...**

If you start the entire domain from the Administration Console, the results displayed in the right pane will consist of a series of links to the results for each Managed Server that was configured for that domain.

To shutdown all Managed Servers in a domain or cluster:

1. Right click on the name of the cluster in the left panel.
2. Select **Kill this cluster...** or **Start this cluster...**

Note: You cannot start or kill the Administration Server using Node Manager.

Start and Stop Servers Using weblogic.Admin

You can use the `weblogic.Admin` utility with the `START`, `SHUTDOWN`, and `FORCESHUTDOWN` commands to start and kill Managed Servers. To `START` a Managed Server, Node Manager must be running on the host. Node Manager is not required to `SHUTDOWN` or `FORCESHUTDOWN` a server instance; however, if the Managed Server cannot successfully complete a shutdown request, Node Manager performs the operation.

See "WebLogic Server Command-Line Interface Reference" in the *Administration Guide* for more information on using `weblogic.Admin`.

Troubleshooting Node Manager

The following sections describe how to diagnose and correct Node Manager problems. Use the Node Manager log files to help troubleshoot problems in starting or stopping individual Managed Servers. Use the steps in "Correcting Common Problems" on page 5-26 to solve problems in Node Manager configuration and setup.

Managed Server Log Files

Note: To reclaim disk space, you should periodically delete the accumulated log files from past Node Manager startup and shutdown attempts.

When you start WebLogic Server, startup or error messages are printed to `STDOUT` or `STDERROR`. These messages can be retrieved by right clicking on the server in the left pane of the Administration Console and selecting the option **View Server log**.

Node Manager saves these messages in the `NodeManagerLogs` directory. By default `NodeManagerLogs` is created in the directory where you start Node Manager. A separate log file subdirectory is created for each Managed Server that Node Manager starts on the machine. Each subdirectory uses the naming convention *`domain_server`* which designates the domain name and Managed Server name, respectively.

Logs files stored in the server directory include:

`servername``_pid`
: This file saves the process ID of the Managed Server named *`servername`*. This is used by Node Manager to kill the server process when requested by the Administration Server to do so.

`config.xml`
: This file saves startup configuration information passed to Node Manager from the Administration Server when starting a Managed Server.

`servername``_output.log`
: This file saves Node Manager startup messages generated when Node Manager attempts to start a Managed Server. If a new attempt is made to start the server, this file is renamed by appending `_PREV` to the file name.

`*servername*_error.log`

This file saves Node Manager error messages generated when Node Manager attempts to start a Managed Server. If a new attempt is made to start the server, this file is renamed by appending `_PREV` to the file name.

The domain's Administration Server also stores a copy of the Managed Server log files in the a directory name `NodeManagerClientLogs`. This directory is created one directory level above the Administration Server's root directory (by default, the directory in which you started the server). The `NodeManagerClientLogs` directory contains a subdirectory for each Managed Server you attempted to start via Node Manager. Each log in these subdirectories corresponds to an attempt to carry out some action, such as starting or killing the server process. The name of the log file includes a timestamp that indicates the time at which the action was attempted.

You can view the standard output and error messages for a server, as well as Node Manager's log messages for a particular Managed Server, using the console. Simply select the server that was started using manager, then select the Monitoring->Process Output tab.

Node Manager Log Files

Note: Because Node Manager creates a new log file (with unique timestamp) each time it starts, you should periodically remove the `NodeManagerLogs` subdirectory to reclaim the space used by old log files.

In WebLogic Server 7.0 Node Manager also generates its own log files, which contain Node Manager startup and status messages that were written to `STDOUT` in previous Node Manager versions. Node Manager log files are placed in the `NodeManagerLogs\NodeManagerInternal` subdirectory. The log files using the naming convention `NodeManagerInternal_`*`timestamp`*, where *`timestamp`* indicates the time at which Node Manager started.

Correcting Common Problems

The table below describes common Node Manager problems and their solutions.

Table 5-3 Troubleshooting Node Manager Problems

Symptom	Explanation
Error message: `Could not start server 'MyServer' via Node Manager - reason: 'Target machine configuration not found'.`	You have not assigned the Managed Server to a machine. Follow the steps under "Configure a Machine to Use Node Manager" on page 5-11.
Error message: `<SecureSocketListener: Could not setup context and create a secure socket on 172.17.13.26:7001>`	The Node Manager process may not be running on the designated machine. See "Starting Node Manager" on page 5-16.
I configured self-health monitoring attributes for a server, but Node Manager doesn't automatically restart the server.	To automatically reboot a server, you must configure the server's automatic restart attributes as well as the health monitoring attributes. See "Configure Monitoring, Shutdown and Restart for Managed Servers" on page 5-14 and "Starting Node Manager" on page 5-16. In addition, you must start Managed Servers using Node Manager. You cannot automatically reboot servers that were started outside of the Node Manager process (for example, servers started directly at the command line).

Symptom	Explanation

Applications on the Managed Server are using the wrong directory for lookups.	Applications deployed to WebLogic Server should never make assumptions about the current working directory. File lookups should generally take place relative to the Root Directory obtained with the `ServerMBean.getRootDirectory()` method (this defaults to the "." directory). For example, to perform a file lookup, use code similar to
	
	`String rootDir =`
	`ServerMBean.getRootDirectory();`
	` //application root directory`
	`File f = new File(rootDir + File.separator +`
	` "foo.in");`
	rather than simply:
	
	`File f = new File("foo.in");`
	If an application is deployed to a server that is started using Node Manager, use the following method calls instead:
	
	`String rootDir //application root directory`
	`if ((rootDir =`
	`ServerMBean.getRootDirectory()) ==`
	` null) rootDir =`
	` ServerStartMBean.getRootDirectory();`
	`File f = new File(rootDir + File.separator +`
	` "foo.in");`
	The `ServerStartMBean.getRootDirectory()` method obtains the Root Directory value that you specified when configuring the server for startup using Node Manager. (This corresponds to the Root Directory attribute specified the Configuration->Remote Start page of the Administration Console.)

Node Manager Internal States

Node Manager defines its own, internal Managed Server states for use when restarting a server. You may observe these states when working with Node Manager in a domain:

- `FAILED_RESTARTING`—Indicates that Node Manager is currently restarting a failed Managed Server.

- `ACTIVATE_LATER`—Indicates that `MaxRestart` restart attempts have been made in current `RestartInterval`, and Node Manager will attempt additional restarts in the next `RestartInterval`.
- `FAILED_NOT_RESTARTABLE`—Indicates that the server is Failed, but Node Manager cannot restart it because the server's `AutoRestart` or `AutoKillIfFailed` attribute is set to False.
- `FAILED_MIGRATABLE`—Indicates that the server is Failed, but Node Manager cannot restart it because the server's `HostsMigratableServices` attribute is set to True.

6 Monitoring a WebLogic Server Domain

The following sections explain how to monitor your WebLogic Server domain:

Overview of Monitoring

The tool for monitoring the health and performance of your WebLogic Server domain is the Administration Console. The Administration Console allows you to view status and statistics for WebLogic Server resources such as servers, HTTP, the JTA subsystem, JNDI, security, CORBA connection pools, EJB, JDBC, and JMS.

Monitoring information is presented in the right pane of the Administration Console. You access a page by selecting a container or subsystem, or a particular entity under a container, on the hierarchical domain tree, in the left pane.

The Administration Console provides three types of page that contain monitoring information:

- Monitoring tab pages for a particular entity (such as an instance of a JDBC Connection Pool or a particular server's performance)

- Tables of data about all entities of a particular type (such as the WebLogic Servers table)
- Views of the domain log and of the local server logs. For information about log messages, see Using Log Messages to Manage WebLogic Servers.

The Administration Console obtains information about domain resources from the Administration Server. The Administration Server, in turn, is populated with Management Beans (MBeans), based on Sun's Java Management Extension (JMX) standard, which provides the scheme for management access to domain resources.

The Administration Server contains both configuration MBeans, which control the domain's configuration, and run-time MBeans. Run-time MBeans provide a snapshot of information about domain resources, such as JVM memory usage or the status of WebLogic Servers. When a particular resource in the domain (such as a Web application) is instantiated, an MBean instance is created which collects information about that particular resource.

When you access a monitoring page for particular resources in the Administration Console, the Administration Server performs a GET operation to retrieve the current attribute values.

The following sections describe some of the monitoring pages that are useful for managing a WebLogic Server domain. These pages have been selected simply to illustrate the facilities provided by the Administration Console.

Monitoring Servers

The servers table and the monitoring tab pages for individual servers enable you to monitor WebLogic Servers. The servers table provides a summary of the status of all servers in your domain. If only a small subset of the log messages from the server are forwarded to the domain log, accessing the local server log may be useful for troubleshooting or researching events.

For more information about the log files and the logging subsystem, see Using Log Messages to Manage WebLogic Servers.

You can access monitoring data for each WebLogic Server from the monitoring tabs for that server. The Logging tab provides access to the local log for the server (that is, the log on the machine where the server is running).

The Monitoring→General tab page indicates the current status and provides access to the Active Queues table, the Connections table, and the Active Sockets table. The Active Execute Queues table provides performance information such as the oldest pending request and the queue throughput.

Performance

The Monitoring→Performance tab graphs real-time data on JVM memory heap usage, request throughput, and queue length. This tab page also enables you to force the JVM to perform garbage collection on the memory heap.

The Java heap is a repository for Java objects (live and dead). Normally you do not need to perform garbage collection manually because the JVM does this automatically. When the JVM begins to run out of memory, it halts all execution and uses a garbage collection algorithm to free up space no longer used by Java applications.

On the other hand, developers debugging applications may have occasion to force garbage collection manually. Manual garbage collection may be useful, for example, if they are testing for memory leaks that rapidly consume JVM memory.

Security

The Monitoring→Security tab provides statistics about invalid login attempts and locked and unlocked users.

JMS

The Monitoring→JMS tab provides statistics on JMS servers and connections. This page also provides links to the tables of active JMS connections and active JMS servers, which monitor such attributes as total current sessions.

JTA

The Monitoring→JTA tab provides statistics on the Java Transactions subsystem such as total transactions and total rollbacks. The page provides links to tables that list transactions by resource and name, and a table of in-flight transactions.

Server Self-Health Monitoring

WebLogic Server 7.0 provides a new self-health monitoring feature to improve the reliability and availability of servers in a domain. Selected subsystems within each WebLogic Server monitor their health status based on criteria specific to the subsystem. (For example, the JMS subsystem monitors the condition of the JMS thread pool while the core server subsystem monitors default and user-defined execute queue statistics.) If an individual subsystem determines that it can no longer operate in a consistent and reliable manner, it registers its health state as "failed" with the host server.

Each WebLogic Server, in turn, checks the health state of all its registered subsystems to determine the overall viability of the server. If the server finds one or more critical subsystems have reached the "failed" state, the server marks its own health state as "failed" to indicate that the it cannot reliably host an application.

When used in combination with the Node Manager application, server self-health monitoring enables you to automatically reboot servers that have failed. This improves the overall reliability of a domain, and requires no direct intervention from an Administrator. See "Managing Server Availability with Node Manager" on page 5-1 for more information.

In WebLogic Server version 7.0, the health state of a server and its registered subsystems is not visible via the Administration Console. However, you can check the health state of a server programmatically by calling the `getHealthState()` method on the `ServerRuntimeMBean`. Similarly, you can obtain the health state of a registered WebLogic Server subsystem by calling the `getHealthState()` method on its MBean. In WebLogic Server version 7.0, the following MBeans automatically register their health states with the host server:

- `JMSRuntimeMBean`
- `JMSServerRuntimeMBean`
- `JTARuntimeMBean`
- `TransactionResourceRuntimeMBean`

See the Javadocs for WebLogic Classes for more information on individual MBeans.

You can also configure the frequency and timing of automated self-health checks by setting attributes of the `ServerMBean`. See "Managing Server Availability with Node Manager" on page 5-1 for information on setting these attributes using the Administration Console.

Monitoring JDBC Connection Pools

Java Database Connectivity (JDBC) subsystem resources can also be monitored via the Administration Console. The Monitoring tab for a JDBC connection pool allows you to access a table listing statistics for the instances of that pool. As with other entity tables in the Administration Console, you can customize the table to select which attributes you want to be displayed.

A number of these attributes provide important information for managing client database access.

The Waiters High field indicates the highest number of clients waiting for a connection at one time. The Waiters field tells you how many clients are currently waiting for a connection. The Connections High field indicates the highest number of connections that have occurred at one time. The Wait Seconds High field tells you the longest duration a client has had to wait for a database connection. These attributes allow you to gauge the effectiveness of the current configuration is in responding to client requests.

If the Connections High field value is close to the value of the Maximum Capacity field (set on the Configuration Connections tab), you might consider increasing the value of Maximum Capacity (the maximum number of concurrent connections). If the value in the Waiters High field indicates that clients are subject to a long wait for database access, then you might want to increase the size of the pool.

The value in the Shrink Period field is the length of time the JDBC subsystem waits before shrinking the pool from the maximum. When the subsystem shrinks the pool, database connections are destroyed. Creating a database connection consumes resources and time. If your system has intermittent bursts of client requests, a short shrink period might mean that database connections are being recreated continually, which may degrade performance.