

# 5

## Domains and Network Configuration

In WebLogic Server, a **domain** is a group of servers, with a common set of configuration information. Every server must be in a domain, whether it is a standalone server or just one of many in a domain. Every domain has one **administrative server**, which provides access to the console and stores configuration information. If there is only one server in the domain, it is the administrative server.

Since every WebLogic Server is in a domain, and how we configure a particular server often depends on how the domain is configured, we'll start this chapter with a detailed look at domains. For development, we'll usually just have "a domain of one", but even then it may be more convenient to add all the development servers to a single domain, so they can use a common services configuration.

Beyond the basic domain configuration, we'll look at how to configure the network ports and protocols used by WebLogic Server. We'll finish with a detailed look at the configuration options for each server within the domain.

## 5.1 WebLogic Domains

The purpose of a domain is to centralize management tasks for a group of servers, and to share a common service configuration across the servers.

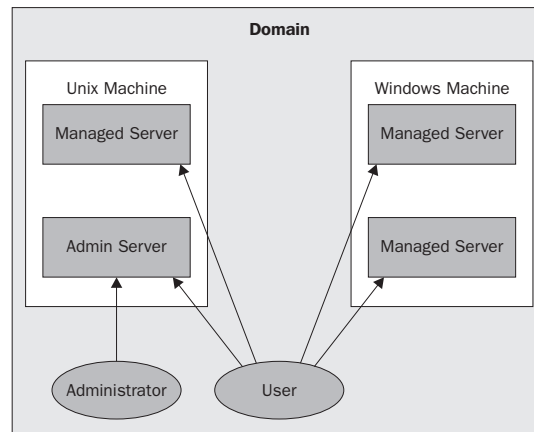
### 5.1.1 Server Types in a Domain

The first and most important server in a domain is the **admin server**. Any other servers in the domain are **managed servers** that are managed by the admin server. From the console on the admin server, we can control the entire domain. The admin server is not solely administrative; it has all the features of a normal server and administrative capability as well. In the case of a single-server configuration, the admin server is the only server.

The strength of a domain configuration shows when we move to a multiple server environment. There are many reasons we might want to do this. We may have a machine available over the Internet, where we don't want any management features to be exposed. Also we don't want the administrative features to be crippled when the server has a heavy load, so we could simply configure one internal server as the admin server, and then add the external server with all the applications to the domain. Another case might involve several servers for several different departments, or several servers for different developers working on the same project, organized into a single domain for consistency and convenience of management.

We can also group multiple servers into a domain for clustering purposes (see *Section 12.2: Creating a Cluster*). When we create a cluster of servers, we can deploy services and applications to more than one server in the cluster. Individual clients (either applications or users working with web browsers) only connect to the cluster as a whole, instead of individual servers within it. This means that additional servers in the cluster can support additional clients, or provide a backup in case one of the servers in the cluster fails. All the servers in a cluster must belong to the same domain, though we could configure several separate clusters in the same domain (for example, in a two-tier cluster, described in *Section 12.1: Cluster Layout*).

Note that separate servers in a domain do not require separate physical machines. For example, in an eight CPU system, we might run four WebLogic servers, each bound to two CPUs and allocated a fraction of the total system RAM. Additionally, different machines in the domain can run different operating systems, so long as they all run the same version of WebLogic Server, running on the same J2SE version (usually J2SE 1.3.x). The following diagram shows the relationship between a domain, servers, machines, and end users (either users of application clients or users with a web browser accessing web applications on the servers):



In this case, the domain has three managed servers, one on the same machine as the admin server, and two on a different machine. To manage the domain, the administrator needs only to deal with the console running on the admin server. A user can connect to applications running on any of the four servers. In a clustered environment, the servers would usually be effectively indistinguishable to the user, while in a non-clustered environment, each of the four would typically be distinct, running different applications, and configured with different services.

### 5.1.2 Domain Configuration

The configuration information for the domain (the `config.xml` file and `userConfig` directory described in *Section 4.3: Configuration Files*) is stored only in the domain directory on the admin server. When the managed servers start they will load any necessary configuration information from the admin server. However, the admin server needs to be configured to recognize each managed server that will connect to it, or else it won't allow the managed server to connect.

**SSL certificates are an exception to the normal behavior, in that they are not loaded from the admin server, but are instead expected to be on the file system of the managed server. However, we still specify the certificate name and keystore name in the console (on the admin server).**

Since the configuration information is only required for the admin server, a managed server needs only a fairly minimal configuration to start. A managed server on the same machine as the admin server can run out of the same domain directory as the admin server. If we want to start managed servers on different machines, each machine will need its own WebLogic Server installation, and a small startup directory configured for the managed server.

### 5.1.3 Services in a Domain

Most WebLogic Server services, such as database pools, JMS servers, and so on, are defined at the domain level, and then assigned to individual servers or clusters within the domain. Once assigned to servers, the service is typically deployed exactly as configured on each of the selected servers (not split in any way across the group). For example, we could define a database connection pool, with a maximum size of 10 connections, and then choose three of five servers in the domain to deploy it to. That would result in one pool on each of the three servers, or a total of 30 connections, maximum, across the entire domain. Application components deployed on each server would only have access to the pool deployed on their own server.

There are a few exceptions to this behavior: logging, security, and SNMP. Each of these operates, to some degree, over the entire domain, instead of on a server-by-server basis.

#### 5.1.3.1 Logging

There are two levels of logging available, a server log for each server, and a domain log for the entire domain. Using the **Domain Log Filters** in the console, we can specify which events on an individual server should be logged to the domain log, while all log events for a specific server go to the server log. The domain log is saved as a file on the admin server, and can also be viewed through the console (using the **View domain log** link at the bottom of all the configuration screens for the domain itself). Server log configuration is discussed in *Section 6.1: The Server Log*.

#### 5.1.3.2 Security

The entire WebLogic Server domain uses a single default security realm. This means that all the applications deployed on any server in the domain will access the same list of users and groups. Of course, the roles may differ from application to application, therefore, the best way to remove a user from an application is to remove a role from them (or remove them from a group that the role is mapped to) rather than removing their account. Otherwise, we may end up removing them from other applications inadvertently. If different applications absolutely need to load users and groups from different sources, they must be deployed on servers in different domains. (See *Chapter 11: Configuring WebLogic Security*, for more information about configuring the security realm, adding and removing users, and setting roles for users or groups.)

**The `weblogic-application.xml` deployment descriptor includes a tag to specify a security realm for an application, but as of WebLogic Server 7.0 SP1, specifying a realm name in this tag prevents the application from deploying.**

### 5.1.3.3 SNMP

WebLogic Server includes an SNMP agent that allows a monitoring product to monitor a WebLogic realm. The SNMP agent always runs on the admin server and operates at the domain level, though it provides information about the various servers in the domain. We can configure the SNMP characteristics and alerts (or "traps") in the console, under Services | SNMP.

## 5.1.4 Creating Domains

We cannot create a domain from within a running WebLogic Server (which itself is already in a domain). Instead, we need to use administrative tools to create the domain, and then use the console to configure it. To create a new domain, we will use a tool called the Configuration Wizard. When we install WebLogic Server, we are prompted to run the Configuration Wizard to create a domain. Thereafter, we can run it using the `dmwiz` script in `/weblogic7/common/bin` or from the Start menu in Windows (the path for the script may also be `/weblogic700/common/bin`).

The Configuration Wizard lets us create domains based on a number of templates. We'll see a different list of templates, depending on how many products from the WebLogic 7.0 suite are installed. There are four templates available with WebLogic Server:

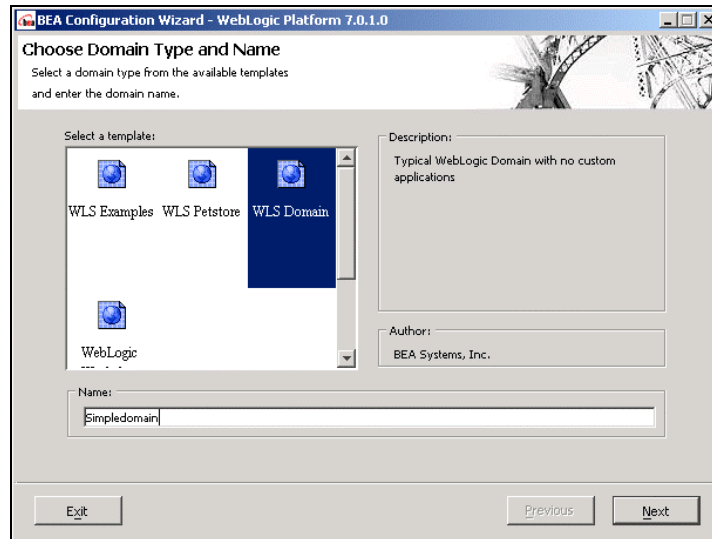
- ❑ **WLS Domain**  
This is an empty domain, and is the configuration we'll generally use, unless we need the specific features offered by one of the other options.
- ❑ **WLS Examples**  
This is a domain with all the examples configured and ready to run, which is helpful when first learning WebLogic Server. However, if we installed the examples while installing WebLogic Server, there is already an Examples domain configured (see *Section 3.1.1: Starting the Examples and Pet Store Domain*).
- ❑ **WLS Pet Store**  
A domain with the J2EE Blueprints "Pet Store" application configured and ready to run. Pet Store is a prototypical J2EE application, demonstrating many of the features of J2EE. Since it is made available by Sun, many application servers support it to demonstrate how each of the corresponding features is implemented. However, if we installed the examples while installing WebLogic Server, there is already a Pet Store domain configured (see *Section 3.1.1: Starting the Examples and Pet Store Domain*).
- ❑ **WebLogic Workshop**  
A domain with the WebLogic Workshop sample project configured and ready to run. Workshop is a developer tool, and the sample project includes code to demonstrate certain features of the tool (see *Section 13.2.3: Web Services Development*).

Each template will result in a slightly different set of applications and services in the initial domain configuration. However, the exact process of configuring a domain depends not on the template, but on whether we want to include managed servers or clusters in the domain. In the following sections, we'll look at each possibility.

### 5.1.4.1 Standalone Server

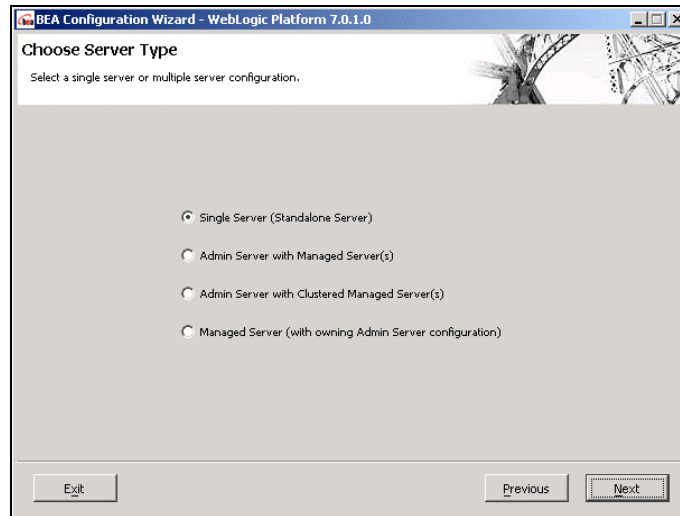
Configuring a standalone server is straightforward; we simply need to enter some information and a domain directory will be created. If the domain is going to include managed servers, it's better to follow one of the other procedures, as it's easier to configure managed servers with the Configuration Wizard.

1. When we start the Configuration Wizard, first we must select a name and template for the domain:



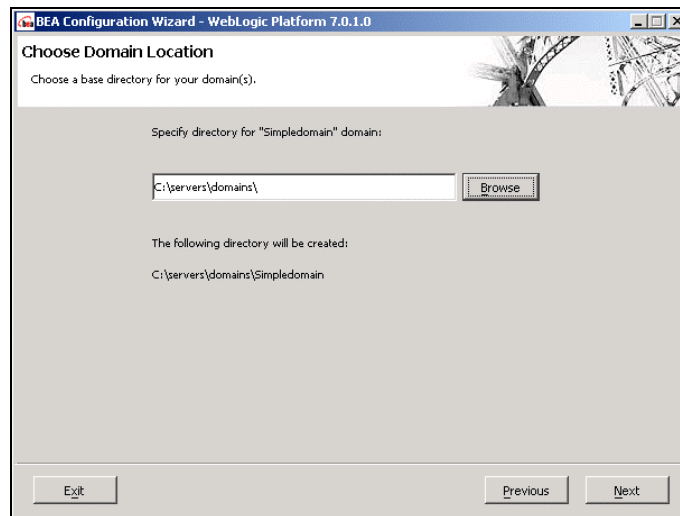
We select the appropriate template for the domain, usually WLS Domain unless we have a good reason to use one of the others. Then we enter a domain name in the Name field. The only restriction is the domain name, which should be unique. (in particular, it should not be the same as a DNS domain name).

2. Next we select the server type:



Here, we select Single Server (Standalone Server).

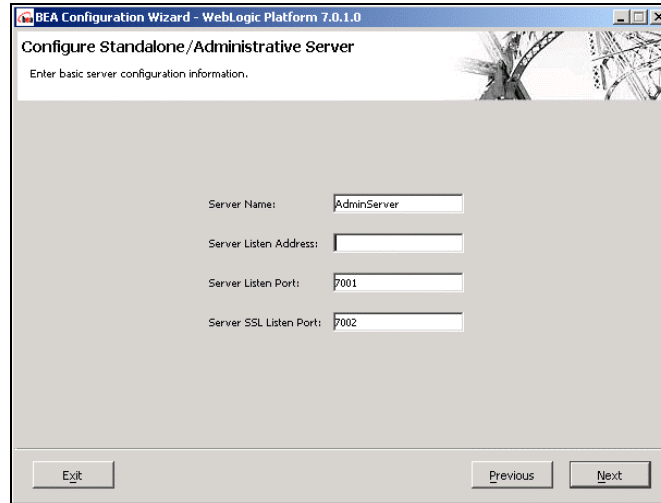
3. Then we select the directory to hold the new domain. This directory can be anywhere; it does not need to be under the WebLogic Server home or BEA home directories.



When we choose a directory here, the domain directory will be created as a subdirectory of it. The domain subdirectory name is the same as the domain name.

*To get a layout like WebLogic Server 6.x, we could create and select the base directory /weblogic7/config.*

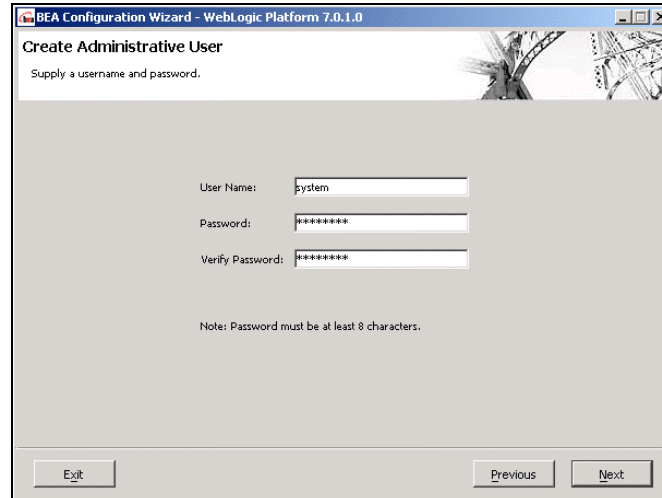
- Now we need to specify the name and basic network configuration for the server:



The screenshot shows the 'Configure Standalone/Administrative Server' window. The title bar reads 'BEA Configuration Wizard - WebLogic Platform 7.0.1.0'. The main heading is 'Configure Standalone/Administrative Server' with the instruction 'Enter basic server configuration information.' Below this, there are four input fields: 'Server Name' (containing 'AdminServer'), 'Server Listen Address' (empty), 'Server Listen Port' (containing '7001'), and 'Server SSL Listen Port' (containing '7002'). At the bottom, there are three buttons: 'Exit', 'Previous', and 'Next'.

The **Server Name** is a unique name, distinguishing this server from the other servers in the domain. It must also be different from the selected domain name. The **Server Listen Address** is the hostname or IP address that this server will listen on. If left blank, this will default to `localhost` and the primary IP address for the current machine. The **Server Listen Port** and **Server SSL Listen Port** are the ports the server will listen on. We don't need to change the defaults unless another server running on the same machine is already using them.

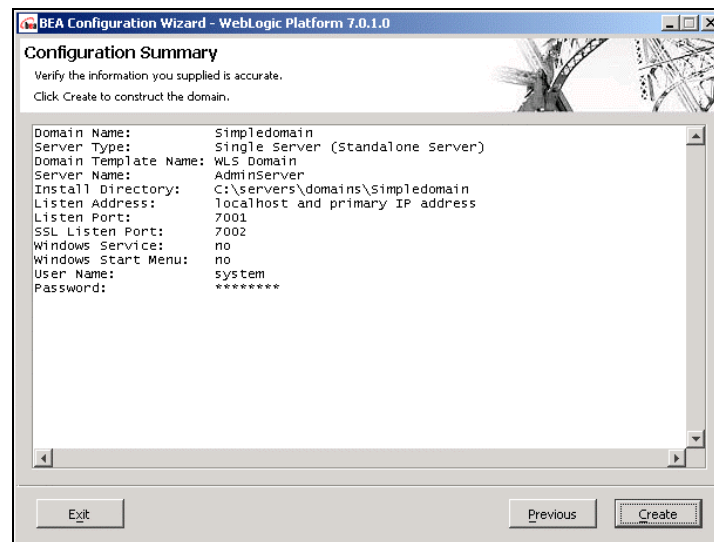
- Next we configure the administrator account:



The screenshot shows the 'Create Administrative User' window. The title bar reads 'BEA Configuration Wizard - WebLogic Platform 7.0.1.0'. The main heading is 'Create Administrative User' with the instruction 'Supply a username and password.' Below this, there are three input fields: 'User Name' (containing 'system'), 'Password' (containing '\*\*\*\*\*'), and 'Verify Password' (containing '\*\*\*\*\*'). A note below the fields states: 'Note: Password must be at least 8 characters.' At the bottom, there are three buttons: 'Exit', 'Previous', and 'Next'.

On this screen, we select a User Name and Password for the user who will start WebLogic Server. This account must be used to start the server, unless we configure a new security realm (described in *Section 11.6: Configuring WebLogic Security Providers*).

6. On Windows platforms, we are prompted to install the server as a Windows service. *Section 3.1.3: Installing as a Windows Service* has more information on installing and uninstalling the service.
7. On Windows platforms, we are prompted to add an entry for the server to the Start menu. This will add a Start menu directory for the domain, with a single shortcut to start the server.
8. Finally, we are given a chance to review the configuration before anything is written to disk:



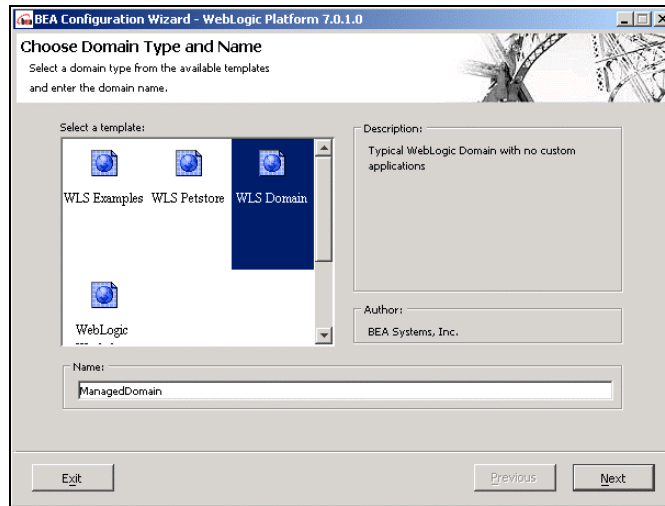
If the information here is correct, we can click **Create** to create the new domain.

9. The final screen simply confirms the creation of the domain, and lets us run the wizard again if necessary.

#### 5.1.4.2 Admin Server with Managed Server(s)

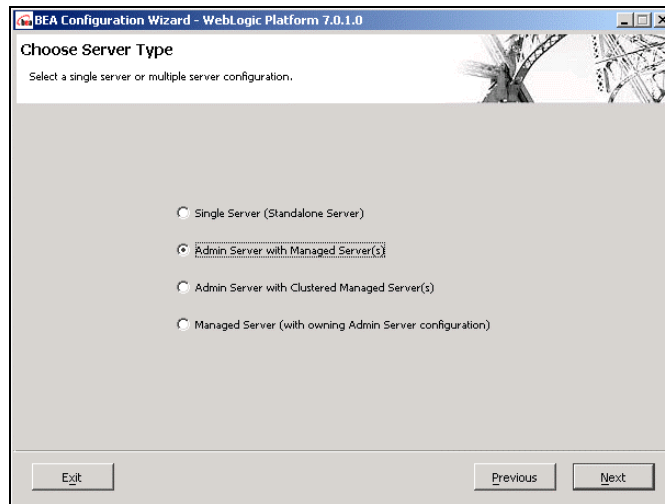
Now, we will create an admin server, and pre-configure the domain with entries for one or more managed servers as well. The wizard will perform all the necessary configuration if the managed servers are going to run on the same machine as the admin servers. However, it can also be used to configure managed servers on different machines; we just need to perform some additional steps to configure the other machines (described below).

1. When we start the Configuration Wizard, we must first select a name and template for the domain:



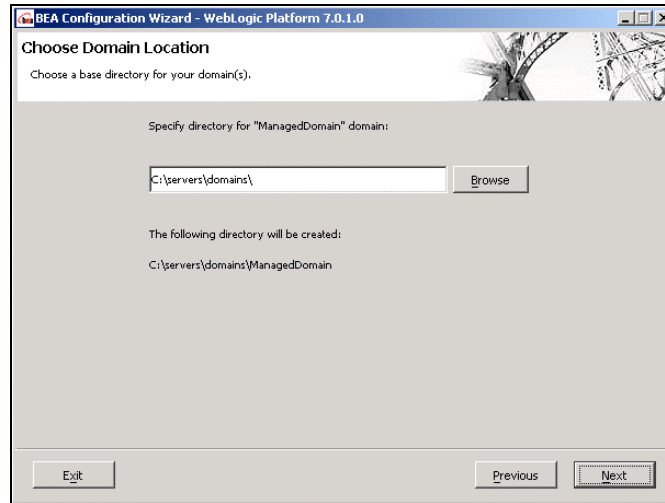
We select the appropriate template for the domain, usually WLS Domain unless we have a good reason to use one of the others. Then we'll enter a domain name in the Name field. The only restriction is on the domain name, which should be unique (in particular, it should not be the same as a DNS domain name).

2. Next we select the server type:



Here we select Admin Server with Managed Server(s).

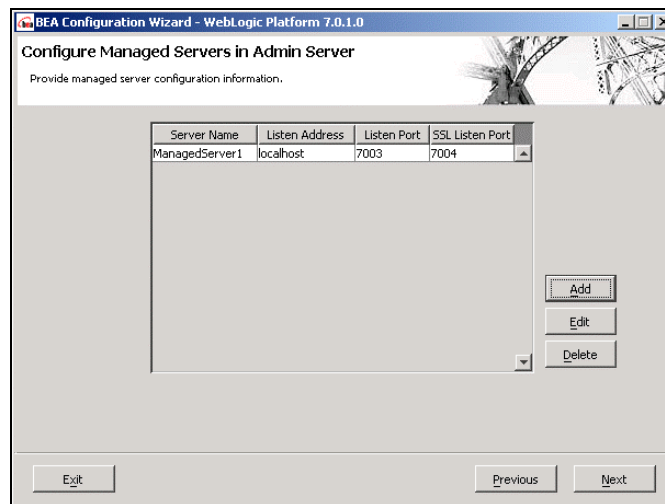
3. We'll select the directory to hold the new domain:



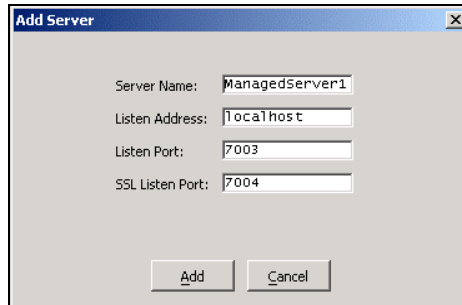
When we choose a directory here, the domain directory will be created as a subdirectory of that. The domain subdirectory name will be the same as the domain name.

*To get a layout like WebLogic Server 6.x, we could create and select the base directory /weblogic7/config.*

4. Now we'll configure the managed servers. Though this screen is empty when it first comes up, the screenshot shows a managed server that is already added:



From here, we can add and edit managed servers for the domain. We must configure at least one. If we click the Add or Edit buttons, we'll get the Managed Server Configuration screen. This screenshot has the settings used to create the entry for the managed server shown above:



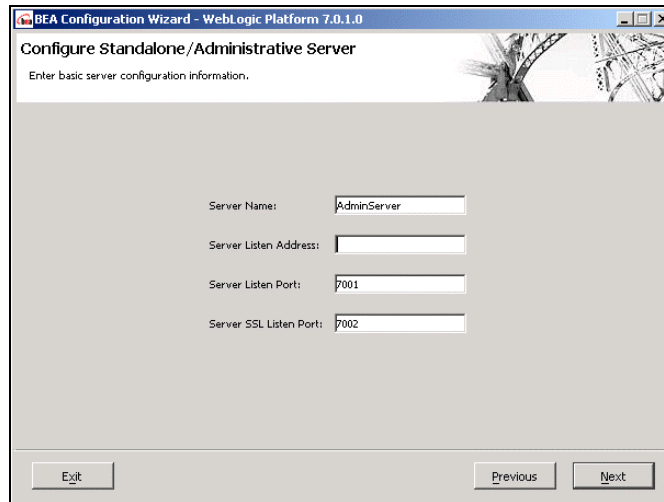
The screenshot shows a dialog box titled "Add Server". It contains the following fields and values:

Field	Value
Server Name	ManagedServer1
Listen Address	localhost
Listen Port	7003
SSL Listen Port	7004

Buttons: Add, Cancel

Here, we can select a **Server Name** for the managed server (which must be different from the name of the domain itself and the names for any other servers in the domain). We also select the **Listen Address**, which is the TCP/IP address that the managed server should listen on (either the host name or IP address). Finally, we choose the **Listen Port** and the **SSL Listen Port**, which are the ports the managed server will listen on. Different servers using the same listen address can't use the same ports, so if the admin server is going to use the standard ports of 7001 and 7002, each managed server will have to select different ports.

5. Now we need to specify the name and basic network configuration for the admin server:



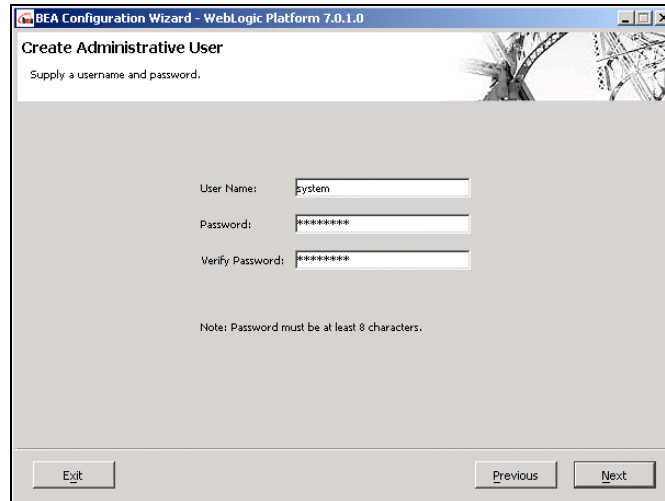
The screenshot shows a dialog box titled "BEA Configuration Wizard - WebLogic Platform 7.0.1.0". The main title is "Configure Standalone/Administrative Server" and the subtitle is "Enter basic server configuration information.". It contains the following fields and values:

Field	Value
Server Name	AdminServer
Server Listen Address	
Server Listen Port	7001
Server SSL Listen Port	7002

Buttons: Exit, Previous, Next

The Server Name is a unique name, distinguishing the admin server from the other servers in the domain. It must also be different from the name of the domain itself. The Server Listen Address is the hostname or IP address that this server will listen on. If left blank, this will default to localhost as well as the primary IP address for the current machine. The Server Listen Port and Server SSL Listen Port are the ports the server will listen on. Again, different servers using the same listen address can't use the same ports.

6. Next we configure the administrator account:



BEA Configuration Wizard - WebLogic Platform 7.0.1.0

Create Administrative User

Supply a username and password.

User Name:

Password:

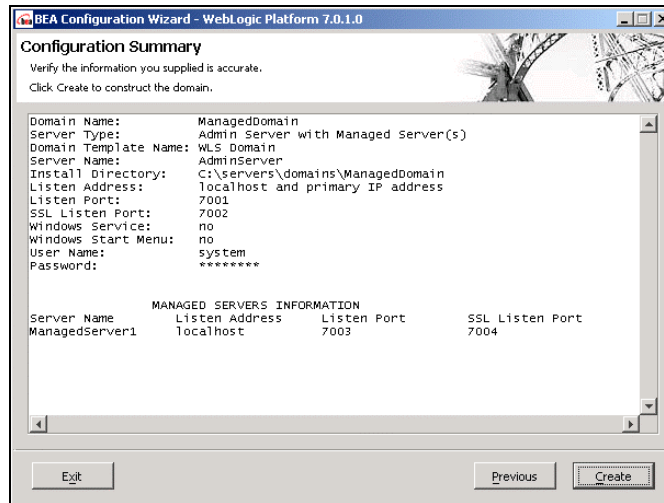
Verify Password:

Note: Password must be at least 8 characters.

Exit Previous Next

Here, we select a User Name and Password for the user who will start WebLogic Server. This account must be used to start the server, unless we configure a new security realm (described in *Section 11.6: Configuring WebLogic Security Providers*).

7. On Windows platforms, we are prompted to install the server as a Windows service. This will only install a service for the admin server, not any managed servers we created with the configuration wizard. *Section 3.1.3: Installing as a Windows Service* has more information on installing and uninstalling the service.
8. On Windows platforms, we are prompted to add an entry for the server to the Start menu. This will add a Start menu directory for the domain, with a single shortcut to start the admin server. No shortcuts are installed for managed servers.
9. Finally, we are given a chance to review the configuration before anything is written to disk:



If the information here is correct, we can click **Create** to create the new domain and managed servers.

10. The final screen simply confirms the creation of the domain, and lets us run the wizard again if necessary.

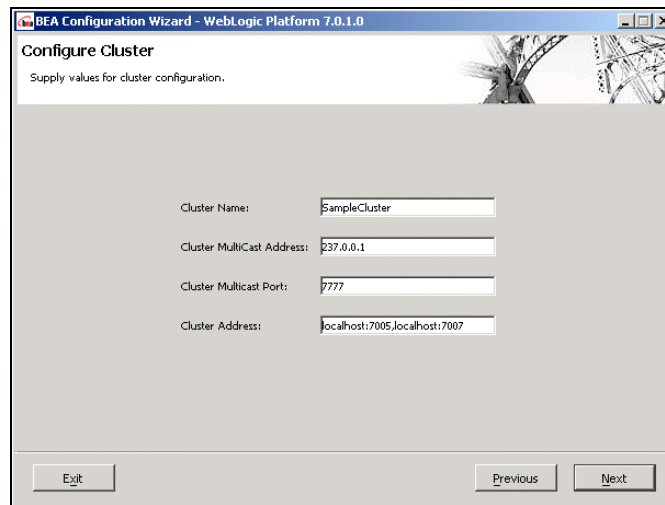
After the wizard has finished, the admin server and any managed servers on the same machine are ready to go. We need to prepare any managed servers that are going to run on different machines.

To prepare the target machine, we use the same procedure described in *Section 5.1.4.4: Adding a Managed Server* to prepare a managed server. However, the configuration wizard has already added the managed server to the domain, so we can skip Steps 1 and 2, and proceed to configure the machine the managed server is going to run on (Step 3).

**! In general, a separate WebLogic Server license is not required for the admin server if it will be strictly administrative, however you should always confirm this with your BEA sales representative.**

### 5.1.4.3 Admin Server with Clustered Managed Server(s)

The procedure for creating a cluster is the same as for creating a domain with managed servers (see *Section 5.1.4.2: Admin Server with Managed Server(s)*), with two exceptions. First, we select Admin Server with Clustered Managed Server(s) on the server type screen, and second, we will get one additional screen with configuration information for the cluster:



BEA Configuration Wizard - WebLogic Platform 7.0.1.0

**Configure Cluster**  
Supply values for cluster configuration.

Cluster Name:

Cluster MultiCast Address:

Cluster Multicast Port:

Cluster Address:

Exit Previous Next

Here, we select a Cluster Name for the cluster, which must be unique in a given domain. We also need to specify a Cluster Multicast Address, which is a valid TCP/IP multicast address (starting with 237. or 238. or 239.). Then we select the Cluster Multicast Port, which is the TCP/IP port that the multicasts will be sent to. Finally, we'll select the Cluster Address, which clients will use to communicate with the cluster.

In a production environment, we'd typically assign a DNS name that could be resolved to any machine in the cluster. Assuming all the servers in the cluster use the same listen port, we could specify `clustername:port` as the cluster address. If we do not have an appropriate DNS name set up, or the servers are listening on different ports, we can set the cluster address to a comma-delimited list of server and port pairs, such as `server1:port,server2:port,server3:port`, and so on.

#### 5.1.4.4 Adding a Managed Server

The Configuration Wizard includes an option to add a managed server only, for use with an existing domain.

**However, we should avoid using this option, as it doesn't really work as expected. It creates a whole new domain and doesn't add a record for the new managed server to the existing domain, even though it configures the managed server to contact the existing domain (not the new domain!).**

We would spend more time trying to clean that up than we would just by adding the managed server without the wizard.

If we haven't yet created the initial domain, it's the best to create the domain and all the managed servers at the same time using the procedure from either *Section 5.1.4.2: Admin Server with Managed Server(s)* or *Section 5.1.4.3: Admin Server with Clustered Managed Server(s)*.

If the domain is already up and running, we have no choice but to add the managed server definition through the console, and then copy the necessary startup scripts over to the machine where the managed server will run.

1. We start in the console for the domain (which always runs on the admin server) and select the Servers entry in the navigation applet. Click **Configure a new Server...** in the content pane to bring up the new server screen:

The screenshot shows the 'Configure a new Server' dialog box in the Oracle WebLogic Server console. The 'General' tab is selected, and the following fields are visible:

- Name:** ManagedServer1
- Machine:** (none)
- Cluster:** (none)
- Listen Address:** (empty)
- Listen Port Enabled:**
- Listen Port:** 7001
- WebLogic Plug-In Enabled:**
- Startup Mode:** RUNNING
- External DNSName:** (empty)

A 'Create' button is located at the bottom right of the dialog box.

2. Here, we need to specify the Name for the new managed server. This must be different from the names for the other servers in the domain. We also need to select a Listen Port for the managed server, which does not conflict with any other servers on the same machine (the default port, 7001, will often be in use if this managed server is going to run on the same machine as the admin server). If the managed server is going to run on a different machine, we'll specify the Listen Address, which is the TCP/IP address the server will bind to. If we have already defined a Machine for the physical machine the server will run on, we can select it here; otherwise, we'll take care of that later.

We can configure other server settings now (see *Section 5.3: Detailed Server Configuration*), including activating SSL on the **Connections | SSL Ports** tab (see *Section 11.7: SSL Configuration* for SSL configuration information). However, note that SSL won't start unless we configure a certificate and keystore for the managed server.

3. Once the new server entry is ready in the console, we need to prepare the directory it will run from. If it's going to run on the same machine as the admin server, we can run it out of the same directory as the domain. We'll just use the `startManagedWebLogic` script in the domain directory, and pass it the managed server name and admin server URL to use.

```
./startManagedWebLogic.sh ManagedServer1
                           http://adminHost:7001/
```

If the managed server is to be run on a different machine, we need to make sure that WebLogic Server is installed on the target machine, though we don't need to create any domains for it at installation time. We need to create a directory on that machine for the managed server (the location of the directory is not important). Then we should copy the `startManagedWebLogic.sh` script (for a UNIX target machine) or the `startManagedWebLogic.cmd` script (for a Windows target machine) from the admin server directory to the managed server directory we created on the target machine. We'll need to update the script to reflect the actual location where WebLogic Server is installed on that machine, as well as specifying the correct `SERVER_NAME` (the name of the managed server to start) and `ADMIN_URL` (the HTTP address of the admin server).

```
SERVER_NAME=ManagedServer1
ADMIN_URL=http://adminHost:7001/
...
. "/directory/on/new/server/weblogic7/server/bin/startWLS.sh"
```

If we are going to run SSL on the managed server, we will need to create a keystore and certificate for it, and add those to the directory for the managed server (see *Section 11.7.3: Configuring SSL for a Server*). The "demo" certificate created for the admin server won't work if the managed server is on a different machine, as it will have the hostname or IP address for the admin server, not for the managed server.

The necessary subdirectories will be created the first time the managed server is started. None of the rest of the files in the typical domain directory is necessary, as the admin server will handle all that.

4. To start the managed server, we'll run the `startManagedWebLogic` script. If we haven't hardcoded the `SERVER_NAME` and `ADMIN_URL`, we'll pass them as arguments on the command line. If we haven't hardcoded the admin username and password for the domain (using the `WLS_USER` and `WLS_PW` variables in the script), we'll be prompted to give them when the server starts. After the password prompt, we should see output like `Connecting to (admin URL)...` indicating that the managed server is contacting the admin server:

```

Konsole - root@localhost:server/TestDomain - Konsole
File Sessions Settings Help

[root@localhost TestDomain]# ./startManagedWebLogic.sh ManagedServer1 http://loc
alhost:7001/
LD_LIBRARY_PATH=/root/.bea/weblogic700/server/lib/linux/i686:/root/.bea/weblogic70
0/server/lib/linux/i686/oc1817_8
CLASSPATH=/root/.bea/jdk131_03/lib/tools.jar:/root/.bea/weblogic700/server:/root/.b
ea/weblogic700/server/lib/weblogic_sp.jar:/root/.bea/weblogic700/server/lib/weblo
gic.jar:
PATH=.:/root/.bea/weblogic700/server/bin:/root/.bea/jdk131_03/jre/bin:/root/.bea/jd
k131_03/bin:/root/.bea/weblogic700/server/lib/linux:/usr/local/sbin:/sbin:/usr/sb
in:/bin:/usr/bin:/usr/bin/X11:/usr/local/bin:/usr/bin:/usr/X11R6/bin:/root/bin:/
root/bin
*****
* To start WebLogic Server, use a username and *
* password assigned to an admin-level user. For *
* server administration, use the WebLogic Server *
* console at http://<hostname>:<port>/console *
*****
** /root/.bea/jdk131_03/bin/java -Xms32m -Xmx200m -Dweblogic.security.SSL.trusted
CAKeyStore=/root/.bea/weblogic700/server/lib/cacerts -classpath /root/.bea/jdk131_
03/lib/tools.jar:/root/.bea/weblogic700/server:/root/.bea/weblogic700/server/lib/w
eblogic_sp.jar:/root/.bea/weblogic700/server/lib/weblogic.jar: -Dweblogic.Name=Ma
nagedServer1 -Dbea.home=/root/.bea -Dweblogic.management.username= -Dweblogic.man
agement.password= -Dweblogic.management.server=http://localhost:7001/ -Dweblogic
.ProductionModeEnabled= -Djava.security.policy=/root/.bea/weblogic700/server/lib/
weblogic.policy weblogic.Server
<Nov 26, 2002 3:53:05 PM IST> <Info> <Security> <090065> <Getting boot identity
from user.>
Enter username to boot WebLogic server:system
Enter password to boot WebLogic server:
Starting WebLogic Server...

```

## 5.2 TCP/IP Ports and Protocols

WebLogic Server uses two ports by default, one for SSL communication, and one for all other unencrypted traffic (clusters use an additional multicast port, described in *Section 12.2: Creating a Cluster*). The default listen port is 7001, and the default SSL listen port is 7002. If the server is behind a firewall, we'll need to allow one or both of these through the firewall, depending on what kind of traffic we want to support.

WebLogic Server runs a number of protocols over those ports, primarily:

- ❑ **T3 and T3S**  
WebLogic's proprietary T3 protocol is the default protocol, used primarily for JNDI and connections to components such as EJBs. T3S is the secure (SSL) version of the T3 protocol.
- ❑ **HTTP and HTTPS**  
HTTP is the standard WWW protocol, used by web and web services clients. HTTPS is the secure (SSL) version of the HTTP protocol. For applets and other cases where communication is limited to HTTP traffic by a firewall, T3 can be tunneled over an HTTP connection (see *Section 20.3.2: Configuring HTTP Tunneling*).
- ❑ **IIOP and IIOPS**  
IIOP is the standard protocol for CORBA, and will be used by CORBA clients attempting to contact EJB running in the server. IIOPS refers to IIOP running over SSL.

By default, T3, HTTP, and IIOP will be supported over the listen port 7001, and T3S, HTTPS, and IIOPS over the SSL listen port 7002. By default WebLogic listens on the primary IP address for the machine as well as the localhost interface. We can change the default listen address and ports for a server in the console. We select the server under Servers in the navigation applet, and then go to Configuration | General to change the listen address and listen port, or Connections | SSL Ports to change the SSL listen port (refer to the screenshot in *Section 5.3.1: Configuration | General*). If the listen address is left blank, that indicates the default behavior.

### 5.2.1 Separating Administration Traffic

Sometimes it is useful to separate administration traffic from application traffic. This can be used as a security measure to hide the console from normal clients. Additionally, providing a dedicated port for administrative traffic will result in faster processing of admin commands, since the administrative traffic is not waiting in line with normal application traffic.

If we want to separate admin traffic from normal traffic, we can configure an administration port for the domain. This is one port number, used by every server in the domain, which all administration traffic must use. When this is enabled no other ports will accept traffic for the console, start managed servers, etc. The administration port supports SSL traffic only.

*If a managed server can't bind to its admin port during startup, it can fall back to its SSL listen port to communicate with the admin server. However, this should only be used long enough to reset the managed server's admin port to a valid value.*

To enable the administration port, all servers in the domain must support SSL. This means we must have proper certificates configured (see *Section 11.7.3: Configuring SSL for a Server*), in particular, on managed servers on different machines from the admin server (which don't have a valid SSL configuration by default).

To enable the administration port, we'll select the entry for the domain in the navigation applet, and go to the Configuration | General tab in the content pane:

Configuration | Security | Monitoring | Notes

General | JTA | SNMP | Logging | Applications

⚠️ ? Name: TestDomain

⚠️ ?  Console Enabled

⚠️ ? Console Context Path: console

⚠️ ?  Enable Domain Wide Administration Port (Please configure SSL)

⚠️ ? Domain Wide Administration Port: 9002

Apply

We select the port in the Domain Wide Administration Port field, and then check the box for Enable Domain Wide Administration Port.

Once enabled, we need to use a URL such as `https://adminhost:9002/console/` to access the console, and a URL such as `https://adminhost:9002/` as the `ADMIN_URL` to start any managed servers for the domain.

Every server in the domain listens on the port specified as the administration port. However, each server can override the default admin port number to listen for admin traffic on a different port. This might be necessary if another process has already claimed the default port on a particular machine. To change the admin port for a server, we select the server in the navigation applet, and then go to the Connections | SSL Ports tab:

Configuration | Connections | Monitoring | Control | Logging | Deployments | Services | Notes

SSL | SSL Ports | HTTP | jCOM | Tuning | Protocols | Summary

Listen Address:

⚠️ ?  Enable SSL Listen Port (Please configure SSL)

⚠️ ? SSL Listen Port: 7002

Enable Domain Wide Administration Port: false

Domain Wide Administration Port: 9002

⚠️ ? Local Administration Port Override (0: no override): 0

[Channel Overrides](#)

Apply

If we set the Local Administration Port Override to a nonzero value, the server will use the specified port as its admin port.

## 5.2.2 Using Network Channels

If we want to separate different protocols onto different ports, or make different network service levels available to different clients, we can configure network channels. This might be used to provide more network resources to higher-priority clients or applications, or to restrict a public HTTP port to only HTTP traffic for security reasons (see *Section 5.2.5: Network Configuration Examples*). It won't affect the default port configuration discussed in the previous section, but will add new ports in addition to the default ports. To add a network channel, select Network Channels in the navigation applet, and then click Configure a new Network Channel... in the content pane. This brings up the network channel configuration screen:

The screenshot shows a configuration window with three tabs: Configuration, Targets, and Notes. The 'Configuration' tab is active and contains three sub-tabs: General, Tuning, and Protocols. The 'General' sub-tab is selected. The form includes the following fields and options:

- Name:** MyNetwork Channel
- Description:** (empty text box)
- Listen Port Enabled:**
- Listen Port:** 8001
- SSL Listen Port Enabled:**
- SSL Listen Port:** 8002
- Cluster Address:** (empty text box)
- T3 Enabled:**
- T3S Enabled:**
- HTTP Enabled:**
- HTTPS Enabled:**
- Tunneling Enabled:**
- COM Enabled:**

A 'Create' button is located at the bottom right of the form.

In addition to selecting a unique Name for the channel, we can enable or disable a Listen Port and SSL Listen Port for this channel. Again, these are additional ports beyond the defaults configured for the server. We can also enable or disable a variety of protocols for this channel. We cannot enable IOP, which is only supported on the "default" channels (that is, the main configuration for a server, exclusive of any configured network channels).

**Enabling T3, HTTP, Tunneling, or COM without selecting a listen port will not actually enable those protocols for this channel. Likewise, selecting T3S or HTTPS without an SSL listen port will not actually enable those protocols, even though their checkboxes are checked.**

The Configuration | Tuning and Configuration | Protocols tabs offer some advanced options for configuring the network and protocol behavior for this channel:

The screenshot shows a configuration window with tabs for 'Configuration', 'Targets', and 'Notes'. Under 'Configuration', there are sub-tabs for 'General', 'Tuning', and 'Protocols'. The 'Tuning' tab is active, displaying the following settings:

- Accept Backlog:** 50
- Login Timeout:** 5000 ms
- SSL Login Timeout:** 25000 ms
- Channel Weight:** 50
- Outgoing Enabled:**

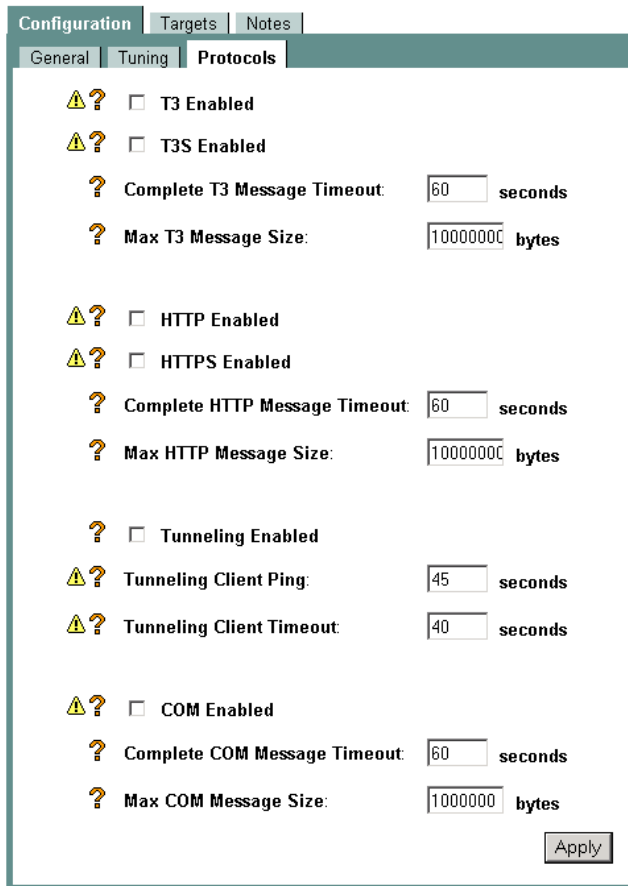
An 'Apply' button is located at the bottom right of the configuration area.

The options on this screen are:

Setting	Description
Accept Backlog	The number of backlogged TCP connection requests allowed, for both plain and SSL connections. It's best to set this greater than zero (due to operating-system constraints).
Login Timeout	The amount of time allowed for a pending HTTP connection to be established. If this expires, the connection is dropped.
SSL Login Timeout	The amount of time allowed for a pending secure connection to be established. Secure connections generally take longer than plain connections. If this expires, the connection is dropped.
Outgoing Enabled	If selected, outgoing server-to-server communication may use this channel, according to its weight. This might be used to communicate between servers, within a cluster, or between managed server and an admin server.

Setting	Description
Channel Weight	When the server needs to connect to another server, it selects the channel that supports the relevant protocol, has outgoing traffic enabled, and has the highest weight compared to other eligible channels. This can be used, for example, to force server-to-server traffic onto a faster or dedicated Network Interface Card (NIC) in the server. The ports available to the server by default (that is, ignoring any network channels) have weight 50.

The other tab with network settings is:



The Protocol tab repeats the checkboxes to enable all the allowed protocols from the general tab. The additional options are:

Setting	Description
Max Protocol Message Size	The maximum size of a message header for the protocol. This is used to prevent denial of service attacks using gigantic headers, which may cause memory problems with the server.
Complete Protocol Message Timeout	The maximum time to spend reading a message for the protocol. This is used to prevent denial of service attacks where the header specifies a certain message size, but the specified data never arrives, hanging the reader thread.
Tunneling Client Ping, Tunneling Client Timeout	Settings to tunnel other protocols over HTTP. Discussed in <i>Section 20.3.2.2: Client Tunneling Configuration</i> .

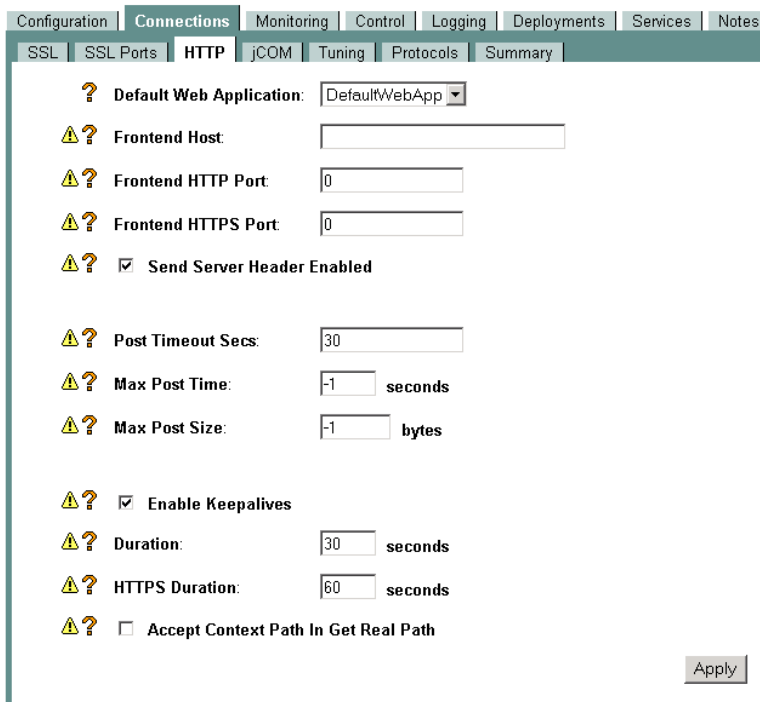
As with most services in WebLogic Server, we must select a target on the **Targets | Servers** tab to activate the network channel. We select the server on the **Available** side, click the right arrow to move it to the **Chosen** side, and then click **Apply** to start the network channel on the selected application server.

Each server using a network channel can override the settings for the channel. We can access the overrides using the **Channel Overrides** link on the **Connections | Tuning** tab for the server. Only the channels targeted to the server are available there for overriding.

*We can bind a network channel to a particular network interface in a server, using the Channel Overrides. The Configuration | General tab for the channel override lets us specify a Listen Address for the channel, which effectively restricts the channel to a single NIC. This is how we separate traffic onto different interfaces in a multihomed system.*

### 5.2.3 Configuring the HTTP Protocol

The bulk of the HTTP protocol configuration is done on the **Connections | HTTP** tab for each server in the console:



The options on this screen are:

Setting	Description
Default Web Application	To invoke a web application, the URL path information typically needs to begin with the context of the web application. This is not the case for the default web application; it will be invoked whenever no context path is provided.
Frontend Host	Used if all HTTP or HTTPS traffic must be funneled through an external host. This is similar to the External DNS Name (see <i>Section 5.3.1: Configuration   General</i> ) but is a completely different machine, with different ports from the WebLogic server.
Frontend HTTP Port	The port to use for HTTP traffic on the front-end host.
Frontend HTTPS Port	The port to use for HTTPS traffic on the front-end host.
Send Server Header Enabled	Whether the server name should be sent with the HTTP response. Can be disabled for wireless applications where header space must be minimized.

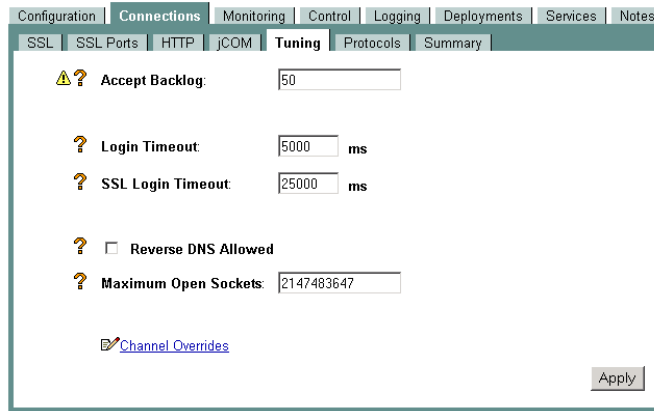
*Table continued on following page*

Setting	Description
Post Timeout Secs	In a denial of service attack, an attacker may initiate an HTTP POST, but then neither send more data nor close the connection. If this timeout expires while WebLogic Server is waiting for more data to be posted, an error will be returned to the client.
Max Post Size	The maximum allowable data sent in an HTTP POST. If the data posted exceeds this size, an error will be returned to the client.
Max Post Time	The maximum amount of time for which WebLogic Server will load data for a single HTTP POST request. If data is still coming after this much time, an error will be returned to the client.
Enable Keepalives	Whether HTTP keep-alive should be enabled. This means the network socket between a client and the server may not be closed after every HTTP operation, but instead left open for further requests. It makes many requests by the same client more efficient in terms of opening and closing sockets, but will result in more simultaneous open sockets for the server to manage.
Duration	If keep-alives are enabled, the amount of time a client can be idle before the connection is closed.
HTTPS Duration	The amount of time a HTTPS client can be idle before the connection is closed.
Accept Context Path In Get Real Path	Starting with this release, if a web component calls <code>ServletContext.getRealPath()</code> , then the argument path may not start with the context name. If it does, it is assumed as a subdirectory with the same name as the context. Enable this option to revert the behavior, to assume a context path prefix is, in fact, the context path and not a subdirectory.

Beyond the settings here, the port used for HTTP traffic is controlled by the listen port configured for the server (see *Section 5.2: TCP/IP Ports and Protocols*), and any network channels configured for the server (see *Section 5.2.2: Using Network Channels*).

## 5.2.4 Network Tuning

Network tuning settings are found on the Connections | Tuning and Connections | Protocols tabs for each server in the console. Tuning settings control both performance and security aspects of the network protocols:



The options on this screen are:

Setting	Description
Accept Backlog	The number of backlogged TCP connection requests allowed, for both plain and SSL connections. It's best to set this above zero (due to operating system constraints).
Login Timeout	The amount of time allowed for a pending HTTP connection to be established. If this expires, the connection is dropped.
SSL Login Timeout	The amount of time allowed for a pending secure connection to be established. Secure connections generally take longer than plain connections. If this expires, the connection is dropped.
Reverse DNS Allowed	If enabled, WebLogic Server will attempt reverse-DNS lookups when clients connect. While superior for logging and security purposes, it can have a significant impact on performance (and requires that reverse DNS is configured correctly for all clients), so it is disabled by default.
Maximum Open Sockets	The limit to the number of open network sockets in the system. After this limit is reached, additional connection attempts will be rejected until some pending requests are completed. This is the same as the setting on the Configuration   Tuning tab.

The Channel Overrides link on this screen lets us override the settings for any network channels targeted to this server. The server can override most of the configuration values from the defaults specified for the channel.

The other tab with network settings is:

The screenshot shows the 'Protocols' configuration page with the following settings:

- 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 IOP:**
- IOP Max Message Size:** 10000000 bytes
- IOP Message Timeout:** 60 seconds
- Default IOPPassword:** [Change...](#)
- Default IOPUser:**
- Enable COM:**
- COM Max Message Size:** 10000000 bytes
- COM Message Timeout:** 60 seconds

At the bottom left, there is a checkbox for [Channel Overrides](#). At the bottom right, there is an **Apply** button.

The options on this screen are:

Setting	Description
Default Protocol	Specifies the default protocol, for traffic on the listen port, which is used when the protocol is not clear from the incoming data.
Default Secure Protocol	Specifies the default protocol, for traffic on the SSL listen port, which is used when the protocol is not clear from the incoming data.
<i>Protocol</i> Max Message Size	The maximum size of a message header for the protocol. This is used to prevent denial of service attacks using gigantic headers, which may cause memory problems with the server.

Setting	Description
Protocol Message Timeout	The maximum time to spend reading a message for the protocol. This is used to prevent denial of service attacks where the header specifies a certain message size, but the specified data never arrives, hanging the reader thread.
Enable Tunneling, Tunneling Client Ping, Tunneling Client Timeout	Settings to tunnel other protocols over HTTP. Discussed in <i>Section 20.3.2.2: Client Tunneling Configuration</i> .
Enable IIOP	Whether the IIOP protocol should be enabled on the main listen ports. It may be enabled on different ports using Network Channels.
Default IIOPUser	The user to use for IIOP requests that are not otherwise authenticated.
Default IIOPPassword	The password to use for the default IIOP user.
Enable COM	Whether the COM protocol should be enabled on the main listen port. It may be enabled on different ports using Network Channels.

## 5.2.5 Network Configuration Examples

### 5.2.5.1 Quality-Of-Service Configuration Example

Suppose we have two applications running on a WebLogic Server, a high-priority accounting application, and a lower-priority inventory system. Both are web applications.

We create a network channel for each application. We select a different listen port for each one, and enable the HTTP protocol only. We tune each channel separately: The accounting application uses default settings, except we uncheck **Outgoing Enabled** to keep extra traffic off this channel. The inventory channel uses a lower **Accept Backlog** of 5, since we'd rather reject extra connections until the server can handle them. The login timeouts are reduced for the same reason. It's harder to connect to the inventory application when the server is heavily loaded, but that's OK as we'd rather spend the server power on the accounting application anyway.

When we give a URL to clients, we give them a different port depending on which application they are connecting to. Though in theory, each application is available on each port, this will split clients for the two applications onto different channels by default.

If the applications used EJBs, we would enable the T3 protocol, and the clients would use different ports in their JNDI connection settings, to connect via the separate channels.

### 5.2.5.2 Security Configuration Example

Assume we have a high-profile Internet application running on the server. It is a web application, using HTTP and HTTPS. We are (perhaps foolishly) using a single server for the domain. It is a multi-homed machine, with one internal NIC and one external NIC.

We specify an administration port for the domain on port 7003, enable SSL on port 7002, and disable the listen port (7001). We specify a listen address with the IP address of the internal NIC. Inbound traffic from the Internet cannot reach these ports, because we are not listening on the external NIC. However, we block the ports on the firewall just the same.

We configure a network channel, with a listen port of 80 and a SSL listen port of 443, supporting the HTTP and HTTPS protocols only. We target the listen port to our server. Then we go into the network channel overrides for the server, and specify a listen address with the IP address of the external NIC. This means Internet users can contact the application over the standard HTTP and HTTPS ports. Internal users cannot access the application, except through the same network interface as external users. We only allow ports 80 and 443 through our firewall.

If we were smarter, we might configure a domain with two servers. The external server would be a managed server with the Internet connection and the application. The admin server would be internal only. It would not run any applications, to keep it maximally responsive. We could still lock down the internal interface of the managed server, allowing SSL and admin port traffic only, and perhaps using a firewall that only allows it to connect to the admin server.

## 5.3 Detailed Server Configuration

The entry in the console for an individual server has numerous configuration screens. Few of the settings here need to be adjusted from a default configuration; the rest are mentioned elsewhere as they are required. This section reviews many of the optional configuration screens available for each server in the domain.

### 5.3.1 Configuration | General

This screen has the main configuration options for the server:

The screenshot shows the 'General' configuration tab for a WebLogic server. The fields are as follows:

- Name:** AdminServer
- Machine:** (none)
- Cluster:** (none)
- Listen Address:** (empty text box)
- Listen Port Enabled:**
- Listen Port:** 7001
- WebLogic Plug-In Enabled:**
- Startup Mode:** RUNNING
- External DNSName:** (empty text box)

An 'Apply' button is located at the bottom right of the configuration window.

The options on this screen are:

Setting	Description
Machine	Typically set when the server is part of a cluster, and the cluster includes multiple servers on the same physical machine. See <i>Section 12.3.2: Machines and Replication Groups</i> .
Cluster	Selects a cluster for this server to participate in. Each cluster should include two or more servers, which may or may not include the admin server for the domain as well as a number of managed servers. (It is usually the best to keep the admin server out of the cluster, for performance reasons.)
Listen Address, Listen Port	Network configuration, described in <i>Section 5.2: TCP/IP Ports and Protocols</i> .
WebLogic Plug-In Enabled	Activates extra measures in identifying the client of a web request. This only works if we're using a web server for static content with the WebLogic plug-in to handle dynamic content. Can also be set for a cluster instead of a single server (see <i>Section 12.2: Creating a Cluster</i> ).
Startup Mode	Selects whether the server runs, when first started (RUNNING), or goes into standby mode (STANDBY). If in standby mode, we must configure an administration port (see <i>Section 5.2.1: Separating Administration Traffic</i> ), and switch the server to running mode using the command-line admin tool (see <i>Section 3.4.2: The Command-Line Administration Tool</i> ).

*Table continued on following page*

Setting	Description
External DNSName	If the server is listening on an IP address where the internal host name for the server is not visible to clients, this field specifies the host name clients can use to communicate with the server.

### 5.3.2 Configuration | Cluster

This screen has options for servers participating in a cluster. Clusters are described in more detail in *Chapter 12: Configuring WebLogic Clusters*:

The screenshot shows the 'Cluster' configuration page in the WebLogic Configuration console. The 'Cluster' tab is active, and the 'Replication Group' section is expanded. It contains four fields: 'Replication Group', 'Preferred Secondary Group', 'Cluster Weight' (with a value of 100), and 'Interface Address'. An 'Apply' button is located at the bottom right of the configuration area.

The options on this screen are:

Setting	Description
Replication Group, Preferred Secondary Group	Used to configure replication groups. See <i>Section 12.3.2: Machines and Replication Groups</i> .
Cluster Weight	When the cluster uses weighted load balancing (see <i>Section 12.2: Creating a Cluster</i> ), the value specified here controls how much of the cluster load this server will bear. For example, if this server has weight 100, and two other servers in the cluster have weight 200, this server will bear 20% of the load on the cluster ( $100 / (100+200+200)$ ).
Interface Address	If the machine has multiple network interfaces, the IP address specified in this field is used for cluster multicast traffic. See <i>Section 12.2: Creating a Cluster</i> .

### 5.3.3 Configuration | Memory

The settings on this screen are used to tune how WebLogic Server handles low-memory conditions. The default values are usually appropriate, but they can be adjusted to tune the aggressiveness of garbage collection and when reduced memory conditions trigger log messages:

The screenshot shows the 'Memory' configuration tab in the WebLogic Server console. It contains four settings, each with a warning icon and a question mark:

- Low Memory GCThreshold:** 5
- Low Memory Granularity Level:** 5
- Low Memory Sample Size:** 10
- Low Memory Time Interval:** 3600 seconds

An 'Apply' button is located at the bottom right of the configuration area.

The options on this screen are:

Setting	Description
Low Memory GCThreshold	When the available memory reaches this percentage of the initial free memory, WebLogic will automatically run garbage collection. Valid values are 0-9.
Low Memory Time Interval	The period over which memory data is gathered, in seconds. The memory state is averaged over a number of samples in this time period.
Low Memory Sample Size	The number of times memory data is sampled in each low memory time interval.
Low Memory Granularity Level	If the average free memory drops by the percentage specified here, between one time interval and the next, log messages will be written.

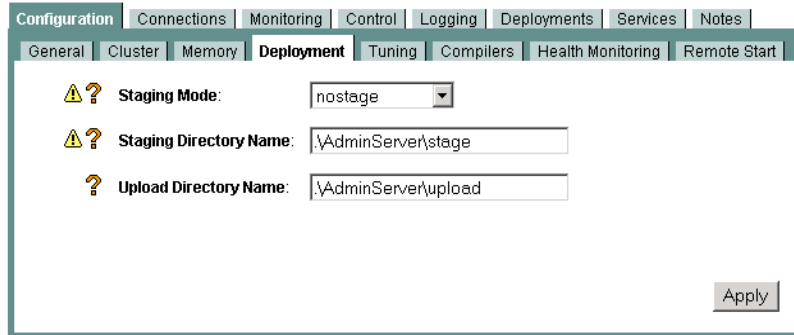
### 5.3.4 Configuration | Deployment

Configures how applications are deployed to the server. Each server has three staging options:

- nostage  
Applications are run from the directory where they are originally located. This is the default for admin servers, and single-server domains.

- ❑ `stage`  
Applications are copied to a staging directory to run. This is the default for managed servers, where the applications are generally copied from the admin server to a local staging directory.
- ❑ `external_stage`  
The system administrator must copy applications to a staging directory on each server to deploy them. This option just causes extra work for the administrator, but could be used when extremely strict security prevents WebLogic Server from writing files to its own application staging directory.

The default settings for each server are typically best. However, if we need to override the staging mode, the options are:

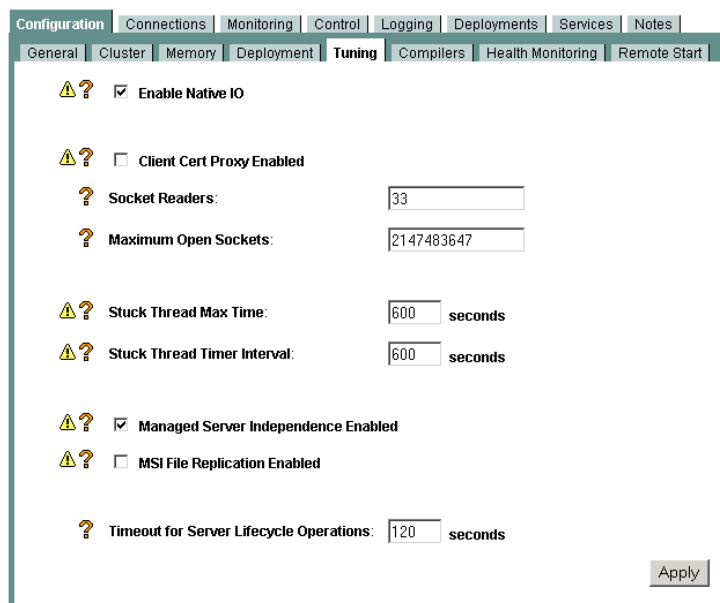


The options on this screen are:

Setting	Description
Staging Mode	Must be one of the staging settings described above.
Staging Directory Name	The staging directory to which applications are copied in stage or external_stage modes. Relative to the startup directory for the server.
Upload Directory Name	Present for admin servers only. This is the directory where uploaded applications are saved. Relative to the domain directory.

### 5.3.5 Configuration | Tuning

This screen has a variety of tuning settings for the server. The default values are acceptable, so we should only change them if we have a good reason to:



The options on this screen are:

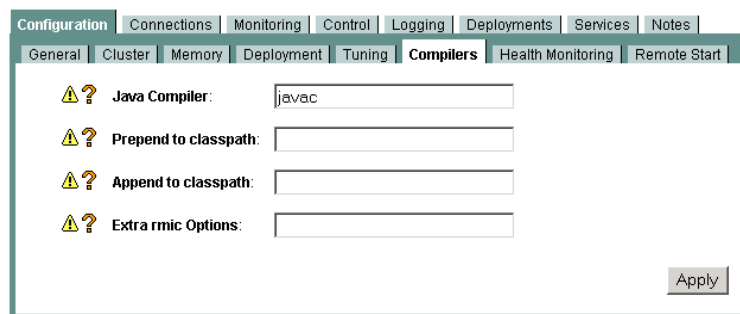
Setting	Description
Enable Native IO	WebLogic Server includes native code networking features, with better performance than the default JDK networking library. If an implementation is available for the current platform, this box defaults to checked. It should not be changed.
Client Cert Proxy Enabled	Enabling this exposes a security hole with regard to SSL client certificates and servlet-based load balancing. It can be enabled for backward compatibility, but should otherwise be left unchecked.
Socket Readers	The threads allocated by WebLogic Server are split into reader threads (which read data from the network) and execute threads (which process requests). This setting indicates the percentage of reader threads vs. execute threads, for all threads. The ratio can be adjusted to favor execute threads if too few concurrent requests are being processed, or reader threads if the execute threads are waiting for work. An optimal setting would match the number of reader threads to the number of open sockets.

*Table continued on following page*

Setting	Description
Maximum Open Sockets	The limit to the number of open network sockets in the system. After this limit is reached, additional connection attempts will be rejected until some pending requests are completed.
Stuck Thread Max Time	If an execute thread has been working for the time specified here, it is marked as stuck. Stuck threads can be monitored, and the server can be automatically restarted by the node manager if all the execute threads are stuck.
Stuck Thread Timer Interval	How often WebLogic Server checks to see whether any threads are stuck.
Managed Server Independence Enabled	If enabled, managed servers can be started even if the admin server is not running. To do this, they must have access to configuration information from the admin server. We need to manually copy the <code>config.xml</code> and <code>SerializedSystemIni.dat</code> files from the admin server domain directory to the managed server startup directory.
MSI File Replication Enabled	If enabled, the files listed under Managed Server Independence Enabled are copied from the admin server to the managed servers every 5 minutes.
Timeout for Server Lifecycle Operations	If a shutdown or force shutdown operation is still working after this many seconds, the server will shut down immediately.

### 5.3.6 Configuration | Compilers

We can use the options on this screen to customize the Java and RMI compilers, though this is not usually necessary:



The options on this screen are:

Setting	Description
Java Compiler	Allows us to specify an alternative compiler, such as Jikes, which will be used to compile JSPs, EJB stubs, etc.
Prepend to classpath	Generally the code we need to compile is either on the server classpath or included with the application; otherwise it wouldn't run even if we could compile it. If for some reason we need additional entries to the classpath, the libraries listed here will be added before the default entries. The value of this field should use the standard path syntax for the server platform (colon-separated for UNIX, semicolon-separated for Windows).
Append to classpath	This is the same as Prepend classpath except that the libraries listed here will be added after the default entries.
Extra rmic options	The options entered here will be passed on the command line whenever rmic is invoked (typically for EJB stubs). This can be used to provide RMI stubs compatible with previous J2SE versions.

**These classpath options only apply to the compiler and will not affect the run-time classpath. To adjust the run-time classpath refer to *Section: 3.1.2.3: Customizing the Classpath*.**

