BEA WebLogic Server™

Managing WebLogic Security

Release 7.0
Document Revised: August 30, 2002

Copyright

Copyright © 2002 BEA Systems, Inc. All Rights Reserved.

Restricted Rights Legend

This software and documentation is subject to and made available only pursuant to the terms of the BEA Systems License Agreement and may be used or copied only in accordance with the terms of that agreement. It is against the law to copy the software except as specifically allowed in the agreement. This document may not, in whole or in part, be copied photocopied, reproduced, translated, or reduced to any electronic medium or machine readable form without prior consent, in writing, from BEA Systems, Inc.

Use, duplication or disclosure by the U.S. Government is subject to restrictions set forth in the BEA Systems License Agreement and in subparagraph (c)(1) of the Commercial Computer Software-Restricted Rights Clause at FAR 52.227-19; subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013, subparagraph (d) of the Commercial Computer Software--Licensing clause at NASA FAR supplement 16-52.227-86; or their equivalent.

Information in this document is subject to change without notice and does not represent a commitment on the part of BEA Systems. THE SOFTWARE AND DOCUMENTATION ARE PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. FURTHER, BEA Systems DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE SOFTWARE OR WRITTEN MATERIAL IN TERMS OF CORRECTNESS, ACCURACY, RELIABILITY, OR OTHERWISE.

Trademarks or Service Marks

BEA, Jolt, Tuxedo, and WebLogic are registered trademarks of BEA Systems, Inc. BEA Builder, BEA Campaign Manager for WebLogic, BEA eLink, BEA Manager, BEA WebLogic Commerce Server, BEA WebLogic Enterprise, BEA WebLogic Enterprise Platform, BEA WebLogic Express, BEA WebLogic Integration, BEA WebLogic Personalization Server, BEA WebLogic Platform, BEA WebLogic Portal, BEA WebLogic Server, BEA WebLogic Workshop and How Business Becomes E-Business are trademarks of BEA Systems, Inc.

All other trademarks are the property of their respective companies.

Managing WebLogic Security

Part Number	Document Revised	Software Version
N/A	August 30, 2002	BEA WebLogic Server Version 7.0

Contents

3. Customizing the Default Security Configuration

4. Using Compatibility Security

5. Configuring the SSL Protocol

About This Document

This document explains how to configure security for WebLogic Server and how to use Compatibility security. It is organized as follows:

- Chapter 1, "Overview of Security Management," explains the differences between security in WebLogic Server 6.x and WebLogic Server 7.0; describes the default security configuration in WebLogic Server 7.0; lists the configuration steps for security in WebLogic Server 7.0, and describes Compatibilty security.
- Chapter 2, "Configuring WebLogic Security," describes the security configuration steps required when using the default security configuration in WebLogic Server 7.0.
- Chapter 3, "Customizing the Default Security Configuration," describes when to customize the default security realm, how to configure WebLogic and Custom security providers, and how to set a security realm as the default security realm.
- Chapter 4, "Using Compatibility Security," describes how to configure Compatibility Security.
- Chapter 5, "Configuring the SSL Protocol," describes how to configure the SSL protocol when using WebLogic Server.
- Chapter 6, "Configuring Security for a WebLogic Domain," describes how to set security attributes on a WebLogic domain.
- Chapter 7, "Using the Java Security Manager," describes how to use the Java security manager with WebLogic Server.

Audience

This document is written for system administrators who will be configuring security for WebLogic Server deployments.

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. [[Note to writers: references to other WLS docs that cover or elaborate on the subject matter of your doc, related Sun docs, etc. go here. If you have more than a couple of references to related docs, make it into a bulleted list.]]

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.

`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 the user is told to enter from the keyboard.
	Examples:
	`import java.util.Enumeration;`
	`chmod u+w *`
	`config/examples/applications`
	`.java`
	`config.xml`
	`float`

`monospace italic text`	Placeholders. *Example*: `String CustomerName;`

`UPPERCASE MONOSPACE 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]`

Convention	Usage

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

CHAPTER

1 Overview of Security Management

The following sections provide an overview of the new security subsystem for WebLogic Server including the differences between WebLogic Server 6.x and WebLogic Server 7.0.

Note: Throughout this document, the term *6.x* refers to WebLogic Server 6.0 and 6.1 and their associated Service Packs.

Audience

Managing WebLogic Security is intended for system administrators responsible for securing WebLogic Server.

How Security Changed in WebLogic Server 7.0

WebLogic Server 7.0 offers a new security service that simplifies the configuration and management of security while offering robust capabilities for securing your WebLogic Server deployment. This section describes how the security service changed in WebLogic Server 7.0.

Change in Scope of Security Realms

In WebLogic Server 6.x, security realms provided authentication and authorization services. You chose from the File realm or a set of alternative security realms including the Lightweight Data Access Protocol (LDAP), Windows NT, Unix, or RDBMS realms. If you wanted to customize authentication, you could write you own security realm and integrate it into the WebLogic Server environment. A security realm applied to a domain and you could not have multiple security realms in a domain.

In WebLogic Server 7.0, security realms act as a scoping mechanism. Each security realm consists of a set of configured security providers, users, groups, roles, and security policies. You can configure multiple security realms in a domain, however, only one can be the default (active) security realm. WebLogic Server provides two default security realms:

- myrealm—Has the WebLogic Authentication, Identity Assertion, Authorization, Adjudication, Role Mapping, and Credential Mapping providers configured by default.
- Compatibility realm—Provides backward compatibility for 6.x security configurations. You can access an existing 6.x security configuration through the Compatibility realm.

You can no longer write a custom security realm using the application programming interfaces in WebLogic Server 7.0; rather, you configure a new security realm to provide the security services you want and then set the new security realm as the default security realm.

For information about the default security configuration in WebLogic Server, see "The Default Security Configuration in WebLogic Server 7.0" on page 1-7.

For information about configuring a security realm and setting it as the default security realm, see [xref]

For information about using Compatibility security, see Chapter 4, "Using Compatibility Security."

What Are Security Providers?

Security providers are modular components that handle specific aspects of security, such as authentication and authorization. Although applications can leverage the services offered via the default WebLogic security providers, the WebLogic Security Service flexible infrastructure also allows security vendors to write their own custom security providers for use with WebLogic Server. WebLogic security providers and custom security providers can be mixed and matched to create unique security solutions, allowing organizations to take advantage of new technology advances in some areas while retaining proven methods in others. The WebLogic Server Administration Console (referred to as the Administration Console) allows you to administer and manage all your security providers through one unified management interface.

The WebLogic Security Service supports the following types of security providers:

- **Auditing**—Auditing is the process whereby information about operating requests and the outcome of those requests are collected, stored, and distributed for the purpose of non repudiation. In other words, auditing provides an electronic trail of computer activity. The WebLogic Auditing provider supplies these services.
- **Authentication**—Authentication is the process whereby the identity or users or system processes are proved or verified. Authentication also involves remembering, transporting, and making identity information available to various components of a system when that information is needed. The WebLogic Security Service supports the following types of authentication:
  - Username and password authentication
  - Certificate-based authentication directly with WebLogic Server

- HTTP certificate-based authentication proxied through an external Web server

The WebLogic Authentication provider supplies these services.

- **Identity Assertion**—An Authentication provider that performs perimeter authentication is called an Identity Assertion provider. Identity assertion involves establishing a client's identity through the use of client-supplied tokens that may exist outside of the request. Thus, the function of an Identity Assertion provider is to validate and map a token to a username. Once this mapping is complete, a JAAS LoginModule converts the username to a Principal. The WebLogic Identity Assertion provider supplies these services.
- **Authorization**—Authorization is the process whereby the interactions between users and WebLogic resources are limited to ensure integrity, confidentiality, and availability. In other words, authorization is responsible for controlling access to WebLogic resources based on user identity or other information. The WebLogic Authorization provider supplies these services.
- **Adjudication**—When multiple Authorization providers are configured in a security realm, each may return a different answer to the "is access allowed" question for a given resource. Determining what to do if multiple Authorization providers do not agree is the primary function of an Adjudication provider. Adjudication providers resolve authorization conflicts by weighting each Authorization provider's answer and returning a final decision.
- **Role Mapping**—Obtains a computed set of rules granted to a requestor for a given resource. Role Mapping providers supply Authorization providers with this information so that the Authorization provider can answer the "is access allowed?" question for WebLogic resources that use role-based security (that is, Web applications and Enterprise JavaBeans (EJBs).
- **Credential Mapping**—Credential mapping is the process whereby a legacy system's authentication mechanism is used to obtain an appropriate set of credentials to authenticate users to a target resource. A Credential Mapping provider supplies credential mapping services and bring new types of credentials into the WebLogic Server environment.

For information about the functionality provided by the WebLogic security providers, see The WebLogic Security Providers in the *Introduction to WebLogic Security*.

For information about the default security configuration, see "The Default Security Configuration in WebLogic Server 7.0" on page 1-7.

For information about writing a custom security provider, see *Developing Security Providers for WebLogic Server*.

Security Policies Instead of ACLs

In WebLogic Server 6.x, access control lists (ACLs) and permissions were used to protect WebLogic resources. In WebLogic Server 7.0, security policies replace ACLs and permissions. Security policies answer the question "who has access" to a WebLogic resource. A security policy is created when you define an association between a WebLogic resource and a user, group, or role. You can also optionally associate a time constraint with a security policy. A WebLogic resource has no protection until you assign it a security policy.

For information about creating security policies, see "Protecting WebLogic Resources" on page 2-10.

For information about using ACLs in Compatibility security, see "Defining ACLs in the Compatibility Realm" on page 4-34.

WebLogic Resources

WebLogic Server defines the following resources:

- Administrative resources such as the WebLogic Server Administration Console (referred to as the Administration Console) and the `weblogic.Admin` tool.
- Application resources (including individual EAR files)
- COM resources
- EIS resources
- EJB resources (including individual EJB method, components, and JAR files)
- JDBC resources
- JNDI resources
- JMS resources

- MBean resources
- Server resources
- Web resources (including individual WAR files)
- Web Services resources

Deployment Descriptors and the WebLogic Security Providers

The WebLogic Security Service now uses information defined in deployment descriptors to grant roles and define security policies for WebLogic resources and create credentials maps for remote users wanting to access WebLogic resources. When WebLogic Server is booted for the first time, role, security policy, and credential map information stored in `weblogic.xml`, `weblogic-ejb-jar.xml`, and `weblogic-ra.xml` files is loaded into the Authorization, Role Mapping, and Credential Mapping providers configured in the default security realm. Changes to the information can then be made through the Administration Console.

To use information in deployment descriptors, at least one Authorization, Credential Mapping, and Role Mapping provider in the security realm must implement the `DeployableAuthorizationProvider`, `DeployableCredentialProvider`, and `DeployableRoleProvider` Security Service Provider Interface (SSPI). This SSPI allows the providers to store (rather than retrieve) information from deployment descriptors. By default, the WebLogic Authorization, Role Mapping, and Credential Mapping providers implement this SSPI.

If you change role, security policy, and credential map information in deployment descriptors through the Administration Console, use the Ignore Security Data in Deployment Descriptors attribute to ensure changes made through the Administration Console are not overwritten by old information in the deployment descriptors.

For more information, see Preventing Overwriting of Administration Console Changes.

The Default Security Configuration in WebLogic Server 7.0

To simplify the configuration and management of security in WebLogic Server, a default security realm (myrealm) is provided. The default security realm has WebLogic Authentication, Identity Assertion, Authorization, Adjudication, Role Mapping, and Credential Mapping providers configured. When using the default security configuration, you only need to define groups, users, and roles for the security realm and create security policies for the WebLogic resources in the domain. You also need to verify that the configuration of the embedded LDAP server configuration is appropriate for your use. Optionally, you can configure an Auditing provider for the default realm.

For a description of the functionality provided by the WebLogic Security providers, see The WebLogic Security providers in *Introduction to WebLogic Security*. If the WebLogic security providers do not fully meet your security requirements, you can supplement or replace them. For more information, see *Developing Security Services for WebLogic Server*.

If the default security configuration does not meet your requirements, you can create a new security realm with any combination of WebLogic and custom security providers and then set the new security realm as the default security realm. For more information, see "Customizing the Default Security Configuration" on page 3-1.

Configuration Steps for Security

Because the security features are closely related, it is difficult to determine where to start when configuring security. In fact, configuring security for your WebLogic Server deployment may be an iterative process. Although more than one sequence of steps may work, BEA Systems recommends the following procedure:

1. Determine whether or not to use the default security configuration by reading "Why Customize the Default Security Configuration?" on page 3-1.
   - If you are using the default security configuration, begin at step 3.

- If you are not using the default security configuration, begin at step 2.

2. Change the configuration of the default security realm or create a new security realm. This step is optional. See [xref].
3. Define groups for the default security realm. See "Defining Groups" on page 2-2.
4. Define users for the default security realm. See "Defining Users" on page 2-4.
5. Define global roles for the default security realm.
6. Grant users and groups the global roles. See "Defining Global Roles" on page 2-13.
7. Protect WebLogic resources with roles (either global or scoped) and security policies. See "Protecting WebLogic Resources" on page 2-10.
8. Configure the embedded LDAP server. See "Configuring the Embedded LDAP Server" on page 2-19.
9. Configure the SSL protocol. (This step is optional but encouraged.) For more information, see "Configuring the SSL Protocol" on page 5-1.
10. Set attributes for logging security information. By default, WebLogic Server enables the logging of security information. However, you should review the attributes and their settings to ensure they are appropriate for your environment.

In addition, you can:

- Map a WebLogic Server user defined in the configured Authentication provider to credentials for another system (for example, the username and password for an Oracle database). See "Providing WebLogic Server Users Access to Other Applications" on page 2-9.
- Configure a connection filter. See "Configuring Connection Filtering" on page 6-3.
- Enable interoperability between WebLogic domains. See "Enabling Trust Between WebLogic Domains" on page 6-1.

What Is Compatibility Security?

Compatibility security refers to the capability to run security configurations from WebLogic Server 6.x in WebLogic Server 7.0. In Compatibility security, you configure 6.x security realms, define users, groups, and ACLs, manage the protection of user accounts, and install custom auditing providers.

The only security realm available in Compatibility security is the Compatibility realm. The Realm Adapter providers in the Compatibility realm allow backward compatibility to the authentication and authorization services in 6.x security realms. You must run Compatibility security in order to access the Compatibility realm and the Realm Adapter providers through the WebLogic Server Administration Console (referred to as the Administration Console). For more information, see "Using Compatibility Security" on page 4-1.

Management Tasks Available in Compatibility Security

Because Compatibility security only allows you to access authentication, authorization, and custom auditing services supported in WebLogic Server 6.x, not all 6.x security tasks are allowed in Compatibility security. Use Compatibility security to:

1. Change the password of the `system` user to protect your WebLogic Server deployment.
2. Change the security realm in the Compatibility realm. By default, WebLogic Server is installed with the File realm in place. However, you may prefer an alternate security realm or a custom security realm.
3. Define users for the security realm in the Compatibility realm. Organize users further by implementing groups in the security realm.
4. Define ACLs and permissions for the resources in your WebLogic Server deployment.

You can still use the SSL protocol, configure connection filters, and enable interoperability between domains; however, you use the security features available in WebLogic Server 7.0 to perform these tasks. For more information, see:

- Chapter 5, "Configuring the SSL Protocol"
- Chapter 6, "Configuring Security for a WebLogic Domain"

2 Configuring WebLogic Security

The following sections describe the security configuration steps in WebLogic Server 7.0:

Note: If you are customizing the default security configuration, follow the procedures in Chapter 3, "Customizing the Default Security Configuration" then complete the relevant procedures in this chapter.

Defining Groups

Note: This section applies to the WebLogic Authentication provider only. If you customize the default security configuration to use another authentication provider, you must use the administration tools supplied by that provider to define a group.

When upgrading to the WebLogic Authentication provider, there is no automatic way to load existing groups into the provider. For WebLogic Server 7.0, defining existing groups in the WebLogic Authentication provider is a manual step. If you have a large number of existing users, consider using the Realm Adapter Authentication provider.

A group represents a set of users who usually have something in common, such as working in the same department in a company. Groups are a means of managing a number of users in an efficient manner. When a group is granted a role or is used to create a security policy, the role or security policy is assigned to all members of the group. For more efficient management, BEA recommends using groups to manage large numbers of users.

By default, WebLogic Server has groups listed in Table 2-1.

Table 2-1 Default WebLogic Groups

Group Name	Permissions
Users	If you identity yourself when you log in (for example, you logged in through a Web page), you are a member of the Users group
Everyone	Regardless of whether you identity yourself when you log in, you are in the Everyone group. (The Everyone group includes the Users group.)
Administrators	View and modify all resource attributes and perform start and stop operations. In order to boot WebLogic Server, a user must be in this group.

Table 2-1 Default WebLogic Groups

Group Name	Permissions
Operators	View all resource attributes and perform server lifecycle operations. By default, this group is empty.
Deployers	View all resource attributes and deploy applications such as EJBs. By default, this group is empty.
Monitors	View all resource attributes, modify resource attributes, and perform operations that are not restricted by a role. By default, this group is empty.

BEA recommends using initial cap, plural names for groups. Group names must be unique.

To define a group:

1. Expand the Security-->Realms nodes.
2. Click the name of the realm you are configuring (for example, myrealm).
3. Click Groups.

   The Groups tab appears. This tab displays the names of all groups defined in the Weblogic Authentication provider.
4. Click the Configure a New Group... link.
5. On the Groups-->General tab, enter the name of the group.
6. Enter a short description of the group (for example, `Product Managers for Code Examples`).
7. Click Apply to save your changes.
8. Click the Membership tab to add existing groups to the new group.
   - All the available groups appear in the Possible Groups table.
   - All the groups currently defined for the group appear in the Current Groups table.

To add a group to another group, highlight the desired group name and click the right arrow to move the group name to the Current Groups table.

9. Click Apply to save your changes.

If you have a large number of groups, use the Filter option on the Groups tab to search the embedded LDAP server and list the groups that match the search criteria. The Filter option uses the asterisk (*) as the wildcard character. Use the Filter option if you have more than 48 groups.

To delete a group:

1. Expand the Security-->Realms nodes.
2. Click the name of the realm you are configuring (for example, myrealm).
3. Click Groups.

   The Groups table appears. This table displays the names of all groups defined in the WebLogic Authentication provider.

4. To delete a group, click the trash can icon in the corresponding row of the Groups table.

Defining Users

Note: This section applies to the WebLogic Authentication provider only. If you customize the default security configuration to use another authentication provider, you must use the administration tools supplied by that provider to define a user.

When upgrading to the WebLogic Authentication provider, there is no automatic way to load existing users into the provider. For WebLogic Server 7.0, adding existing users to the WebLogic Authentication provider is a manual step. If you have a large number of existing users, consider using the Realm Adapter Authentication provider.

Users are entities that can be authenticated. A user can be a person or software entity, such as a Java client. Each user is given a unique identity within a security realm. The minimum password length for a user in the WebLogic Authentication provider is 8

characters. As a system administrator you must guarantee that no two users in the system are identical. For more efficient management, BEA recommends adding users to groups. User names must be unique.

The `guest` user is no longer supplied by default in WebLogic Server 7.0. To use the `guest` user, use Compatibility security or define the `guest` user as a user in the Authentication provider in the default security realm. For more information, see Upgrading Security in the *Upgrade Guide for BEA WebLogic Server 7.0*.

In WebLogic Server 6.x, any unauthenticated user (anonymous user) was identified as a `guest` user. WebLogic Server allowed the `guest` user access to WebLogic resources. However, this functionality presented a potential security risk so the functionality was modified. In 7.0, WebLogic Server distinguishes between the `guest` user and an anonymous user. To use the `guest` user as you did in WebLogic Server 6.x, add the `guest` user to the Authentication provider in the default security realm by setting the following argument when starting WebLogic Server:

```
-Dweblogic.security.anonymousUserName=guest
```

Without this argument, an anonymous user has the name `<anonymous>`. This command-line argument was added to assist existing WebLogic Server customers to upgrade their security functionality. You should take great caution when using the `guest` user in a production environment.

Note: Do not use the username/password combination `weblogic`/`weblogic` in production.

To define a user:

1. Expand the Security-->Realms nodes.
2. Click the name of the realm you are configuring (for example, myrealm).
3. Click Users.

   The Users table appears. This table displays the names of all users defined in the WebLogic Authentication provider.
4. Click the Configure a New User... link.
5. On the User-->General tab, enter the name of the user.
6. Enter a password for the user.
7. Click Apply to save your changes.

8. Select the Groups tab to add the user to a group.
   - All the available groups appear in the Possible Groups table.
   - All the groups to which the user belongs appear in the Current Groups table.

   To add a user to a group, highlight the desired group name and click the right arrow to move the group name to the Current Groups table.
9. Click Apply to save your changes.

If you have a large number of users, use the Filter option on the Users tab to search the store and list the users that match the search criteria. The Filter option uses the asterisk (*) as the wildcard character. Use the Filter option if you have more than 48 users.

To delete a user:

1. Expand the Security-->Realms nodes.
2. Click the name of the realm you are configuring (for example, myrealm).
3. Click Users.

   The Users table appears. This table displays the names of all users defined in the WebLogic Authentication provider.
4. To delete a user, click the trash can icon in the corresponding row of the Users table.

Protecting User Accounts

Weblogic Server provides a set of attributes to protect user accounts from intruders. By default, these attributes are set for maximum protection. As a system administrator, you have the option of turning off all the attributes, increasing the number of login attempts before a user account is locked, increasing the time period in which invalid login attempts are made before locking the user account, and changing the amount of time a user account is locked. Remember changing these attributes lessens security and leaves user accounts vulnerable to security attacks.

To set the User Lockout attributes:

1. Expand the Security-->Realms nodes.

2. Click the name of the realm you are configuring (for example, myrealm).
3. Select the User Lockout tab.
4. Define the desired attributes on this tab by entering values at the appropriate prompts and selecting the required checkboxes. (For details, see the following table).

   If a user account exceeds the values set for the attributes on this tab, the user account becomes locked and the Users table has the word `Details` in the table row for the user.
5. To save your changes, click Apply.
6. Reboot WebLogic Server.

The following table describes each attribute on the User Lockout tab.

Table 2-2 User Lockout Attributes

Attribute	Description
Lockout Enabled	Requests the locking of a user account after invalid attempts to log in to that account exceed the specified Lockout Threshold. By default, this attribute is enabled.
Lockout Threshold	Number of failed user password entries that can be tried before that user account is locked. Any subsequent attempts to access the account (even if the username/password combination is correct) raise a Security exception; the account remains locked until it is explicitly unlocked by the system administrator or another login attempt is made after the lockout duration period ends. Invalid login attempts must be made within a span defined by the Lockout Reset Duration attribute. The default is 5.
Lockout Duration	Number of minutes that a user's account remains inaccessible after being locked in response to several invalid login attempts within the amount of time specified by the Lockout Reset Duration attribute. The default is 30 minutes.

Table 2-2 User Lockout Attributes

Attribute	Description
Lockout Reset Duration	Number of minutes within which invalid login attempts must occur in order for the user's account to be locked. An account is locked if the number of invalid login attempts defined in the Lockout Threshold attribute happens within the amount of time defined by this attribute. For example, if the value in Lockout Reset Duration attribute is 5 minutes, the Lockout Threshold is 3, and 3 invalid login attempts are made within a 6 minute interval, then the account is not locked. If 3 invalid login attempts are made within a 5 minute period, however, then the account is locked. The default is 5 minutes.
Lockout Cache Size	Specifies the intended cache size of unused and invalid login attempts. The default is 5.
Lockout GC Threshold	The maximum number of invalid login records that the server keeps in memory. If the number of invalid login records is equal to or greater than the value of this attribute, the server's garbage collection purges the records that have expired. A record expires when the user associated with the record have been locked out. The default is 400 records.

Note: The UserLockout attributes apply to the security realm and all its security providers. If you are using an Authentication provider that has its own mechanism for protecting user accounts, disable the Lockout Enabled attribute.

If a user account becomes locked and you delete the user account and add another user account with the same name and password, the UserLockout attribute will not be reset.

Unlocking a User Account

To unlock a user account:

1. Click the Details link in the Users table.

   The Details tab describes the event that occurred when the user was locked out.

2. Click Unlock.

Providing WebLogic Server Users Access to Other Applications

Resource adapters defined by the J2EE Connector Architecture can acquire the credentials necessary to authenticate WebLogic Server users to another application. The Connector container in WebLogic Server can retrieve the appropriate set of credentials for a WebLogic user using a credential map defined in a Credential Mapping provider. A credential map creates an association between a WebLogic Server user and an identity (by default, a username and password combination) used to authenticate that WebLogic Server user to a remote application such as an Oracle database, a SQL server, or a SAP application.

To create a credential map:

1. Expand the Connectors node.
2. Right-click on the resource adapter for the remote application.
3. Click the CredMaps... link.

   The CredMaps tab displays all the credential maps defined in the configured Credential Mapper.

4. Click the Configure a New Cred Map... link.
5. Enter a username/password combination name for the remote application.

6. Click Apply. The information is stored in the embedded LDAP server.
7. Click the RoleMaps... link.
8. On the RoleMaps tab, click the Configure a New Role Map... link.
9. In the WLS User attribute, enter a WebLogic Server user name.
10. In the Remote User attribute, enter the user name that provides access to the remote application (the user you defined when creating a credential map).
11. Click Apply.

For more information about resource adapters, credential maps, and the WebLogic Server implementation of the J2EE Connector Architecture, see *Programming the J2EE Connector Architecture*.

Protecting WebLogic Resources

TBS

[This section to be replace by new section from Jen Hocko.]

Understanding Roles

Roles are an abstract, logical collections of users similar to a group. The difference between groups and roles is a group is a static identity that a system administrator assigns, while membership in a role is dynamically calculated based on data such as username, group membership, or the time of day. Roles are granted to individual users or to groups. For more efficient management, BEA recommends granting roles to groups rather than to individual users. Once you create a role, you define an association between the role and a WebLogic resource. This association (called a security policy) specifies who has what access to the WebLogic resource.

WebLogic Server supports two types of roles:

- Global roles—Apply to all WebLogic resources in a security realm. WebLogic Server defines a set of default global roles for protecting WebLogic resources.

You do not have to use the default global roles. You can create global roles that more closely reflect your own business structure and practices.

- Scoped roles—Apply to a specific instance of a WebLogic resource such as a method of an EJB or a branch of a JNDI tree. Most roles are scoped.

In WebLogic Server 6.x, roles were used to protect EJBs and Web applications only. WebLogic Server version 7.0 expands the use of roles to protect all of the defined WebLogic Server resources. For a list of the defined WebLogic resources, see "WebLogic Resources" on page 1-5.

Multiple roles (either scoped or global) can be applied to a WebLogic resource. Both scoped and global roles are stored in the default Role Mapping provider. To use a role to create a security policy for a WebLogic resource, the role must be in the Role Mapping provider configured in the default security realm. By default, the WebLogic Role Mapping provider is the configured Role Mapping provider.

Default Global Roles and Permissions

Table 2-3 lists the default global roles provided by WebLogic Server.

Table 2-3 Global Roles and Permissions

Global Role	Permissions
Admin	View and modify the server configuration. Deploy applications, EJBs, startup and shutdown classes, J2EE Connectors, and Web Service components, and edit deployment descriptors.
Deployer	View the server configuration. Deploy applications, EJBs, startup and shutdown classes, J2EE Connectors, and Web Service components, and edit deployment descriptors.
Operator	View the server configuration. Start, resume, and stop servers by default.
Monitor	View the server configuration.

Table 2-3 Global Roles and Permissions

Global Role	Permissions
anonymous	All users (the group `Everyone`) are granted this role. This convenience role can be specified in security deployment descriptors in `weblogic.xml` and `weblogic-ejb-jar.xml` files.

Note: If you create a scoped role with the same name as a global role, the scoped role takes precedence over the global role.

Default Group Associations

By default, WebLogic Server grants four of the default groups a global role. By adding a user to one of these groups, the user is automatically granted the global role. (See Table 2-4.)

Table 2-4 Default Group Associations

Members of This Group	Are In This Role
Administrators	Admin
Deployers	Deployer
Operators	Operator
Monitors	Monitor

What Is a Role Statement?

A Role statement contains expressions that define who is granted the role. The use of `and` or `or` in these expressions is important. (You create Role statements when you define roles in the Administration Console.)

- `And` is used in an expression to specifiy that all the conditions in the expression must be met in order for the role to be granted.

- `Or` is used in an expression to specify that at least one of the conditions in the expression must be met in order for the role to be granted.

For example, the following Role statements are different:

```
Donna is member of group BankTeller
and Donna is member of group BankManagers
```

```
Donna is member of group BankTeller
or Donna is member of group BankManagers
```

The first expression specifies that user `Donna` must be a member of the `BankTeller` group and the `BankManager` group to be granted the role.

The second expression specifies that user `Donna` must be either a member of the `BankTeller` group or the `BankManager` group to be granted the role.

When granting users and groups roles, the ordering of expressions in a Role statement is important. More restrictive expressions should come later in the Role statement. The entire Role statement must be true in order for the user or group to be granted the role.

Defining Global Roles

Note: BEA recommends using initial cap, singular names for global roles, for example, SecurityEng.

To define a global role:

1. Expand the Security-->Realms nodes.
2. Click the name of the realm you are configuring (for example, myrealm).
3. Click Roles.

   The Roles tab appears. This tab displays the names of all roles defined in the default Role Mapping provider.
4. Click the Configure a New Role... link.
5. On the Roles-->General tab, enter the name of the role. (For example, `SecurityEng`).
6. Click Apply.
7. Grant the role to users and/or groups.

Select the Roles-->Conditions tab. Use the following options available in the Role Condition table to grant a role to users and/or groups:

- User name of the caller—Grants a user the role. Multiple users can be granted the same role, however, BEA recommends adding the users to a group and then granting the group the role. The user must be defined in the Authentication provider configured in the default security realm.
- Caller is member of group—Grants a group the role. Multiple groups can be granted the same role. The group must be defined in the Authentication provider in the default security realm.
- Hours of Access are between—Specifies a time period in which the role is in effect.

8. Chose an option and click Add.
9. On the Add window, enter the name of the user or group that is to be granted the role.

   Use the Add window to grant multiple users or groups a role. As you grant users and groups a role, the users or groups are listed on the Add window.

10. Click Add.
11. Use the buttons on the Add window to modify the Role statement:
    - Move Up and Move Down change the ordering of expressions in the Role statement.
    - Change switches the `and` and `or` statements in the highlighted expression.
    - Remove deletes the highlighted expression.
12. Click OK.
13. In addition to granting a role to users or groups, specify a time period (called a time constraint) in which the role is in effect. For example, you might create a `BankTeller` role that is only in effect when the bank is open. To associate a time constraint with a role, choose the Hours of Access Are Between option on the Role Condition tab.
14. In the Time Constraint window, specify the hours during which this role is to be active.
15. Click OK.

16. The expressions for the users and groups granted the role and the time constraints for the role appear in the Role statement on the Role Conditions tab. Use Move Up, Move Down, Change, and Edit to modify the Role statement. Reset clears the Role statement.
17. When you have finished making changes to the users, groups, and time constraints for a role, click Apply to grant the global role.

    The information is stored in a Role Mapping provider. By default, the WebLogic Role Mapping provider is configured and the role information is stored in the embedded LDAP server.

Removing a User, Group, or Time Constraint From a Global Role

1. Highlight the expression for the user, group, or time constraint in the Role statement on the Role Conditions tab.
2. Click Remove.
3. Click Apply to save your change.

Deleting Global Roles

To delete a global role:

1. Expand the Security -->Realms nodes.
2. Click the name of the realm you are configuring (for example, myrealm).
3. Click Roles.

   The Roles table appears. This table displays the names of the global roles defined in the security realm.
4. To delete a role, click the trash can icon in the corresponding row of the Roles table.

   A confirmation window appears.

5. Click Yes to delete the global role.

Creating Scoped Roles

To create a scoped role for a WebLogic resource:

1. Expand the node that contains the WebLogic resource for which you want to create a role and highlight the resource.
2. Click the right mouse button.
3. Select the Define Role... link.
4. Grant users and groups the scoped role as you granted a global role. For more information, see Step 7 in "Defining Global Roles" on page 2-13. The role applies only to the highlighted resource.
5. Click Apply.

Understanding WebLogic Security Policies

In the previous release of WebLogic Server, ACLs were used to protect WebLogic resources. In WebLogic Server 7.0, security policies replace ACLs and answer the question "who has access" to a WebLogic resource. A security policy is created when you define an association between a WebLogic resource and one or more users, groups, or roles. You can optionally define a time constraint for a security policy. A WebLogic resource has no protection until you assign it a security policy.

You assign security policies to any of the defined WebLogic resources (for example, an EJB resource or a JNDI resource) or to attributes or operations of a particular instance of a WebLogic resource (an EJB method or a servlet within a Web Application). If you assign a security policy to a type of resource, all new instances of that resource inherit that security policy. Security policies assigned to individual resources or attributes override security policies assigned to a type of resource.

Security policies are stored in an Authorization provider. By default, the WebLogic Authorization provider is configured and security policies are stored in the embedded LDAP server.

To use a user or group to create a security policy, the user or group must be defined in the Authentication provider configured in the default security realm. To use a role to create a security policy, the role must be defined in the Role Mapping provider configured in the default security realm. By default, the WebLogic Authentication and Role Mapping providers are configured.

Default Security Policies

By default, WebLogic Server has defined security policies for the WebLogic resources. These security policies are based on roles and groups. You also have the option of basing a security policy on a user. BEA recommends basing security policies on roles rather than users or groups. Basing security policies on roles allows you to manage access based on a role a user or group is granted, which is a more efficient method of management.

Table 2-5 lists the default security policies.

Table 2-5 Default Security Policies for WebLogic Resources

WebLogic Resource	Security Policy
Administrative resources	Role Admin, Role Deployer, Role Operator, Role Monitor
COM resources	Role jCOMRole
EIS resources	Group Everyone
EJB resources	Group Everyone
JMS resources	Group Everyone
JDBC resources	Group Everyone
JNDI resources	Group Everyone
MBean resources	Group Everyone
Server resources	Role Admin, Role Operator
Web Service resources	Group Everyone

What Is a Policy Statement?

A policy statement contains expressions that define who is granted access to the WebLogic resource. The use of `and` and `or` in these expressions as well as the ordering of the expressions is important.

- `And` is used in an expression to specify that all the conditions must be met in order for the security policy to be created.
- `Or` is used in an expression to specify that at least one of the conditions in the expression must be met in order for the security policy to be create.

More restrictive expressions should come later in the policy statement. The entire policy statement must be true in order for security policy to be created.

Deleting a Security Policy

To delete a security policy:

1. Navigate to and highlight the WebLogic resource you want to protect with a security policy.
2. Click the right mouse button.
3. Select the Define Policy... link.

   The Policy Conditions tab appears.
4. Click Delete.

   The expressions in the Policy statement are deleted.
5. Click Apply.

   The security policy is deleted from the Authorization provider.

Removing a User, Group, or Time Constraint From a Policy Statement

To remove a specific user, group, or time constraint from a Policy statement:

1. Highlight the user, group, or time constraint in the Policy statement on the Policy Conditions tab.
2. Click Remove.
3. Click Apply to make the change.

Configuring the Embedded LDAP Server

The embedded LDAP server contains user, group, group membership, role, security policy and credential information. The WebLogic Authentication, Authorization, Role Mapping, and Credential Mapping providers use the embedded LDAP server as a storage mechanism. If you use any of these WebLogic security providers, you need to configure the embedded LDAP server.

To configure the embedded LDAP server:

1. Expand the Domain node (for example, Examples).
2. Select the Security-->Embedded LDAP tabs.
3. Configure the attributes on this tab by entering values at the appropriate prompts and selecting the required checkboxes. (See Table 2-6 for a description of the attributes on the Embedded LDAP tab.)
4. Click Apply to save your changes.
5. Reboot WebLogic Server.

Table 2-6 Embedded LDAP Server Attributes

Attribute	Description
Credential	The credential (usually a password) used to connect to the embedded LDAP server.
Backup Hour	The hour at which to backup the embedded LDAP server. The default is 23.
Backup Minute	The minute at which to backup the embedded LDAP server. This attribute is used in conjunction with the Back Up Hour attribute to determine the time at which the embedded LDAP server is backed up. The default is 5 minutes.
Backup Copies	The number of backup copies of the embedded LDAP server. The default is 7.
Cache Enabled	Specifies whether or not a cache is used with the embedded LDAP server.
Cache Size	The size of the cache (in K) that is used with the embedded LDAP server. The default is 32K.
Cache TTL	The time-to-live (TTL) of the cache in seconds. The default is 60 seconds.
Refresh Replica At Startup	Specifies whether or not a Managed Server should refresh all replicated data at boot time. This attribute is useful if you have made a large number of changes while the Managed Server was not active and you want to download the entire replica instead of having the Admin Server push each change to the Managed Server. The default is false.
Master First	Specifies that connections to the master LDAP server should always be made instead of connections to the local replicated LDAP server.

When you use the embedded LDAP Server in a WebLogic Server domain, updates are sent to a master LDAP server. The master LDAP server maintains a log of all changes. The master LDAP server also maintains a list replicated servers and the current change

status for each one. The master LDAP server sends appropriate changes to each replicated server and updates the change status for each server. This process occurs every 30 seconds. The master LDAP server is the embedded LDAP server on the Admin server. The replicated servers are all the Managed Servers in the WebLogic Server domain.

Note: The WebLogic Securiy providers stored their data in the embedded LDAP server. When you delete a WebLogic Security provider, the security data in the embedded LDAP server is not automatically deleted. The security data remains in the embedded LDAP server in case you want to use the provider again. Use an external LDAP browser to delete the security data from the embedded LDAP server.

Configuring Backups for the Embedded LDAP Server

To configure the backups of the embedded LDAP server:

1. Expand the Domain node (for example, Examples).
2. Click the Security-->the Embedded LDAP tabs.
3. Set the Backup Hour, Backup Minute, and Backup Copies attributes on the Embedded LDAP Server tab.
4. Click Apply to save your changes.
5. Reboot WebLogic Server.

3 Customizing the Default Security Configuration

The following sections provide information and procedures for creating a new security realm.

Why Customize the Default Security Configuration?

Customize the default security configuration if you want to:

- Replace one of the WebLogic security providers with a custom security provider.
- Upgrade from Compatibility security to the security providers in WebLogic Server 7.0.
- Change the default attribute settings for a WebLogic Security provider.
- Configure additional security providers in the default security realm. (For example, if you want to use two Authentication providers, one that uses the embedded LDAP server and one that uses an existing store of users and groups.)
- Use an Authentication provider that accesses an LDAP server other than the embedded LDAP server.

- Use an existing store of users and groups instead of defining users and groups in the WebLogic Authentication provider.
- Add an Auditing provider to the default security realm.

The easiest way to customize the default security configuration is to modify the default security realm (myrealm) to contain the security providers you want. You can also customize the default security configuration by creating a new security realm, configuring security providers in that realm, and setting the new security realm as the default security realm. BEA recommends this process when upgrading a security configuration.

For example, if you are upgrading to the new security features in WebLogic Server 7.0, you may want to create a test security realm, configure security providers in the test security realm, and populate the security providers with security data (users, groups, roles, and security policies) for your application. Once the test security realm is working properly, you can set this test security realm as the default security realm.

For any security realm to be valid, you must configure each of the following types of security providers (in any order):

- Authentication
- Authorization
- Adjudication
- Credential Mapping
- Role Mapping

At least one Authorization, Credential Mapping, and Role Mapping provider in the security realm must implement the `DeployableAuthorizationProvider`, `DeployableCredentialProvider`, and `DeployableRoleProvider` Security Service Provider Interface (SSPI). This SSPI allows the providers to store (rather than retrieve) information from deployment descriptors.

For information about customizing the default security configuration, see:

- Configuring a Custom Security Provider [xref]
- Configuring the WebLogic Auditing provider [xref]
- Configuring an LDAP Authentication provider [xref]
- Configuring the Realm Adapter Authentication provider [xref]

Configuring a WebLogic Adjudication Provider

When multiple Authorization providers are configured in a security realm, each may return a different answer to the "is access allowed" question for a given WebLogic resource. This answer may be `PERMIT`, `DENY`, or `ABSTAIN`. Determining what to do if multiple Authorization providers do not agree on the answer is the primary function of the Adjudication provider. Adjudication providers resolve authorization conflicts by weighting each Authorization provider's answer and returning a final decision.

Each security realm must have an Adjudication provider configured. You can use either a WebLogic Adjudication provider or a Custom Adjudication provider in a security realm. This section describes how to configure a WebLogic Adjudication provider. For information about configuring a custom security provider (including a Custom Adjudication provider), see "Configuring a Custom Security Provider" on page 3-25.

Note: The Administration Console refers to the WebLogic Adjudication provider as the Default Adjudicator.

To configure a WebLogic Adjudication provider:

1. Expand the Security-->Realms nodes.
2. Click the name of the realm you are configuring (for example, TestRealm.)
3. Expand the Providers node.
4. Click Adjudicators.

   The Adjudicators table displays the name of the default Adjudication provider for the realm that is being configured.
5. Click the Configure a new Default Adjudicator... link.
6. Optionally, on the General tab, set the Require Unanimous Permit attribute.

   The Require Unanimous Permit attribute determines how the WebLogic Adjudication provider handles a combination of `PERMIT` and `ABSTAIN` votes from the configured Authorization providers.

- If the attribute is enabled, all Authorization providers must vote `PERMIT` in order for the Adjudication provider to vote `true`. By default, the Require Unanimous Permit attribute is enabled.
- If the attribute is disabled, `ABSTAIN` votes are counted as `PERMIT` votes. To disable the Require Unanimous Permit attribute, click the checkbox.

7. Click Apply to save your changes.
8. Reboot WebLogic Server.

Configuring a WebLogic Auditing Provider

Auditing is the process whereby information about operating requests and the outcome of those requests are collected, stored, and distributed for the purposes of non-repudiation. In other words, Auditing providers produce an electronic trail of computer activity. Configuring an Auditing provider is optional. The default security realm (myrealm) does not have an Auditing provider configured.

You can use either a WebLogic Auditing provider or a Custom Auditing provider in a security realm. This topic describes how to configure a WebLogic Auditing provider. For information about configuring a custom security provider (including a Custom Auditing provider), see "Configuring a Custom Security Provider" on page 3-25.

Warning: Using an Auditing provider affects the performance of WebLogic Server even if only a few events are logged.

The Administration Console refers to the WebLogic Auditing provider as the Default Auditor.

To configure the WebLogic Auditing provider:

1. Expand the Security-->Realms nodes.
2. Click the name of the realm you are configuring (for example, TestRealm).
3. Expand the Providers node.
4. Click Auditors.

The Auditors table displays the name of the default Auditing provider for the realm that is being configured. By default, this table is empty.

5. Click the Configure a new Default Auditor... link.

   The General tab appears.

6. Choose the severity level appropriate for your WebLogic Server deployment.

   The Auditing provider audits a particular security event based on the event level specified in the Severity attribute. Auditing can be initiated when the following levels of security events occur:

   - INFORMATION
   - WARNING
   - ERROR (the default setting)
   - SUCCESS
   - FAILURE
   - none

7. Click Create to save your changes.

8. Reboot WebLogic Server.

The audit events generated by the WebLogic Auditing provider are saved in the `DefaultAuditRecorder.log` file, which is located in the *`bea_home`*`\user_projects\`*`domain`* directory (where *`bea_home`* represents the central support directory for all BEA products installed on one machine, and *`domain`* represents the name of the domain you create). The WebLogic Auditing provider logs the following events:

Table 3-1 WebLogic Auditing Provider Events

Audit Event	Indicates...
AUTHENTICATE	Simple authentication (username and password) occurred.
ASSERTIDENTITY	Perimeter authentication (based on tokens) occurred.

Table 3-1 WebLogic Auditing Provider Events

Audit Event	Indicates...
USERLOCKED	A user account is locked because of invalid login attempts.
USERUNLOCKED	The lock on a user account is cleared.
USERLOCKOUTEXPIRED	The lock on a user account expired.

Choosing an Authentication Provider

Authentication is the process whereby the identity of users or system processes are proved or verified. Authentication also involves remembering, transporting, and making identity information available to various components of a system when that information is needed.

The WebLogic Server security architecture supports: certificate-based authentication directly with WebLogic Server; HTTP certificate-based authentication proxied through an external Web server; perimeter-based authentication (Web server, firewall, VPN); and authentication based on multiple security token types and protocols.

Authentication is performed by an Authentication provider. WebLogic Server offers the following types of Authentication providers:

- *The WebLogic Authentication provider* allows you to manage users and groups in one place, the embedded LDAP server.
- *LDAP Authentication providers* access external LDAP stores. WebLogic Server provides LDAP Authentication providers which access Open LDAP, Netscape iPlanet, Microsoft Active Directory and Novell NDS stores.
- *The Realm Adapter Authentication provider* accesses user and group information stored in 6.x security realms.
- *The WebLogic Identity Assertion provider* validates X.509 and IIOP-CSIv2 tokens and maps a token to a username.

In addition, you can use:

- Custom Authentication providers, which offer different types of authentication technologies.
- Custom Identity Assertion providers, which support different types of tokens.

Note: The Administration Console refers to the WebLogic Authentication provider as the Default Authenticator and the WebLogic Identity Assertion provider as the Default Identity Asserter.

Each security realm must have one at least one Authentication provider configured. The WebLogic Security Framework is designed to support multiple Authentication providers (and thus multiple LoginModules) for multipart authentication. Therefore, you can use multiple Authentication providers as well as multiple types of Authentication providers in a security realm. For example, if you want to use both a retina scan and a username/password-based form of authentication to access a system, you configure two Authentication providers. Use the JAAS Control Flag attribute to set up dependencies between Authentication providers and allow single-sign on between providers.

Configuring an Authentication Provider: Main Steps

To configure an Authentication provider:

1. Expand the Security-->Realms nodes.
2. Click the name of the realm you are configuring (for example, TestRealm).
3. Expand the Providers-->Authentication Providers nodes.

   The Authenticators table displays the name of the default Authentication and Identity Assertion providers for the realm that is being configured.
4. Choose an Authentication and/or Identity Assertion provider by selecting the appropriate link:
   - Configure a new Active Directory Authenticator...
   - Configure a new Realm Adapter Authenticator...

- Configure a new Novell Authenticator...
- Configure a new iPlanet Authenticator...
- Configure a new Default Authenticator...
- Configure a new Default Identity Asserter...
- Configure a new OpenLDAP Authenticator...

5. Go to the appropriate sections to configure an Authentication and/or Identity Assertion provider.
   - [xref]
6. Repeat these steps to configure additional Authentication and/or Identity Assertion providers.

   If you are configuring multiple Authentication providers, refer to [xref to JAAS].
7. After you finish configuring Authentication and/or Identity Assertion providers, reboot WebLogic Server.

Setting the JAAS Control Flag Attribute

If a security realm has multiple Authentication providers configured, the Control Flag attribute on the Authenticator-->General tab determines the ordered execution of the Authentication providers. The values for the Control Flag attribute are as follows:

- REQUIRED—This LoginModule must succeed. Even if it fails, authentication proceeds down the list of LoginModules for the configured Authentication providers. This setting is the default.
- REQUISITE—This LoginModule must succeed. If other Authentication providers are configured and this LoginModule succeeds, authentication proceeds down the list of LoginModules. Otherwise, return control to the application.
- SUFFICIENT—This LoginModule needs not succeed. If it does succeed, return control to the application. If it fails and other Authentication providers are configured, authentication proceeds down the LoginModule list.

- OPTIONAL—This LoginModule need not succeed. Whether it succeeds or fails, authentication proceeds down the LoginModule list.

Configuring an LDAP Authentication Provider

WebLogic Server does not support or certify any particular LDAP servers. Any LDAP v2 compliant LDAP server should work with WebLogic Server. The following LDAP directory servers have been tested:

- Netscape iPlanet version 4.1.3
- Active Directory shipped as part of Windows 2000
- Open LDAP version 2.0.7
- Novell NDS version 8.5.1

For more information, see:

- "Setting LDAP Server and Caching Information" on page 3-11
- "Locating Users in the LDAP Directory" on page 3-12
- "Locating Groups in the LDAP Directory" on page 3-14
- "Locating Members of a Group in the LDAP Directory" on page 3-15

Requirements for Using an LDAP Authentication Provider

If an LDAP Authentication provider is the only configured Authentication provider for a security realm, you must have the `Admin` role to boot WebLogic Server and use a user or group in the LDAP directory. Do one of the following in the LDAP directory:

- By default in WebLogic Server, the `Admin` role includes the `Administrators` group. Create an `Administrators` group in the LDAP directory. Make sure the LDAP user from which you want to boot WebLogic Server is included in the group.

  The Active Directory LDAP directory has a default group called `Administrators`. Add the user from which you will be booting WebLogic Server to the `Administrators` group and define the Group Base Distinguished Name (DN) attribute so that the `Administrators` group is found.

- If you do not want to create an `Administrators` group in the LDAP directory (for example, because the LDAP directory uses the `Administrators` group for a different purpose), create a new group (or use an existing group) in the LDAP directory and include the user from which you want to boot WebLogic Server in that group. In the WebLogic Server Administration Console (referred to as the Administration Console), assign that group the `Admin` role.

Configuring a LDAP Authentication Provider

To configure an LDAP Authentication provider:

1. Expand the Security-->Realms nodes.
2. Click the name of the realm you are configuring (for example, TestRealm).
3. Expand the Providers-->Authentication Providers nodes.

   The Authenticators table displays the name of the default Authentication and Identity Assertion providers for the realm that is being configured.
4. Choose an LDAP Authentication provider.
   - Configure a new Active Directory Authenticator...
   - Configure a new Novell Authenticator...
   - Configure a new iPlanet Authenticator...
   - Configure a new Open LDAP Authenticator...
5. If you using multiple Authentication providers, define a value for the Control Flag attribute on the General tab. The Control Flag attribute determines how the LDAP Authentication provider is used with other LDAP Authentication providers. For more information, see [xref].

6. Click Apply to save your changes.
7. Proceed to [xref].

Setting LDAP Server and Caching Information

To set LDAP server and caching information:

1. Click the LDAP tab under the Configuration tab for the LDAP Authentication provider you want to use.

   For example, click the iPlanet LDAP tab under the iPlanet Configuration tab.

2. Enable communication between WebLogic Server and the LDAP server by defining values for the attributes shown on the LDAP tab.

   The following table describes the attributes you set on the LDAP tab.

Table 3-2 Attributes on the LDAP Tab

Attribute	Description
Host	The host name of the computer on which the LDAP server is running.
Port	The port number on which the LDAP server is listening. If you want WebLogic Server to connect to the LDAP server using the SSL protocol, use the LDAP server's SSL port in this attribute.
SSL Enabled	Option for enabling the SSL protocol to protect communications between the LDAP server and WebLogic Server. Disable this attribute if the LDAP server is not configured to use the SSL protocol.
Principal	The Distinguished name (DN) of the LDAP user used by WebLogic Server to connect to the LDAP server. Generally, this user is the system administrator of the LDAP directory server. If you want to change passwords, this attribute must be the system administrator..

Table 3-2 Attributes on the LDAP Tab

Attribute	Description
Credential	Password that authenticates the LDAP user defined in the Principal attribute.
Cache Enabled	Enables the use of a data cache with the LDAP server.
Cache Size	Maximum size of lookups in cache. The default is 32kb.
Cache TTL	Number of seconds to retain the results of an LDAP lookup.

3. To save your changes, click Apply.
4. Proceed to [xref].

For a more secure deployment, BEA recommends using the SSL protocol to protect communications between the LDAP server and WebLogic Server. For more information, see Chapter 5, "Configuring the SSL Protocol."

Locating Users in the LDAP Directory

To specify how users are located in the LDAP directory:

1. Click the Users tab under the Configuration tab for the LDAP server you chose.

   For example, click the Users tab under the iPlanet Configuration tab.

2. Define information about how users are stored and located in the LDAP directory by defining values for the attributes shown on the Users tab.

   The following table describes the attributes you set on the Users tab.

Table 3-3 Attributes on the Users Tab

Attribute	Description
User Object Class	The LDAP object class that stores users.

Table 3-3 Attributes on the Users Tab

Attribute	Description
User Name Attribute	The attribute on an LDAP user object that specifies the name of the user.
User Dynamic Group DN Attribute	The attribute of an LDAP user object that specifies the distinguished name of dynamic groups to which this user belongs. Dynamic groups are not supported with the Active Directory, Open LDAP, or Novell NDS directory servers, so set this attribute to NULL for these servers. If this attribute does not exist, WebLogic Server looks at the Dynamic Group Object Class attribute to determine the groups to which this user belongs. If a group contains other groups, WebLogic Server evaluates the URLs of any of the descendents of the group.
User Base DN	The base DN of the tree in the LDAP directory that contains users.
User Search Scope	Specifies how deep in the LDAP directory tree to search for users. Valid values are `subtree` and `onelevel`.
User from Name Filter	An LDAP search filter for finding a user given the name of the user. If a search filter is not specified (that is, if the attribute is null or empty), a default search filter is created based on the user schema. Refer to the documentation for your LDAP server for more information about writing an LDAP search filter.

Table 3-3 Attributes on the Users Tab

Attribute	Description
All Users Filter	An LDAP search filter for finding all users beneath the base DN. If a search filter is not specified (that is, if the attribute is null or empty), a default search filter is created based on the user schema. Refer to the documentation for your LDAP server for more information about writing an LDAP search filter.

3. To save your changes, click Apply.
4. Proceed to [xref].

Locating Groups in the LDAP Directory

To specify how groups are stored and located in the LDAP directory:

1. Click the Groups tab under the Configuration tab.

   For example, click the Groups tab under the iPlanet Configuration tab.

2. Define information about how groups are stored and located in the LDAP directory by defining values for the attributes shown on the Groups tab.

   The following table describes the attributes you set on the Groups tab.

Table 3-4 Attributes on the Groups Tab

Attribute	Description
Group Base DN	The base DN of the tree in the LDAP directory that stores groups.
Group Search Scope	Specifies how deep in the LDAP directory tree to search for groups. Valid values are `subtree` and `onelevel`.

Table 3-4 Attributes on the Groups Tab

Attribute	Description
Group From Name Filter	An LDAP search filter for finding a group given the name of the group. Refer to the documentation for your LDAP server for more information about writing an LDAP search filter.
All Groups Filter	An LDAP search filter for finding all groups beneath the base group DN. If the attribute is not specified (that is, if the attribute is null or empty), a default search filter is created based on the Group schema. Refer to the documentation for your LDAP server for more information about writing an LDAP search filter.
Static Group Object Class	The name of the LDAP object class that stores static groups.
Static Group Name Attribute	The attribute of a static LDAP group object that specifies the name of the group.

3. To save your changes, click Apply.
4. Proceed to [xref].

Locating Members of a Group in the LDAP Directory

Note: The iPlanet Authentication provider supports dynamic groups. To use dynamic groups, set the Dynamic Group Object Class, Dynamic Group Name Attribute, and Dynamic Member URL Attribute attributes.

To define how groups members are stored and located in the LDAP directory:

1. Click on the Membership tab under the Configuration tab.

   For example, click the Membership tab under the iPlanet Configuration tab.

2. Define information about how group members are stored and located in the LDAP directory by defining values for the attributes shown on the Membership tab.

   The following table describes the attributes you set on the Membership tab.

Table 3-5 Attributes on the Membership Tab

Attribute	Definition
Static Member DN Attribute	The attribute of an LDAP group object that specifies the DNs of the members of the group.
Static Group DNs from Member DN Filter	An LDAP search filter that, given the DN of a member of a group, returns the DNs of the static LDAP groups that contain that member. If the attribute is not specified (that is, if the attribute is null or empty), a default search filter is created based on the group schema. Refer to the documentation for your LDAP server for more information about writing an LDAP search filter.
Dynamic Group Object Class	The name of the LDAP object class that stores dynamic groups. Dynamic groups are not supported with the Active Directory, Open LDAP, or Novell NDS directory servers, so set this attribute to NUL if you are using these servers.
Dynamic Group Name Attribute	The attribute of a dynamic LDAP group object that specifies the name of the group. Dynamic groups are not supported with the Active Directory, Open LDAP, or Novell NDS directory servers, so set this attribute to NUL if you are using these servers.
Dynamic Member URL Attribute	The attribute of the dynamic LDAP group object that specifies the URLs of the members of the dynamic group. Dynamic groups are not supported with the Active Directory, Open LDAP, or Novell NDS directory servers, so set this attribute to NUL if you are using these servers.

3. To save your changes, click Apply.
4. Optionally, configure additional Authentication and/or Identity Assertion providers.
5. Reboot WebLogic Server.

Configuring a WebLogic Authentication Provider

Note: The Administration Console refers to the WebLogic Authentication provider as the Default Authenticator.

The WebLogic Authentication provider is case insensitive. Ensure user names are unique.

The WebLogic Authentication provider allows you to edit, list, and manage users and group membership. User and group membership information for the WebLogic Authentication provider is stored in the embedded LDAP server.

To configure the WebLogic Authentication provider:

1. Configure the embedded LDAP server as described in "Configuring the Embedded LDAP Server" on page 2-19.

1. Expand the Security-->Realms nodes.
2. Click the name of the realm you are configuring (for example, TestRealm).
3. Expand the Providers-->Authentication Providers nodes.

   The Authenticators table displays the name of the default Authentication and Identity Assertion providers for the realm being configured.
4. Choose the Configure a new Default Authenticator... link.
5. Define values for the attributes on the General tab.
   - The Minimum Password Length attribute applies to the passwords you specify when defining users in the WebLogic Authentication provider.

- The Control Flag attribute determines how the WebLogic Authentication provider is used with other Authentication providers. For more information, see [xref].

6. Click Apply to save your changes.
7. Optionally, configure additional Authentication and/or Identity Assertion providers.
8. Reboot WebLogic Server.

Configuring a Realm Adapter Authentication Provider

The Realm Adapter Authentication provider allows you to use users and groups from 6.x security realms with the WebLogic security providers in WebLogic Server version 7.0. Use the Realm Adapter Authentication provider if you store users and groups in the 6.x Windows NT, UNIX, RDBMS or custom security realms. (There are no equivalents to the 6.x Windows NT, UNIX, RDBMS security realms in WebLogic Server 7.0). When using Compatibility Security, a Realm Adapter Authentication provider is by default configured for the Compatibility realm.

The Realm Adapter Authentication provider also allows you to use implementations of the `weblogic.security.acl.CertAuthenticator` class with WebLogic Server 7.0. The Realm Adapter Authentication provider includes an Identity Assertion provider which provides identity assertion based on X.509 tokens. For information about using a CertAuthenticator with WebLogic Server 7.0, see Using a CertAuthenticator in Compatibility Security in the *Upgrade Guide for BEA WebLogic Server 7.0*.

Note: The Subjects produced by the Realm Adapter Authentication provider do not contain Principals for the groups to which a user belongs. Use the `weblogic.security.SubjectUtils.isUserInGroup()` method to determine whether a user is in a group. When using Subjects produced by the Realm Adapter Authentication provider there is no way to iterate the complete set of groups to which a user belongs.

To configure the Realm Adapter Authentication provider to access groups and users from 6.x security realms:

1. Boot WebLogic Server to run Compatibility Security. For more information, see [xref Upgrade].
2. Ensure that the Realm Adapter Authentication provider in the Compatibility realm is populated with users and groups from the 6.x security realm (Windows NT, UNIX, RDBMS or Custom security realms). The users and groups should appear in the Users and Groups table. The user and group information is copied into a `filerealm.properties` file.
3. Expand the Security-->Realms nodes.
4. Click the name of the realm you are configuring (for example, TestRealm).
5. Expand the Providers-->Authentication Providers nodes.

   The Authenticators table displays the name of the default Authentication and Identity Assertion providers for the realm that is being configured.
6. Choose the Configure a new Realm Adapter Authenticator... link.
7. Define values for the attributes on the General tab.
   - The Control Flag attribute determines how the Realm Adapter Authentication provider is used with other Authentication providers. Set the Control Flag attribute to OPTIONAL.
   - The Supported Types attribute lists the type of tokens supported by the Identity Assertion provider. The Identity Assertion provider in the Realm Adapter Authentication provider supports only X.509 tokens. This token type provides backward compatibility for implementations of the `weblogic.security.acl.CertAuthenticator` class. This attribute is read-only.
   - The Active Type attribute defines what type of token is currently being used by the Identity Assertion provider. Multiple Identity Assertion providers in a security realm can support the same token type. However, only one Identity Assertion provider in a security realm can have a particular token type active. If two Identity Assertion providers have the same type of token defined in the Active Type attribute, the first Identity Assertion provider configured in the realm handles that type of token. The Identity Assertion provider in the Realm Adapter Authenticator supports X.509 token types.

8. Click Apply to save your changes.
9. Set the Control Flag attribute on the WebLogic Authentication provider to SUFFICIENT.
10. Optionally, configure additional Authentication and/or Identity Assertion providers.
11. Reboot WebLogic Server.

Configuring a WebLogic Identity Assertion Provider

Note: The Administration Console refers to the WebLogic Identity Assertion provider as the Default Identity Asserter.

An Identity Assertion provider verifies the tokens and performs whatever actions are necessary to establish validity and trust in the token. Each Identity Assertion provider is designed to handle one or more token formats. If you are using perimeter authentication, you need to use an Identity Assertion provider. In perimeter authentication, a system outside of WebLogic Server establishes trust via tokens (as opposed to simple authentication, where WebLogic Server establishes trust via usernames and passwords).

You can use either a WebLogic Identity Assertion provider or a Custom Identity Assertion provider in a security realm. This section describes how to configure a WebLogic Identity Assertion provider. For information about configuring a custom security provider (including a Custom Identity Assertion provider), see "Configuring a Custom Security Provider" on page 3-25.

The WebLogic Identity Assertion provider supports identity assertion using X509 certificates and CORBA Common Secure Interoperability version 2 (CSI v2). When you finish defining attributes for the WebLogic Identity Assertion provider, reboot WebLogic Server.

To define attributes for the WebLogic Identity Assertion provider:

1. Expand the Security-->Realms nodes.

2. Click the name of the realm you are configuring (for example, TestRealm).
3. Expand the Providers-->Authentication Providers nodes.

   The Authenticators table displays the name of the default Authentication and Identity Assertion providers for the realm that is being configured.
4. Choose the Configure a new Default Identity Asserter... link from the Authenticators tab.
5. Define values for the attributes on the General tab.

   - The User Name Mapper Class Name attribute specifies your implementation of the `weblogic.security.providers.authentication.UserNameMapper` interface. This interface maps a certificate to a user name according to whatever scheme is appropriate for your needs. You can also use this interface to map from a X.501 distinguished name to a user name.

Note: The implementation of the `weblogic.security.providers.authentication.UserNameMapper` interface must be specified in your CLASSPATH.

   - The Trusted Client Principals attribute defines a list of client principals that can use CSIv2 identity assertion. You can use an asterisk (`*`) to specify all client principals.

     When you use client certificates with the SSL protocol, there is trust associated with a certificate. However, because CSIv2 contains no proof, there is no trust associated with any of the CSIv2 identity assertion tokens. A malicious user could create CSIv2 tokens for any user. This attribute provides a way to limit the set of principals that can assert identity via CSIv2.
6. Click Apply to save your changes.
7. Optionally, configure additional Authentication and/or Identity Assertion providers.
8. Reboot WebLogic Server.

Configuring a WebLogic Authorization Provider

Authorization is the process whereby the interactions between users and resources are limited to ensure integrity, confidentiality, and availability. In other words, authorization is responsible for controlling access to resources based on user identity or other information.

You can use either a WebLogic Authorization provider or a Custom Authorization provider in a security realm. This section describes how to configure a WebLogic Authorization provider. For information about configuring a custom security provider (including a Custom Authorization provider), see "Configuring a Custom Security Provider" on page 3-25.

Note: The Administration Console refers to the WebLogic Authorization provider as the Default Authorizer.

To configure a WebLogic Authorization provider:

1. Expand the Security-->Realms nodes.
2. Click the name of the realm you are configuring (for example, TestRealm).
3. Expand the Providers node.
4. Click Authorizers.

   The Authorizers table displays the name of the default Authorization provider for the realm that is being configured.
5. Click the Configure a new Default Authorizer... link.
6. Define values for the attributes on the General tab.

   The Policy Deployment Enabled attribute specifies whether or not this Authorization provider stores policy information (as opposed to retrieving policy information from a deployment descriptor) for the security realm. In order to support the Policy Deployment Enabled attribute, an Authorization provider must implement the `DeployableAuthorizationProvider` Security Service Provider Interface (SSPI). By default, the WebLogic Authorization provider has

this attribute enabled. The policy information is stored in the embedded LDAP server.

For more information, see The Components of an Authorization Provider in *Developing Security Services for WebLogic Server.*

7. Click Apply to save your changes.
8. Reboot WebLogic Server.

Configuring a WebLogic Credential Mapping Provider

Credential mapping is the process whereby the authentication and authorization mechanisms of a remote system (for example, a legacy system or application) are used to obtain an appropriate set of credentials to authenticate users to a target WebLogic resource.

For more information about credential maps and their use in resource adapters, see "Providing WebLogic Server Users Access to Other Applications" on page 2-9 and the Security topic in *Programming the J2EE Connector Architecture*.

You can use either a WebLogic Credential Mapping provider or a Custom Credential Mapping provider in a security realm. This section describes how to configure a WebLogic Credential Mapping provider. For information about configuring a custom security provider (including a Custom Credential Mapping provider), see "Configuring a Custom Security Provider" on page 3-25.

To configure a WebLogic Credential Mapping provider:

1. Expand the Security-->Realms nodes.
2. Click the name of the realm you are configuring (for example, TestRealm).
3. Expand the Providers node.
4. Click Credential Mappers.

   The Credential Mappers table displays the name of the default Credential Mapping provider for the realm that is being configured.

5. Click the Configure a new Default Credential Mapper... link.
6. On the General tab, set the Credential Mapping Deployment Enabled attribute.

   The Credential Mapping Deployment Enabled attribute specifies whether or not this Credential Mapping provider imports credential maps from a 6.x Resource Adapter Archive (RAR). In order to support the Credential Mapping Deployment Enabled attribute, a Credential Mapping provider must implement the `DeployableCredentialProvider` SSPI. By default, the WebLogic Credential Mapping provider has this attribute enabled. The credential mapping information is stored in the embedded LDAP server.

   For more information, see Implementing the DeployableCredentialMappingProvider SSPI in *Developing Security Services for WebLogic Server*.
7. Click Apply to save your changes.
8. Reboot WebLogic Server.

Configuring a WebLogic Role Mapping Provider

Role Mapping providers compute the set of roles granted to a subject for a given resource. Role Mapping providers supply Authorization providers with this role information so that the Authorization Provider can answer the “is access allowed?” question for WebLogic resources.

You can use either a WebLogic Role Mapping provider or a Custom Role Mapping provider in a security realm. This topic describes how to configure a WebLogic Role Mapping provider. For information about configuring a custom security provider (including a Custom Role Mapping provider), see “Customizing the Default Security Configuration” on page 3-1.

To configure an Role Mapping provider:

1. Expand the Security node.
2. Expand the Realms node.

3. Click the name of the realm you are configuring (for example, TestRealm).
4. Click the Providers node.
5. Click Role Mappers.

   The Role Mappers table appears. This table displays the name of the default Role Mapping provider for the realm that is being configured.
6. Click the Configure a new Default Role Mapper... link.

   The General tab appears.
7. Define values for the attributes on the General tab.

   The Role Mapping Deployment Enabled attribute specifies whether or not this Role Mapping provider imports information from deployment descriptors for Web applications and EJBs into the security realm. In order to support the Role Mapping Deployment Enabled attribute, a Role Mapping provider must implement the `DeployableRoleProvider` SSPI. By default, the WebLogic Role Mapping provider has this attribute enabled. Roles are stored in the embedded LDAP server.

   For more information, see The Role Mapping Providers in *Developing Security Services for WebLogic Server.*
8. Click Apply to save your changes.
9. Reboot WebLogic Server.

Configuring a Custom Security Provider

To configure a Custom security provider:

1. Write a Custom security provider. For more information, see *Developing Security Providers for WebLogic Server.*
2. Put the MBean JAR file for the provider in the `WL_HOME\lib\mbeantypes` directory.
3. Start the Administration Console.

4. Expand the Security-->Realms nodes.
5. Click on the name of the realm you are configuring (for example, TestRealm.)
6. Expand the Providers node.
7. Expand the node for the type of provider you are configuring. For example, expand the Authenticator node to configure a Custom Authentication provider.

   The tab for the provider appears.
8. Click the Configure a new Custom *Security_Provider_Type*... link

   where *`Security_Provider_Type`* is the name of your custom security provider. This name is read from the `DisplayName` attribute in the *`MBeanType`* tag of the MBean Definition File (MDF).
9. The General tab appears.

   The Name attribute displays the name of your Custom Security provider.
10. If desired, adjust the values for the attributes for the Custom Security provider.
11. Click Apply to save your changes.
12. Reboot WebLogic Server.

Deleting a Security Provider

To delete a security provider:

1. Expand the Security-->Realms nodes.
2. Click the name of the realm in which the provider you want to delete is configured (for example, TestRealm).
3. Expand the Providers node.
4. Click the type of provider you want to delete (for example, TestRealm-->Authorizers).
5. The table page for the provider appears (for example, the Authorizers table). The table page for the provider displays the names of all the available providers.

6. To delete a provider, click on the trash can icon in the corresponding row of the provider table.
7. Reboot WebLogic Server.

Note: The WebLogic Securiy providers stored their data in the embedded LDAP server. When you delete a WebLogic Security provider, the security data in the embedded LDAP server is not automatically deleted. The security data remains in the embedded LDAP server in case you want to use the provider again. Use an external LDAP browser to delete the security data from the embedded LDAP server.

Creating a New Security Realm: Main Steps

To configure a new security realm:

1. Expand the Security node.
2. Expand the Realms node.

   All the security realms available for the WebLogic domain are listed in the Realms table.
3. Click the Configure a new Realm... link.
4. Enter the name of the new security realm in the Name attribute on the General tab.
5. Set the Ignore Security Data in Deployment Descriptors attribute as desired. For more information, see [xref].
6. Click Create.
7. Configure the required security providers for the security realm. In order for a security realm to be valid, you must configure an Authentication provider, an Authorization provider, an Adjudication provider, a Credential Mapping provider, and a Role Mapping provider. Otherwise, you will not be able to set the new security realm as the default security realm.
8. Optionally, define an Identity Assertion and Auditing provider.

9. Define groups and users for the security realm. [xref]
10. Grant users and groups in the security realm roles. [xref]
11. Protect WebLogic resources in the security realm with security policies.
12. Reboot WebLogic Server. If you do not reboot WebLogic Server, you cannot set the realm to the default security realm.
13. Set the new realm as the default security realm for the WebLogic domain. [xref]

Loading Security Data from Deployment Descriptors into the Security Providers

On application deployment, WebLogic Server reads security and credential information from the `weblogic.xml`, `weblogic-ejb-jar.xml`, and `weblogic-ra.xml` files. Once the security and credential information is loaded into the Administration Console, security and credential mapping changes are not persisted to the `weblogic.xml`, `weblogic-ejb-jar.xml`, and `weblogic-ra.xml` files.

Before you redeploy the application (which will happen when you redeploy it through the Administration Console, modify it on disk, or restart WebLogic Server), you need to enable the Ignore Security Data in Deployment Descriptors attribute on the Security-->Realm General tab. Otherwise, the old data in the `weblogic.xml`, `weblogic-ejb-jar.xml`, and `weblogic-ra.xml` files will overwrite any changes made through the Administration Console.

To avoid overwriting Administration Console changes with old information from the `weblogic.xml`, `weblogic-ejb-jar.xml`, and `weblogic-ra.xml` files:

1. Expand the Security-->Realms nodes.
2. Click the name of the realm (for example, TestRealm).
3. In the General tab, enable the Ignore Security Data in Deployment Descriptors attribute.
4. Click Apply to save your changes.

5. Reboot WebLogic Server.

Changing the Default Security Realm

By default, WebLogic Server sets the myrealm as the default security realm.

To change the default security realm:

1. Configure a new security realm. For more information, see [xref].
2. Reboot WebLogic Server.
3. Expand the Domain node.
4. Select the Security-->General tab.

   The pull-down menu on the Default Realm attribute displays the security realms configured in the WebLogic domain.

   Note: If you create a new security realm but do not configure the required security providers, the realm will not be available from the pull-down menu.
5. Select the security realm you want to set as the default security realm.
6. Click Apply.
7. Reboot WebLogic Server. If you not reboot WebLogic Server, the new realm is not set as the default security realm.

To verify you set the default security realm correctly:

1. Expand the Security-->Realms nodes.

   The General tab shows all realms configured for the WebLogic domain. The default security realm has the Default Realm attribute set to `true`.

After you set a realm as the default security realm, complete the process of configuring security by performing the tasks in Chapter 2, "Configuring WebLogic Security."

Deleting a Security Realm

When you delete a security realm, the user, group, role, security policy, and credential map information is not deleted from the embedded LDAP server.

To delete a security realm:

1. Expand the Security-->Realms nodes.

   The Realm table lists all realms configured for the WebLogic domain.

2. To delete a security realm, click the trash can icon in the corresponding row of the Realm table.

3. Click Yes in response to the following question:

   `Are you sure you want to permanently delete` ***`OldRealm`*** `from the domain configuration?`

   A confirmation message appears when the security realm is deleted.

4 Using Compatibility Security

The following sections describe how to set up Compatibility Security and manage an existing 6.x security configuration:

Note: Compatibility security is deprecated in WebLogic Server 7.0. You should only use Compatibility security while upgrading your WebLogic Server deployment to the security features in WebLogic Server 7.0.

Setting Up Compatibility Security: Main Steps

To set up Compatibility security:

1. Make a back-up copy of your 6.x WebLogic domain (including your `config.xml` file) before using Compatibility security. A sample `config.xml` file that can be used to boot Compatibility Security can be found in Booting WebLogic Server in Compatibility Security in the *Upgrade Guide for BEA WebLogic Server 7.0*.
2. Install WebLogic Server 7.0 in a new directory location. Do not overwrite your existing 6.x installation directory. For more information, see the *WebLogic Server Installation Guide*.
3. Modify the startup script for your 6.x server to point to the WebLogic Server 7.0 installation. Specifically, you need to modify:
   - The classpath to point to the `weblogic.jar` file in the WebLogic Server version 7.0 installation.
   - The `JAVA_HOME` variable to point to the WebLogic Server version 7.0 installation.

   For more information, see the *Upgrade Guide for BEA WebLogic Server 7.0*.
4. Use the startup script for your 6.x server to boot WebLogic Server.

To verify whether you are correctly running Compatibility security, do the following:

1. In the Administration Console, expand the Domain node.
2. Click on your WebLogic Server domain (referred to as the domain).
3. Click the View the Domain Log link.

   The following message appears in the log:

   ```
   Security initializing using realm CompatibilityRealm
   ```

Changing the System Password

During installation WebLogic Server does the following to the File realm in *mydomain*:

1. Adds the username and password supplied during installation to the File realm.
2. Sets the system password to password specified during installation.

These steps ensure that a `system` user is defined in the 7.0 version of the File realm and that the `SerializedSystemIni.dat` file is created.

To improve security, BEA recommends frequently changing the system password that was set during installation. Each WebLogic Server deployment must have a unique password.

1. In the console for the Administration Server, expand the Compatibility Security node.
2. Select the Users tab.
3. In the User Configuration window, under Change a User's Password, enter `system` in the Name attribute.
4. In the Old Password attribute, enter password you specified when installing WebLogic Server 6.x..
5. Enter a new password in the New Password attribute.
6. Enter the new password again in the Confirm the Password attribute.

When you use an Administration Server and Managed Servers in a domain, the Managed Server must always use the password for the Administration Server in the domain. Always change the password for the Administration Server through the Administration Console. The new password is propagated to all the Managed Servers in the domain.

Maintaining the secrecy of WebLogic passwords is critical to keeping your WebLogic Server deployment and data secure. For your protection, BEA recommends keeping the password of WebLogic Server secret.

Specifying a Compatibility Security Realm

The following sections describes configuring a security realm for your WebLogic Server deployment:

- Configuring the File Realm
- Configuring the Caching Realm
- Configuring the LDAP V1 Security Realm
- Configuring the Windows NT Security Realm
- Configuring the UNIX Security Realm
- Configuring the RDBMS Security Realm
- Installing a Custom Security Realm

Note: The File realm, Caching realm, LDAP, Windows NT, UNIX, and RDBMS security realms are deprecated in WebLogic Server 7.0. In addition, the use of WebLogic Server 6.x custom security realms is deprecated in WebLogic Server 7.0.

Configuring the File Realm

By default WebLogic Server is installed with the File realm in place. The File Realm stores user and group data for the purpose of Authentication. Before using the File realm, you need to define several attributes that govern the use of the File realm.

To configure the File realm:

1. Expand the Domains node.
2. Select the Security-->File Realm tab.
3. Enter values in the attribute fields on the File Realm tab.

   The following table describes each attribute on the File Realm tab.

Table 4-1 File Realm Attributes

Attribute	Description
Caching Realm	The File realm does not use a Caching realm. Set this attribute to None.
Max Users	Maximum number of users in the File realm. The minimum value for this attribute is 1 and the maximum value is 10,000. The default is 1,000.
Max Groups	Maximum number of Groups to be used with the File realm**.** The minimum value for this attribute is 1 and the maximum value is 10,000. The default is 1,000.
Max ACLs	Maximum number of access control lists (ACLs) to be used with the File realm. The minimum value for this attribute is 1 and the maximum value is 10,000. The default is 1,000.

4. Click Apply to save your changes.

All user and group data for the File realm is stored in the `fileRealm.properties` file. If the `fileRealm.properties` file becomes corrupted or is destroyed, you must reconfigure the security information for WebLogic Server. Compatibility security cannot run without a `fileRealm.properties` file. Even if you write a custom security realm, you still need a `fileRealm.properties` file to boot WebLogic Server because the custom security realm is not initially called during the start-up sequence. Therefore, BEA recommends that you take the following steps:

1. Make a backup copy of the `fileRealm.properties` file and put it in a secure place.
2. Set the permissions on the `fileRealm.properties` file such that the administrator of the WebLogic Server deployment has write and read privileges and no other users have any privileges.

Note: Also make a backup copy of the `SerializedSystemIni.dat` file for the File realm. For more information about the `SerializedSystemIni.dat` file, see "Protecting User Accounts" on page 4-36.

If, instead of the File realm, you want to use one of the alternate security realms provided by WebLogic Server, or use a custom security realm, set the attributes for the desired realm and reboot WebLogic Server. If you use one of the alternate security realms, you must enable the Caching realm. Alternate security realms include the LDAP, Windows NT, UNIX, and RDBMS security realms

Configuring the Caching Realm

The Caching realm works with alternate security realms, or custom security realms to fulfill client requests with the proper authentication and authorization. The Caching realm stores the results of both successful and unsuccessful realm lookups. It manages separate caches for users, groups, permissions, ACLs, and authentication requests. The Caching realm improves the performance of WebLogic Server by caching lookups, thereby reducing the number of calls into other security realms.

The Caching realm is installed automatically when you run Compatibility security: the cache is set up to delegate to the other security realms; however, caching is not enabled. You have to enable caching through the Administration Console. If you are using an alternate security realm or a custom security realm, you must configure and enable the Caching realm after you configure an alternate or custom security realm.

When you enable caching, the Caching realm saves the results of a realm lookup in its cache. Lookup results remain in the cache until either the specified number of seconds defined for the time-to-live (TTL) attributes has passed (the lookup result has expired) or the cache has filled. When the cache is full, new lookup results replace the oldest cached results. The TTL attributes determine how long a cached object is valid. The higher you set these attributes, the less often the Caching realm calls the secondary security realm. Reducing the frequency of such calls improves the performance. The trade-off is that changes to the underlying security realm are not recognized until the cached object expires.

Note: When you obtain an object from a security realm, the object reflects a snapshot of the object. To update the object, call the object's `get()` method again. For example, the membership of a group is set when the group is retrieved from the security realm with a call to the `getGroup()` method. To update the members of the group, you must call the `getGroup()` method again.

By default, the Caching realm operates on the assumption that the alternate security realm is case-sensitive. In a case-sensitive security realm, the owners of usernames `bill` and `Bill`, for example are treated as two distinct users. The Windows NT Security realm and the LDAP Security realm are examples of security realms that are not case-sensitive. If you are using a security realm that is not case-sensitive, you must disable the CacheCaseSensitive attribute. When this attribute is set, the Caching realm converts usernames to lowercase so that WebLogic Server gives correct results for the security realm when it performs case-sensitive comparisons. When defining or referencing users or groups in a case-sensitive security realm, type usernames in lowercase.

To configure the Caching realm:

1. Configure the alternate or custom security realm with which you will use the Caching realm. See the appropriate realm configuration procedures in the following sections:
   - "Configuring the LDAP V1 Security Realm" on page 4-11
   - "Configuring the Windows NT Security Realm" on page 4-18
   - "Configuring the UNIX Security Realm" on page 4-22
   - "Configuring the RDBMS Security Realm" on page 4-25
   - "Installing a Custom Security Realm" on page 4-29
2. Expand the Compatibility Security-->Caching Realms nodes.
3. Click the Configure a new Caching Realm... link.
4. Enter values in the attribute fields on the General tab.

   The following table describes the attributes you set on the General tab.

Table 4-2 Caching Realm Attributes on the General Tab

Attribute	Description
Name	Displays the active security realm as defined in the Administration Console. This attribute can not be changed.
Basic Realm	Name of the alternate security realm or custom security realm to be used with the Caching realm. The names of the configured realms appear on the pull-down menu.
Case Sensitive Cache	Defines whether the specified security realm is case-sensitive. By default, this attribute is enabled: the realm is case-sensitive. To use a realm that is not case-sensitive (such as the Windows NT and LDAP security realms), you must disable this attribute.

5. Click Create.

6. Configure and enable the ACL cache by defining values for the attributes shown on the ACL tab.

   The following table describes the attributes you set on the ACL tab.

Table 4-3 ACL Cache Attributes

Attribute	Description
Enable Cache	Option for enabling the ACL cache.
ACL Cache Size	Maximum number of ACL lookups to cache. This attribute should be a prime number for best lookup performance. The default is 211.
ACL Cache Positive TTL	Number of seconds to retain the results of a successful lookup. The default is 60 seconds.
ACL Cache Negative TTL	Number of seconds to retain the results of an unsuccessful lookup. The default is 10 seconds.

7. Click Apply to save your changes.
8. Configure and enable the Authentication cache by defining values for the attributes shown on the Authentication tab.

   The following table describes the attributes you set on the Authentication tab.

Table 4-4 Authentication Cache Attributes

Attribute	Description
Enable Authentication Cache	Option for enabling the Authentication cache.
Authentication Cache Size	Maximum number of Authenticate requests to cache. This attribute should be a prime number for best lookup performance. The default is 211.

Table 4-4 Authentication Cache Attributes

Attribute	Description
Authentication Cache TTLPositive	Number of seconds to retain the results of a successful lookup. The default is 60 seconds.
Authentication Cache TTLNegative	Number of seconds to retain the results of an unsuccessful lookup. The default is 10 seconds.

9. Click Apply to save your changes.
10. Configure and enable the Group cache by defining values for the attributes shown on the Groups tab.

    The following table describes the attributes you set on the Group tab.

Table 4-5 Group Cache Attributes

Attribute	Description
Enable Group Cache	Option for enabling the Group cache.
Group Cache Size	Maximum number of Group lookups to cache. This attribute should be a prime number for best lookup performance. The default is 211.
Group Cache TTLPositive	Number of seconds to retain the results of a successful lookup. The default is 60 seconds.
Group Cache TTLNegative	Number of seconds to retain the results of an unsuccessful lookup. The default is 10 seconds.
Group Membership Cache TTL	Number of seconds to store the members of a group before updating it. The default is 300 seconds.

11. Click Apply to save your changes.

12. Configure and enable the User cache by defining values for the attributes shown on the Users tab.

    The following table describes the attributes you set on the User tab.

Table 4-6 User Cache Attributes

Attribute	Description
Enable User Cache	Option for enabling the User cache.
User Cache Size	Maximum number of user lookups to cache. This attribute should be a prime number for best lookup performance. The default is 211.
User Cache TTLPositive	Number of seconds to retain the results of a successful lookup. The default is 60 seconds.
User Cache TTLNegative	Number of seconds to retain the results of an unsuccessful lookup. The default is 10 seconds.

13. Click Apply to save your changes.

14. Configure and enable the Permission cache by defining values for the attributes shown on the Permissions tab.

    The following table describes each attribute on the Permission tab.

Table 4-7 Permission Cache Attributes

Attribute	Description
Enable Permission Cache	Option for enabling the Permission cache.
Permission Cache Size	Maximum number of Permission lookups to cache. This attribute should be a prime number for best lookup performance. The default is 211.
Permission Cache TTLPositive	Number of seconds to retain the results of a successful lookup. The default is 60 seconds.

Table 4-7 Permission Cache Attributes

Attribute	Description
Permission Cache TTLNegative	Number of seconds to retain the results of an unsuccessful lookup. The default is 10 seconds.

15. Click Apply to save your changes.
16. When you finish defining attributes for the Caching realm, reboot WebLogic Server.

Configuring the LDAP V1 Security Realm

The Lightweight Directory Access Protocol (LDAP) V1 security realm provides authentication through an LDAP server. This server allows you to manage all the users for your organization in one place: the LDAP directory. The LDAP V1 security realm supports Open LDAP, Netscape iPlanet, Microsoft Site Server, and Novell NDS.

Configuring the LDAP V1 security realm involves defining attributes that enable the LDAP V1 security realm in WebLogic Server to communicate with the LDAP server and attributes that describe how users and groups are stored in the LDAP directory. The LDAP tree and schema is different for every LDAP server. Therefore, the LDAP V2 security realm provides a set of templates that define default attributes for the supported LDAP servers.

The Difference between the LDAP V1 and LDAP V2 Security Realms

Choose between two versions of the LDAP security realm:

- LDAP V1 security realm—The LDAP security realm packaged in releases of WebLogic Server before WebLogic Server 6.0 SP01. With the exception of Microsoft Site Server, the LDAP V1 security realm works with all the supported LDAP servers and is provided for BEA customers that are currently using the LDAP security realm in an older release of WebLogic Server. However, the

LDAP V1 security realm is deprecated in this release and BEA recommends users upgrade to the LDAP V2 security realm.

- LPAP realm V2—An updated LDAP security realm with improved performance and configurability. This is the same LDAP security realm provided in WebLogic Server 6.0 Service Pack 1.0. When running Windows 2000, BEA recommends using LDAP V2 security realm to authenticate against the Windows 2000 User and Group store.

Note: LDAP V1 security realm allows you to view users and members of a group stored in the LDAP directory server through the Administration Console. However, with LDAP V2 security realm, you can only view the groups stored in the LDAP directory server through the Administration Console.

You need to use the administration tools available with the LDAP server to manage users and groups (for example, adding or deleting users or groups or adding members to groups). If you make a change in the LDAP directory store, reset the User cache and the Group cache to immediately view your changes in the Administration Console.

LDAP Realm Performance Tips

To improve LDAP Security realm performance:

- Have the LDAP server index all of the attributes that you use as search keys in your LDAP realm search filters. Not indexing the attributes could cause linear search performance.
- Use the Caching realm carefully. Changes in the LDAP server's information are not propagated to the LDAP Security realm until the cache is cleared.

Restrictions When Using the LDAP Security Realm

The LDAP security realm has the following restrictions:

- When the LDAP server in Microsoft Site Server is installed and the root of the LDAP directory is created, a number of organizational units are created by default. Under groups, a default organization unit called `NTGroups` has a default group named `Administrators`, which is empty. By default, WebLogic Server also provides a group called `Administrators` that contains a member `system`,

which is the user under which WebLogic Server is started. If you use the defaults in Microsoft Site Server and start creating your own groups under the default organizational unit, WebLogic Server will not start. In order to start WebLogic Server with the LDAP security realm, you need to create your own unique organizational unit in the LDAP directory and create groups for your WebLogic Server deployment under that organizational unit.

- If you have two groups within the LDAP directory with the same name, WebLogic Server cannot properly authenticate the users in the second group that it locates. The LDAP security realm uses the group's distinguished name (DN) to locate groups in the LDAP directory. If you create more than one group with the same name, WebLogic Server only authenticates the users in the first group it locates. You must use unique group names when using the LDAP security realm.
- The LDAP V2 security realm does not provide the following functionality provided in LDAP V1 security realm:
  - Listing all users
  - Listing the members of a group
  - The authProtocol and userAuthentication mechanisms have been removed. You need to use the JNDI bind mechanism to pass security credentials to the LDAP server. For more information, see Setting Up JNDI Environment Properties for the Initial Context in *Programming with WebLogic JNDI.*

Configuring an LDAP V1 Security Realm

To use the LDAP V1 security realm instead of the File realm:

1. Expand the Compatibility Security-->Realms nodes.
2. Click on Configure a New LDAP Realm V1 link to display the name of the class that implements the LDAP V1 security realm.
3. Click Create.
4. To enable communication between the LDAP server and WebLogic Server, define values for the attributes on the LDAP Server tab.

   The following table describes the attributes you set on the LDAP Realm V1 tab.

Table 4-8 LDAP V1 Security Realm Attributes on the LDAP Tab

Attribute	Description
LDAPURL	Location of the LDAP server. Change the URL to the name of the computer on which the LDAP server is running and the number of the port at which it is listening. For example: `ldap://ldapserver:385`. If you want WebLogic Server to connect to the LDAP server using the SSL protocol, use the LDAP server's SSL port in the URL.
Principal	Distinguished name (DN) of the LDAP User used by WebLogic Server to connect to the LDAP server. This user must be able to list LDAP users and groups.
Credential	Password that authenticates the LDAP User defined in the Principal attribute.
Enable SSL	Option for enabling the SSL protocol to protect communications between the LDAP server and WebLogic Server. Keep in mind the following guidelines: ■ Disable this attribute if the LDAP server is not configured to use the SSL protocol. ■ If you set the UserAuthentication attribute on the LDAP Users tab to `external`, this attribute must be enabled.
AuthProtocol	The type of authentication used to authenticate the LDAP server. Set this attribute to one of the following values: ■ `None` for no authentication ■ `Simple` for password authentication ■ `CRAM-MD5` for certificate authentication Netscape iPlanet supports `CRAM-MD5`. Microsoft Site Server, Netscape iPlanet, and OpenLDAP and Novell NDS support `Simple`.

5. Click Apply to save your changes.

6. To specify how users are stored in the LDAP V1 security realm, define the attributes shown on the Users tab.

   The following table describes the attributes you set on the Users tab.

Table 4-9 LDAP V1 Security Realm Attributes on the Users Tab

Attribute	Description
User Authentication	Determines the method for authenticating users. Set this attribute to one of the following values: ■ `Bind` specifies that the LDAP security realm retrieves user data, including the password for the LDAP server, and checks the password in WebLogic Server. ■ `External` specifies that the LDAP security realm authenticates a user by attempting to bind to the LDAP server with the username and password supplied by the WebLogic Server client. If you choose the `External` setting, you must also use the SSL protocol. ■ `Local` specifies that the LDAP security realm authenticates a user by looking up the UserPassword property in the LDAP directory and checking it against the passwords in WebLogic Server.
User Password Attribute	If the User Authentication attribute is set to Local, this attribute is used to find out what LDAP property contains the passwords for the LDAP users.
User DN	A list of attributes and their values that, when combined with the attributes in the `User Name Attribute` attribute, uniquely identifies an LDAP user.
User Name Attribute	The login name of the LDAP user. The value of this attribute can be the common name of an LDAP user but usually it is an abbreviated string, such as the common name.

7. Click Apply to save your changes.

8. To specify how Groups are stored in the LDAP directory, define the attributes shown on the Groups tab.

   The following table describes the attributes you set on the Groups tab.

Table 4-10 LDAP V1 Security Realm Attributes on the Groups Tab

Attribute	Description
Group DN	List of attributes and values that, combined with the Group Name Attribute attribute, uniquely identifies a group in the LDAP directory.
Group Name Attribute	Name of a group in the LDAP directory. It is usually a common name.
Group Is Context	Boolean checkbox that specifies how group membership is recorded in the LDAP directory. ■ Check this checkbox if each Group entry contains one user. By default, the box is selected. ■ Uncheck this checkbox if one Group entry contains an attribute for each group member.
Group Username Attribute	Name of the LDAP attribute that contains a group member in a Group entry.

9. Click Apply to save your changes.
10. When you have finished defining all the attributes, reboot WebLogic Server.
11. Configure the Caching realm as described on "Configuring the Caching Realm" on page 4-6.

    When configuring the Caching realm, select the LDAP Realm V1 from the pull-down menu for the Basic Realm attribute on the General tab. The Basic Realm attribute defines the association between the Caching realm and the alternate security realm (in this case, the LDAP V1 security realm).
12. Expand the Domains node.
13. Select the Security-->File Realm tab.
14. In the Caching Realm attribute, choose the name of the Caching realm to be used with the LDAP V1 security realm. A list of configured Caching realms appears on the pull-down menu.

15. Reboot WebLogic Server.

The Caching realm caches users and groups internally to avoid frequent lookups in the LDAP directory. Each object in the users and groups caches has a TTL attribute that you set when you configure the Caching realm. If you make changes in the LDAP directory, those changes are not reflected in the LDAP V1 security realm until the cached object expires or is flushed from the cache. The default TTL is 10 seconds for unsuccessful lookups and 60 seconds for successful lookups. Unless you change the TTL attributes for the User and Group caches when you configured the Caching realm, changes in the LDAP directory should be reflected in the LDAP V1 security realm in 60 seconds. For more information about TTL attributes, see "Configuring the Caching Realm" on page 4-6.

If some server-side code has performed a lookup in the LDAP V1 security realm, such as a `getUser()` call on the LDAP V1 security realm, the object returned by the realm cannot be released until the code releases it. Therefore, a user authenticated by WebLogic Server remains valid as long as the connection persists, even if you delete the user from the LDAP directory.

Configuring an LDAP V2 Security Realm

When using the LDAP V2 security realm, WebLogic Server provides templates for the supported LDAP servers. These templates specify default configuration information used to represent users and groups in each of the supported LDAP servers. You choose the template that corresponds to the LDAP server you want to use and then fill in the host and port of the LDAP server, and the GroupDN, the UserDN, Principal, and Credential attributes. For information about these attributes, see "Configuring an LDAP V1 Security Realm."

To use a LDAP V2 security realm:

1. Expand the Compatibility Security-->Realms nodes.
2. Choose the LDAP server you want to use with WebLogic Server. The following templates are available:
   - defaultLDAPRealmforOpenLDAPDirectoryServices
   - defaultLDAPRealmforNovellDirectoryServices
   - defaultLDAPRealmforMicrosoftSiteServer
   - defaultLDAPRealmforNetscapeDirectoryServer

3. On the Configuration tab, enter the host and port of the LDAP server in the server.host and server.port attributes in the Configuration Data box.
4. If necessary, update the information defined for the GroupDN, the UserDN, Principal, and Credential attributes for your LDAP directory server in the Configuration Data box.
5. Optionally, define a password for the LDAP server. The Password attribute defines the password for the Principal. Once the password is defined, WebLogic Server encrypts it.
6. Click Apply to save your changes.
7. When you have finished defining the attributes, reboot WebLogic Server.
8. Configure the Caching realm as described in "Configuring the Caching Realm" on page 4-6.

   When configuring the Caching realm, select the LDAP V2 security realm from the pull-down menu for the Basic Realm attribute on the General tab. The Basic Realm attribute defines the association between the Caching realm and the alternate security realm (in this case, the LDAP V2 security realm).
9. Expand the Domains node.
10. Select the Security-->File Realm tab.
11. In the Caching Realm attribute, choose the name of the Caching realm to be used with the LDAP V2 security realm. A list of configured Caching realms appears on the pull-down menu.
12. Reboot WebLogic Server.

Configuring the Windows NT Security Realm

The Windows NT Security realm uses account information defined for a Windows NT domain to authenticate users and groups. You can view users and groups in the Windows NT Security realm through the Administration Console, but you must manage users and groups through the facilities provided by Windows NT.

The Windows NT Security realm provides authentication (users and groups) but not authorization (ACLs). To update the ACL information in the `filerealm.properties` file that WebLogic Server uses, click Refresh on the General tab in the Security node after you change an ACL. If you use groups with your ACLs, reduce the frequency with which you must refresh the information in WebLogic Server. Changing the members of a Windows NT group allows you to manage individual users' access to WebLogic Server resources dynamically.

It is possible to use the Windows NT Security realm to authenticate against a Windows 2000 Active Directory primary domain controller. However, the authentication must be from a machine which is a member of the domain, not from the domain controller itself. There is no way to authenticate to the local user and group store if the machine running the Windows NT Security realm is a member of another domain.

The Windows NT Security realm can be run on the primary domain controller, on a machine that is a member of a Windows NT domain, or on a machine that is a member of the Windows NT domain and wants to use a mutually trusted domain.

To use the Windows NT Security realm:

1. Expand the Compatibility Security-->Realms nodes.
2. Click the Configure a New NT Realm... link.
3. Set attributes on the Configuration tab that define a name for the Windows NT realm and the computer on which the Windows NT domain is running.

   The following table describes the attributes you set in the NT Realm Configuration window.

Table 4-11 Windows NT Security Realm Attributes

Attribute	Description
Name	The name of the Windows NT Security realm, such as AccountingRealm.
Primary Domain	The host and port number of the computer where users and groups are defined for the Windows NT domain. If you enter multiple host and port numbers, use a comma delineated list.

4. Click Apply to save your changes.

5. Reboot WebLogic Server.
6. Configure the Caching realm as described in "Configuring the Caching Realm" on page 4-6

   When configuring the Caching realm, select your Windows NT Security realm from the pull-down menu for the Basic Realm attribute on the General tab. The Basic Realm attribute defines the association between the Caching realm and the alternate security realm (in this case, the Windows NT Security realm).
7. Expand the Domains node.
8. Click the Security-->File Realm tab.
9. In the Caching Realm attribute, choose the name of the Caching realm to be used with the Windows NT security realm. A list of configured Caching realms appears on the pull-down menu.
10. Reboot WebLogic Server.

Use the following command to verify that you have the correct privileges to run WebLogic Server as the specified Windows NT user:

```
java weblogic.security.ntrealm.NTRealm username password
```

where *username* and *password* are the username and password of the Windows NT account under which WebLogic Server runs.

The output from this command indicates if the specified username and password authenticated properly.

Table 4-12 Windows NT Authentication Verification

Command Output	Meaning
`auth?poppy`	The entered username and password authenticated correctly.
`auth?null`	The entered username and password did not authenticate properly.

If the test comes up with an immediate failure stating that the client or user running WebLogic Server does not have the privileges to run the Windows NT Security realm, you need to update the permissions (referred to as rights) for the Windows user running WebLogic Server.

Updating Users Permissions for Windows NT and Windows 2000

To update the rights in Windows NT:

1. On the Start menu, select Programs→Administrative Tools.
2. Select User Manager.
3. Under the Policies menu, choose the User Rights option.
4. Check the Show Advanced Users Rights option.
5. Give the following rights to the Windows user running WebLogic Server:
   - Act as part of the operating system
   - Create a token object
   - Replace a process level token
6. Verify that the Windows user running WebLogic Server is a member of the Administrators group.
7. Reboot Windows NT to ensure all the modifications take effect.
8. Verify that the Logon as System Account option is checked. Note that the Allow System to Interact with Desktop option does not need to be checked. Running the Windows NT Security realm under a specific Windows NT user account does not work.

To update the rights in Windows 2000:

1. On the Start menu, select Programs→Administrative Tools.
2. Select Local Security Policy.
3. Go to Local Policies→User Rights Assignment.
4. Give the following rights to the Windows user running WebLogic Server:
   - Act as part of the operating system
   - Create a token object
   - Replace a process level token

5. Verify that the Windows user running WebLogic Server is a member of the Administrators group.
6. Reboot Windows 2000 to ensure all the modifications take effect.
7. Verify that the Logon as System Account option is checked. Note that the Allow System to Interact with Desktop option does not need to be checked. Running the Windows NT Security realm under a specific Windows NT user account does not work.

The following are common Windows NT error codes that occur when using the Windows NT Security realm:

Table 4-13

Error Code	Meaning
1326	The host machine running the security realm does not have a trust relationship with the primary domain controller. The host machine may not be a member of the domain or the domain may not trust the host machine.
53	A network error has indicates that the path to the primary domain controller could not be located. This error can occur if the domain name is misspelled or if the domain name is specified rather than the host name of the primary domain controller.

A full explanation of the Windows NT error codes is found in the `winerror.h` file.

Configuring the UNIX Security Realm

Note: The UNIX Security realm runs only on the Solaris and Linux platforms.

The UNIX Security realm executes a small native program, `wlauth`, to look up users and groups and to authenticate users on the basis of their UNIX login names and passwords. The `wlauth` program uses Pluggable Authentication Modules (PAM), which allow you to configure authentication services in the operating system without altering applications that use the service.

After you change an ACL, click Refresh on the General tab in the Security to update the information in the `filerealm.properties` file that WebLogic Server uses. If you use groups with your ACLs, reduce the frequency with which you must refresh the information in WebLogic Server. Changing the members of a UNIX group allows you to manage individual users' access to WebLogic Server resources dynamically.

The `wlauth` program runs `setuid root`. You need root permissions to modify the ownership and file attributes on the `wlauth` program and to set up the PAM configuration file for `wlauth`.

To set up the `wlauth` program for the UNIX Security realm:

1. If WebLogic Server is installed on a network drive, copy the `wlauth` file to a file system on the computer that executes WebLogic Server, for example, the `/usr/sbin` directory. The `wlauth` file is in the `weblogic/lib/`*`arch`* directory, where *`arch`* is the name of your platform.

2. As the root user, run the following commands to change the `wlauth` owner and permissions:

   ```
   # chown root wlauth
   # chmod +xs wlauth
   ```

3. Set up the PAM configuration for `wlauth`.

   Solaris—Add the following lines to your `/etc/pam.conf` file:

   ```
   # Setup for WebLogic authentication on Solaris machines
   #
   wlauth auth required      /usr/lib/security/pam_unix.so.1
   wlauth password required  /usr/lib/security/pam_unix.so.1
   wlauth account required   /usr/lib/security/pam_unix.so.1
   ```

   Linux—Create a file called `/etc/pam.d/wlauth` containing the following:

   ```
   #%PAM-1.0
   #
   # File name:
   # /etc/pam.d/wlauth
   #
   # If you do not use shadow passwords, delete "shadow".
   auth required     /lib/security/pam_pwdb.so shadow
   account required  /lib/security/pam_pwdb.so
   ```

   Note: Omit `shadow` if you are not using shadow passwords.

To configure the UNIX Security realm:

1. Expand the Compatibility Security-->Realms nodes.
2. Click the Configure a New Unix Realm... link.
3. Set attributes on the Configuration tab that define a name for the realm and the program that provides authentication services for the UNIX Security realm.

Table 4-14 UNIX Security Realm Attributes

Attribute	Description
Name	The name of the UNIX Security realm, such as AccountingRealm.
AuthProgram	The name of the program used to authenticate users in the UNIX security realm. In most cases, the name of the program is `wlauth`.

4. Click Create.
5. When you have finished defining the attributes, reboot WebLogic Server.
6. Configure the Caching realm as described in "Configuring the Caching Realm" on page 4-6.

   When configuring the Caching realm, select your UNIX Security realm from the pull-down menu for the Basic Realm attribute on the General tab. The Basic Realm attribute defines the association between the Caching realm and the alternate security realm (in this case, the UNIX Security realm).
7. Expand the Domains node.
8. Select the Security-->File Realm tab.
9. In the Caching Realm attribute, choose the name of the Caching realm to be used with the UNIX security realm. A list of configured Caching realms appears on the pull-down menu.
10. Reboot WebLogic Server.

If `wlauth` is not in the WebLogic Server path or if you have given the program a name other than `wlauth`, you must add a Java command-line property when you start WebLogic Server. Edit the script you use to start WebLogic Server and add the following option after the `java` command:

```
-Dweblogic.security.unixrealm.authProgram=wlauth_prog
```

Replace *wlauth_prog* with the name of the `wlauth` program, including the full path if the program is not in the search path. Start WebLogic Server. If the `wlauth` program is in the WebLogic Server path and is named `wlauth`, this step is not needed.

Configuring the RDBMS Security Realm

Note: If your implementation of the RDBMS security realm uses the `getActiveDomain()` method, you need to edit and recompile your `RDBMSDelegate` class in order to use the RDBMS security realm with Compatibility security. Replace the `getActiveDomain()` method with the `getSecurityConfig()` method in the `weblogic.server` package.

The RDBMS Security realm is a BEA-provided custom security realm that stores users, groups and ACLs in a relational database.The RDBMS Security realm is an example and is not meant to be used in a production environment. You can perform the following managment functions on the RDBMS Security realm.

Table 4-15 Management Functions in the RDBMS Security Realm

Management Function	Support for Functions
Create User	Administration Console, but only in memory
Delete User	Administration Console
Change Password	`user()` interface in `weblogic.security.acl`
Create Group	`manageableRealm()` interface in `weblogic.security.acl`
Delete Group	Administration Console
Add Group Member	Administration Console
Delete Group Member	Administration Console
Create ACL	`manageableRealm()` interface in `weblogic.security.acl`
Delete ACL	`manageableRealm()` interface in `weblogic.security.acl`

Table 4-15 Management Functions in the RDBMS Security Realm

Management Function	Support for Functions
Add Permission	`acl()` interface in `weblogic.security.acl`
Delete Permission	`acl()` interface in `weblogic.security.acl`

SQL scripts that populate a database are used to create groups for the RDBMS Security realm.

The RDBMS Security realm can be used as a starting point for creating a production security realm. Extend the RDBMS Security realm by using the following interfaces in the `weblogic.security.acl` package to add management capabilities to the RDBMS Security realm:

- `ManageableRealm`—Create groups, create and delete ACLs, and perform lookups of users, groups, and ACLs.
- `User`—Change the password.
- `ACL`—Add and remove permissions for users and groups.

If you extend the RDBMS Security realm with any of these interfaces, you may also need to update the database schema.

Note: The RDBMS example does not work with databases that have an autocommit feature enabled. If you use the RDBMS example as a starting point for your RDBMS implementation, use explicit commit statements in your code and make sure the autocommit feature in the database you are using is disabled.

To configure the RDBMS Security realm:

1. Expand the Compatibility Security-->Realms nodes.
2. Choose the database you want to use with WebLogic Server. The following templates are available:
   - defaultRDBMSRealmForOracle
   - defaultRDBMSRealmForMSSQLServerType4
   - defaultRDBMSRealmForCloudScape
   - defaultRDBMSRealmForODBC

A configuration window for the chosen database appears.

3. Set attributes on the General tab that define a name for the realm and the class that implements the RDBMS security realm.

   The following table describes the attributes you set on the General tab.

Table 4-16 RDBMS Security Realm Attributes on the General Tab

Attribute	Description
Name	Name of the RDBMS Security realm, such as AccountingRealm.
Realm Class	Name of the WebLogic class that implements the RDBMS Security realm. The Java class needs to be in the CLASSPATH of WebLogic Server.

4. Click Apply to save your changes.
5. Select the Database tab. Define attributes for the JDBC driver being used to connect to the database.

   The following table describes the attributes you set on the Database tab.

Table 4-17 RDBMS Security Realm Attributes on the Database Tab

Attribute	Description
Driver	Full class name of the JDBC driver. This class name must be in the CLASSPATH of WebLogic Server.
URL	URL for the database you are using with the RDBMS realm, as specified by your JDBC driver documentation.
User Name	Default user name for the database.
Password	Password for the default user of the database.

6. Click Apply to save your changes.

7. Select the Schema tab. Define the schema used to store Users, Groups, and ACLs in the database in the Schema Properties box on the Schema tab.

   Listing 4-1 contains the database statements entered in the Schema properties for the RDBMS code example shipped with WebLogic Server in the `\samples\server\src\examples\security\rdbmsrealm` directory.

Listing 4-1 Sample Schema for RDBMS Security Realm

```
“getGroupNewStatement=true;getUser=SELECT U_NAME, U_PASSWORD FROM
users WHERE U_NAME = ?;
getGroupMembers=SELECT GM_GROUP, GM_MEMBER from groupmembers WHERE
GM_GROUP = ?;
getAclEntries=SELECT A_NAME, A_PRINCIPAL, A_PERMISSION FROM
aclentries WHERE A_NAME = ? ORDER BY A_PRINCIPAL;
getUsers=SELECT U_NAME, U_PASSWORD FROM users;
getGroups=SELECT GM_GROUP, GM_MEMBER FROM groupmembers;
getAcls=SELECT A_NAME, A_PRINCIPAL, A_PERMISSION FROM aclentries
ORDER BY A_NAME, A_PRINCIPAL;
getPermissions=SELECT DISTINCT A_PERMISSION FROM aclentries;
getPermission=SELECT DISTINCT A_PERMISSION FROM aclentries WHERE
A_PERMISSION = ?;
newUser=INSERT INTO users VALUES ( ? , ? );
addGroupMember=INSERT INTO groupmembers VALUES ( ? , ? );
removeGroupMember=DELETE FROM groupmembers WHERE GM_GROUP = ? AND
GM_MEMBER = ?;
deleteUser1=DELETE FROM users WHERE U_NAME = ?;
deleteUser2=DELETE FROM groupmembers WHERE GM_MEMBER = ?;
deleteUser3=DELETE FROM aclentries WHERE A_PRINCIPAL = ?;
deleteGroup1=DELETE FROM groupmembers WHERE GM_GROUP = ?;
deleteGroup2=DELETE FROM aclentries WHERE A_PRINCIPAL = ?”
```

8. Click Apply to save your changes.
9. When you have finished defining the attributes, reboot WebLogic Server.
10. Configure the Caching realm as described in “Configuring the Caching Realm” on page 4-6.

    When configuring the Caching realm, select the RDBMS Security realm from the pull-down menu for the Basic Realm attribute on the General tab. The Basic Realm attribute defines the association between the Caching realm and the alternate security realm (in this case, the RDBMS Security realm).

11. Expand the Domains node.
12. Select the Security-->File Realm tab.
13. In the Caching Realm attribute, choose the name of the Caching realm to be used with the RDBMS security realm. A list of configured Caching realms appears on the pull-down menu.
14. Reboot WebLogic Server.

Installing a Custom Security Realm

You can create a custom security realm that draws from an existing store of users such as directory server on the network. To use a custom security realm, you create an implementation of the `weblogic.security.acl.AbstractListableRealm` interface or the `weblogic.security.acl.AbstractManageableRealm` interface and then use the Administration Console to install your implementation. For more information, see Writing a Custom Security Realm.

To install a custom security realm:

1. Expand the Compatibility Security-->Realms nodes.
2. Click the Configure a New Custom Realm... link.
3. Set attributes on the Configuration tab that define a name for the custom security realm, specify the interface that implements the realm, and define how the users, groups, and optionally ACLs are stored in the custom security realm on the tab.

   The following table describes the attributes you set on the Custom Security Realm Configuration window.

Table 4-18 Custom Security Realm Attributes

Attribute	Description
Name	Name of the Custom Security realm, such as AccountingRealm.
Realm Class Name	Name of the WebLogic class that implements the Custom Security realm. The Java class needs to be in the CLASSPATH of WebLogic Server.

Table 4-18 Custom Security Realm Attributes

Attribute	Description
Configuration Data	Information needed to connect to the security store.
Password	Password for the Custom Security realm. If a password is supplied, WebLogic Server encrypts it.

4. Click Create.
5. Reboot WebLogic Server.
6. Configure the Caching realm as described in "Configuring the Caching Realm" on page 4-6.

   When configuring the Caching realm, select the Custom Security realm from the pull-down menu for the Basic attribute on the General tab. The Basic attribute defines the association between the Caching realm and the custom security realm.
7. Expand the Domains node.
8. Click the Security-->File Realm tab.
9. In the Caching Realm attribute, choose the name of the Caching realm to be used with the custom security realm. A list of configured Caching realms appears on the pull-down menu.
10. Reboot WebLogic Server.

Defining Users in the Compatibility Realm

Note: This section explains how to add users to a manageable security realm (for example, the File realm) in the Compatibility realm. If you are using a security realm that is not manageable through the Administration Console, you must use the administration tools provided in that realm to define a user.

Users are entities that can be authenticated in a WebLogic Server security realm. A user can be a person or a software entity, such as a Java client. Each user is given a unique identity within a WebLogic Server security realm. As a system administrator you must guarantee that no two users in the same security realm are identical.

Defining users in a security realm involves specifying a unique name and password for each user that accesses resources in the WebLogic Server security realm in the users window of the Administration Console.

For information about how to upgrade the `guest` users, see Guest User in the *WebLogic Server 7.0 Upgrade Guide*.

WebLogic Server has two special users:

- The `system` user is the administrative user who controls system-level WebLogic Server operations, such as starting and stopping servers, and locking and unlocking resources. The `system` user and its password are defined during the WebLogic Server installation procedure. As a security precaution, BEA recommends changing the password for the `system` user. For more information, see "Changing the System Password" on page 4-3.
- The `guest` user is automatically provided by WebLogic Server. When authorization is not required, WebLogic Server assigns the `guest` identity to a client, thus giving the client access to any resources that are available to the `guest` user. A client can log in as the `guest` user by entering `guest` as the username and `guest` as the password when prompted by a Web browser or by supplying the `guest` username and password in a Java client. By default, the `guest` account is disabled.

  For a more secure deployment, BEA recommends running WebLogic Server with the `guest` account disabled. To disable the `guest` account, select the Guest Disabled attribute on the General tab of the Security Configuration window. Disabling the `guest` account just disables the ability to log in into the account `guest`; it does not disable the ability of unauthenticated users to access a WebLogic Server deployment.

The `system` and `guest` users are like other users in a WebLogic Server security realm:

- To access WebLogic Server resources, they must have appropriate ACLs.
- To execute an operation on a WebLogic Server resource, they must provide a username and password (or digital certificate).

To define a user:

1. Expand the Compatibility Security node.
2. Click Users.
3. In the User Configuration window, enter the name of the user in the Name attribute.
4. Enter a password for the user in the Password attribute.
5. Enter the password again in the Confirm Password attribute.
6. Click Create.

To delete a user:

1. Expand the Compatibility Security node.
2. Click Users.
3. In User Configuration window, enter the name of the user in the Delete Users box.
4. Click Delete.

To change the password of a user:

1. Expand the Compatibility Security node.
2. Click Users.
3. In the User Configuration window, enter the name of the user in the Name attribute.
4. Enter the old password in the Old Password attribute.
5. Enter the new password in the New Password attribute.
6. Enter the new password again to confirm the password change.

While using WebLogic Server, you may have users that are locked. Perform the following steps to unlock a user:

1. Expand the Compatibility Security node.
2. Click Users.
3. In the User Configuration window, click the Unlock Users link.

4. Enter the names of the user accounts you want to unlock in the Users to Unlock field.
5. Choose the servers on which you want the user accounts unlocked.
6. Click Unlock.

Defining Groups in the Compatibility Realm

Note: This section explains how to add groups to a manageable security realm (for example, the File realm) in the Compatibility realm. If you are using a security realm that is not manageable through the Administration Console, you must use the administration tools provided in that realm to define a group.

A group represents a set of users who usually have something in common, such as working in the same department in a company. Groups are a means of managing a number of users in an efficient manner. When a group is granted a permission in an ACL, all members of the group effectively receive that permission. BEA recommends assigning permissions to groups rather than to individual users.

By default, WebLogic Server has the following groups:

- All users defined in the defined security realm are automatically members of the `everyone` group.
- The `system` user is a member of the `Administrators` group. This group should be given the permissions appropriate for a user responsible for starting and stopping servers and maintaining a running WebLogic Server deployment. Access to this group should be limited.

To define a group in the Compatibility realm:

1. Expand the Compatibility Security node.
2. Click Groups.
3. Click the Create a New Group... link.

4. In the Group Configuration window, enter the name of the group in the Name attribute. BEA recommends naming groups in the plural. For example, Administrators instead of Administrator.
5. Click on the Users attribute and select the WebLogic Server users you want to add to the group.
6. Click on the Groups attribute and select the WebLogic Server Groups you want to add to the Group.
7. Click Apply to create a new Group.

To delete groups, enter the name of the group in the Remove These Groups list box on the Group Configuration window and click Remove.

Defining ACLs in the Compatibility Realm

Users access resources in a WebLogic Server security realm. Whether or not a user can access a resource is determined by the access control lists (ACLs) for that resource. An ACL defines the permissions by which a user can interact with the resource. To define ACLs, you create an ACL for a resource, specify the permission for the resource and then grant the permission to a specified set of users and groups. BEA recommends assigning ACLs to groups.

Each WebLogic Server resource has one or more permissions that can be granted. The following table summarizes the functions for various WebLogic Server resources for which permissions can be restricted with an ACL.

Table 4-19 ACLs for WebLogic Server Resources

For this WebLogic Server resource...	This ACL...	Grants Permission for these functions...
WebLogic Servers	weblogic.server weblogic.server.*servername*	boot

Table 4-19 ACLs for WebLogic Server Resources

For this WebLogic Server resource...	This ACL...	Grants Permission for these functions...
Command-line Administration Tools	weblogic.admin **Note:** To add ACLs through the Administration Console, you need to define `weblogic.admin.acl.modify`.	shutdown, lockServer unlockServer, modify
WebLogic Events	`weblogic.event.`*`topicName`*	send receive
WebLogic JDBC connection pools	`weblogic.jdbc.connectionPool.`*`poolname`*	reserve admin
WebLogic Passwords	weblogic.passwordpolicy	unlockuser
WebLogic JMS destinations	`weblogic.jms.topic.`*`topicName`* `weblogic.jms.queue.`*`queueName`*	send, receive
WebLogic JNDI contexts	`weblogic.jndi.`*`path`*	lookup modify list

Note: ACLs on MBeans are not supported in WebLogic Server 7.0. For more information, see Protecting System Administration Operations in the *BEA WebLogic Server Administration Guide*.

To create ACLs for a WebLogic Server resource, open the Administration Console and perform the following steps:

1. Expand the Compatibility Security node.
2. Click the ACLs tab.
3. Click the Create a New ACL... link.
4. In the ACL Configuration window, in the New ACL Name attribute specify the name of WebLogic Server resource that you want to protect with an ACL.

For example, create an ACL for a JDBC connection pool named `demopool`.

5. Click Create.
6. Click on the Add a New Permission link.
7. Specify a permission for the resource.

   Either create separate ACLs for each permission available for a resource or one ACL that grants all the permissions for a resource. For example, you can create three ACLs for the JDBC connection pool, `demopool`: one with `reserve` permission, one with `reset` permission, and one with `shrink` permission. Or you can create one ACL with `reserve`, `reset`, and `shrink` permissions.
8. Specify Weblogic users or groups that have the specified permission to the resource.
9. Click Apply.

When creating ACLs for resources in WebLogic Server, you need to use the syntax in Table 4-19 to refer to the resource. For example, the JDBC connection pool named `demopool` would be specified as `weblogic.jdbc.connectionPool.demopool`.

If you modify an existing ACL, click Refresh on the General tab in the Security node to update the information in the `filerealm.properties` file that WebLogic Server uses.

Protecting User Accounts

Weblogic Server provides a set of attributes to protect user accounts from intruders. By default, these attributes are set for maximum protection. As a system administrator, you have the option of turning off all the attributes, increasing the number of login attempts before a user account is locked, increasing the time period in which invalid login attempts are made before locking the user account, and changing the amount of time a user account is locked. Remember that changing the attributes lessens security and leaves user accounts vulnerable to security attacks.

To protect the user accounts in your WebLogic Server domain, perform the following steps:

1. Click on the Domains node.
2. Select the Security-->Passwords tab.
3. Define the desired attributes on this tab by entering values at the appropriate prompts and selecting the required checkboxes. (For details, see the following table).
4. Click Apply to save your choices.
5. Reboot WebLogic Server.

The following table describes each attribute on the Passwords tab.

Table 4-20 Password Protection Attributes

Attribute	Description
Minimum Password Length	Number of characters required in a password. Passwords must contain a minimum of 8 characters. The default is 8.
Lockout Enabled	Requests the locking of a user account after invalid attempts to log in to that account exceed the specified Lockout Threshold. By default, this attribute is enabled.
Lockout Threshold	Number of failed password entries for a user that can be tried to log in to a user account before that account is locked. Any subsequent attempts to access the account (even if the username/password combination is correct) raise a Security exception; the account remains locked until it is explicitly unlocked by the system administrator or another login attempt is made after the lockout duration period ends. Invalid login attempts must be made within a span defined by the `Lockout Reset Duration` attribute. The default is 5.

Table 4-20 Password Protection Attributes

Attribute	Description
Lockout Duration	Number of minutes that a user's account remains inaccessible after being locked in response to several invalid login attempts within the amount of time specified by the `Lockout Reset Duration` attribute. In order to unlock a user account, you need to have the `unlockuser` permission for the `weblogic.passwordpolicy`. The default is 30 minutes.
Lockout Reset Duration	Number of minutes within which invalid login attempts must occur in order for the user's account to be locked. An account is locked if the number of invalid login attempts defined in the `Lockout Threshold` attribute happens within the amount of time defined by this attribute. For example, if the value in this attribute is five minutes and three invalid login attempts are made within a six-minute interval, then the account is not locked. If five invalid login attempts are made within a five-minute period, however, then the account is locked. The default is 5 minutes.
Lockout Cache Size	Specifies the intended cache size of unused and invalid login attempts. The default is 5.

Using a 6.x Auditor with Compatibility Security

If your WebLogic Server 6.x security configuration uses an implementation of the `weblogic.security.audit.AuditProvider` class, the Auditor is not automatically configured in Compatibility security. Configure a Realm Adapter Auditing provider in the Compatibility realm to access the 6.x Auditor.

1. Start WebLogic Server.
2. Run the admin command line tool, entering the following three commands:

   ```
   java weblogic.Admin -url t3://localhost:7001 -username
   adminusername -password adminpassword CREATE -mbean Security:
   Name=CompatibilityRealmRealmAdapterAuditor -type
   weblogic.security.providers.realmadapter.RealmAdapterAuditor
   commotype
   ```

   ```
   java weblogic.Admin -url t3://localhost:7001 -username
   adminusername -password adminpassword SET -mbean Security:
   Name=CompatibilityRealmRealmAdapterAuditor -property Realm
   Security:Name=CompatibilityRealm commotype
   ```

   ```
   java weblogic.Admin -url t3://localhost:7001 -username
   adminusername -password adminpassword SET -mbean Security
   Name=CompatibilityRealm -property Auditors
   Security:Name=CompatibilityRealmRealmAdapterAuditor commotype
   ```

3. Reboot WebLogic Server.

5 Configuring the SSL Protocol

This following sections describe how to configure the SSL protocol for WebLogic Server:

Note: This chapter applies to WebLogic Server deployments using the security features in WebLogic Server 7.0 as well as deployments using Compatibility Security.

Configuring the SSL protocol is an optional step, however, BEA recommends using the SSL protocol in production.

SSL Protocol: Introduction

The Secure Sockets Layer (SSL) protocol provides secure connections by allowing two applications connecting over a network connection to authenticate the other's identity and by encrypting the data exchanged between the applications. Authentication allows a server and optionally a client to verify the identity of the application on the other end of a network connection. Encryption makes data transmitted over the network intelligible only to the intended recipient.

The SSL protocol in WebLogic Server 7.0 implements the SSL 3.0 and TLS 1.0 specifications.

Note: The proxy plug-ins in WebLogic Server 7.0 only support SSL 3.0.

WebLogic Server supports the SSL protocol on a dedicated listen port which defaults to 7002. To establish an SSL connection, a Web browser connects to WebLogic Server by supplying the SSL listen port and the HTTPs schema in the connection URL, for example, `https://myserver:7002`.

Using the SSL protocol is computationally intensive and adds overhead to a connection. You should avoid using the SSL protocol when it is not necessary. However, you should always use the SSL protocol in a production environment.

Private Keys, Digital Certificates and Trusted Certificate Authorities

Private keys, digital certificates, and trusted certificate authorities (CAs) establish and verify server identity.

The SSL protocol uses public key encryption technology for authentication. With public key encryption, a public key and a *private key* are generated for a server. The keys are related such that data encrypted with the public key can only be decrypted using the corresponding private key. The private key is carefully protected so that only the owner can decrypt messages.

The public key is embedded into a *digital certificate* with additional information describing the owner of the public key, such as name, street address, and e-mail address.

The data embedded in a digital certificate is verified by a certificate authority (also referred to as trusted CA) and digitally signed with the certificate authority's digital certificate. Well-know certificate authorities include Verisign and Entrust.net.

An application participating in an SSL connection is authenticated when the other party evaluates and accepts their digital certificate. Web browsers, servers, and other SSL-enabled applications generally accept as genuine any digital certificate that is signed by a trusted CA and is otherwise valid. For example, a digital certificate can be invalidated because it has expired or the digital certificate of the CA used to sign it expired. A server certificate can be invalidated in the URL of the server that presented the certificate does not match the URL embedded in the certificate.

One-Way and Two-Way SSL

The SSL protocol can be configured one-way or two-way:

- With one-way SSL, the server is required to present a certificate to the client but the client is not required to present a certificate to the server. To successfully negotiate an SSL connection, the client must authenticate the server but the

server will accept any client into the connection. One-way SSL is common on the Internet where customers want to create secure connections before they share personal data.

- With two-way SSL, the client also presents a certificate to the server. WebLogic Server can be configured to require clients to submit certificates issued by any of up to four trusted certificate authorities before completing the SSL connection.

Setting Up the SSL Protocol: Main Steps

To set up the SSL protocol:

1. Obtain a private key, digital certificate, and trusted CA for WebLogic Server. Use the demonstration digital certificates, private keys, and trusted CAs provided in the WebLogic Server kit, the Cert Gen utility, the Certificate Request Generator servlet, or Sun Microsystem's `keytool` utility to perform this step.
2. Store the private keys, digital certificates, and trusted CAs. Digital certificates are always stored in file in the domain directory of WebLogic Server. Private keys and trusted CAs can either be stored in the WebLogic Keystore provider or in a file in the domain directory.
3. Enable the SSL protocol. By default, the SSL protocol is not enabled because of the performance impact of its use.
4. Set SSL attributes in the Administration Console. The SSL attributes define the location of the private key, digital certificate, and trusted CAs. Optionally, set attributes that require the presentation of client certificates (for two-way SSL).

For more information on these steps, see:

"Obtaining Private Keys, Digital Certificates and Trusted CAs" on page 5-5

"Storing Private Keys, Digital Certificates, and Trusted CAs" on page 5-11

"Enabling the SSL Protocol" on page 5-15

"Setting Attributes for One-Way SSL" on page 5-16

"Setting Attributes for Two-Way SSL" on page 5-17

Obtaining Private Keys, Digital Certificates and Trusted CAs

To use the SSL protocol, you need a private key, a digital certificate containing the matching public key, and at least one trusted CA that can verify the data embedded in the digital certificate. WebLogic Server supports private keys, digital certificates, and trusted CAs from the following sources:

- The demonstration digital certificates, private keys, and trusted CAs in the `\weblogic700\server\lib` directory.

  When you create a domain through the Configuration Wizard, the demonstration certificates and private keys are copied into the domain directory. The digital certificate file contains the name of the local host and the private key file is time-stamped by the build. The demonstration digital certificates, private keys, and trusted CAs should only be used for demonstration or testing purposes and not in a production environment

- The Cert Gen utility generates digital certificate and private keys that should only be used for demonstration or testing purposes and not in a production environment. For more information about using the Cert Gen utility to obtain private keys and digital certificates, see "Using the Cert Gen Utility" on page 5-6.

- The Certificate Request Generator servlet generates a private key file and a Certificate Signature Request (CSR). To acquire a digital certificate from a trusted CA (such as VeriSign or Entrust.net), you must submit your request in a particular format called a CSR. The digital certificate you receive from the certificate authority and the private key from the Certificate Request Generator servlet are suitable for production environments. For more information, see "Using the Certificate Request Generator Servlet" on page 5-8.

- Sun Microsystem's `keytool` utility can also be used to generate a private key, a self-signed digital certificate for WebLogic Server, and a CSR. Submit the CSR to a certificate authority to obtain a digital certificate for WebLogic Server. Use `keytool` to update the self-signed digital certificate with a new digital certificate. For more information about Sun's `keytool` utility, see keytool—Key and Certificate Management Tool.

WebLogic Server can use digital certificates in either `.pem` or `.der` format. The file extension determines the format of the digital certificate file.

A `.pem` (privacy-enhanced mail) format file begins with this line:

```
----BEGIN CERTIFICATE----
```

and ends with this line:

```
----END CERTIFICATE----
```

A `.der` format file contains binary data. WebLogic Server requires that the file extension match the contents of the digital certificate, so be sure to save the file you receive from the trusted CA with the correct file extension.

Private keys must be in PKCS#5/PKCS#8 PEM format.

Using the Cert Gen Utility

Note: The CertGen utility generates digital certificates and private keys that should only be used for demonstration or testing purposes and not in a production environment.

The CertGen utility creates a private key and digital certificate signed by the demonstration certificate authority (`CertGenCA`). The digital certificates generate by the Cert Gen utility have the hostname of the machine on which they were generated as the common name, so you must generate a digital certificate for every machine on which you wish to use the SSL protocol.

The CertGen utility generates two `.pem` files and two `.der` files. View the `.der` files in a Web browser to view the details of the generated digital certificate. Use the `.pem` files when you boot WebLogic Server or use the digital certificates with a client.

To generate a certificate:

1. Copy the following files to the directory in which you run the CertGen utility:
   - `WL_HOME/server/lib/CertgenCA.der`—The digital certificate for a certificate authority trusted by WebLogic Server.
   - `WL_HOME/server/lib/CertGenCAKey.der`—The private key for a certificate authority trusted by WebLogic Server.
2. Enter the following command at a command prompt:

prompt> `java utils.CertGen` *`password certfile keyfile`* `[export]`

where

- *`password`* is the password for the private key.
- *`certfile`* is the name of the digital certificate file. The file is put in the domain directory.
- *`keyfile`* is the name of the generated private key file. The file is put in the domain directory.

By default, the CertGen tool generates domestic strength certificates. Specify the `[export]` option if you want the tool to generate export strength certificates.

The CertGen tool uses the JDK version 1.3 `InetAddress.getLocalHost().getHostname()` method to get the hostname it puts in the common name field of a certificate. The `getHostName()` method works differently on different platforms. It returns a fully qualified domain name (FQDN) on some platforms (for example, Solaris) and a short host name on other platforms (for example, Windows NT). If WebLogic Server is acting as a client (and by default hostname verification is enabled), you need to ensure the hostname specified in the URL matches the common name field in the certificate. Otherwise, your connection fails because the hostnames do not match.

On Solaris, when you type `hostname` on the command line the server looks at the `/etc/hosts` file and gets a short hostname. When you invoke `java.net.InetAddress.getHostName(),` the host goes to the `/etc/nsswitch.conf` file and depending on how the host is configured returns a FQDN or a short hostname.

If the host entry is configured as:

```
hosts:   dns nis [NOTFOUND=return]
```

The host performs a name service look up first and uses the information `/etc/hosts` file only if DNS is not available. In this case, a FQDN is returned.

If the host entry is configured as:

```
hosts:   files dns nis [NOTFOUND=return]
```

The host goes to the `/etc/hosts` file first and then goes to DNS. In this case, a short hostname is returned.

Using the Certificate Request Generator Servlet

Before using a WebLogic Server deployment in a production environment, you need to obtain a private key and certificate from a trusted certificate authority such as VeriSign or Entrust.net. To acquire a digital certificate from a certificate authority, you must submit your request in a particular format called a CSR. The Certificate Request Generator servlet collects information from you and generates a private key file and a CSR. You then submit the CSR to a certificate authority.

To generate a CSR, perform the following steps:

1. Copy the `certificate.war` file to the applications directory (copy the file before the server boots or while the server is running).
2. In a Web browser, enter the URL for the Certificate Request Generator servlet as follows:

   `https://`*`hostname:port`*`/certificate/`

   The components of this URL are defined as follows:

   - *`hostname`* is the DNS name of the machine running WebLogic Server.
   - *`port`* is the number of the port at which WebLogic Server listens for SSL connections. The default is 7002.

     For example, if WebLogic Server is running on a machine named `ogre` and it is configured to listen for SSL communications at the default port `7002` to run the Certificate Request Generator servlet, you must enter the following URL in your Web browser:

     ```
     https://ogre:7002/certificate/
     ```

3. The Certificate Request Generator servlet loads a form in your Web browser. Complete the form displayed in your browser, using the information in the following table:

Table 5-1 Fields on the Certificate Request Generator Form

Field	Description
Country code	Two-letter ISO code for your country. The code for the United States is US.

Table 5-1 Fields on the Certificate Request Generator Form

Field	Description
Organizational unit name	Name of your division, department, or other operational unit of your organization.
Organization name	Name of your organization. The certificate authority may require any host names entered in this attribute belong to a domain registered to this organization.
E-mail address	E-mail address of the administrator. The digital certificate is mailed to this e-mail address.
Full host name	Fully qualified name of the WebLogic Server on which the digital certificate will be installed. This name is the one used for DNS lookups of the WebLogic Server, for example, `node.com`. Web browsers compare the host name in the URL to the name in the digital certificate. If you change the host name later, you must request a new digital certificate.
Locality name (city)	Name of your city or town. If you operate with a license granted by a city, this attribute is required; you must enter the name of the city that granted your license.
State name	Name of the state or province in which your organization operates if your organization is in the United States or Canada, respectively. Do not abbreviate.
Private Key Password	The password used to encrypt the private key. Enter a password in this field if you want to use a protected key with WebLogic Server. If you choose to use a protected key, you are prompted for the password whenever the key is used. If you specify a password, you get a PKCS-8 encrypted private key. BEA recommends using a password to protect private keys.
Strength	The length (in bits) of the keys to be generated. The longer the key, the more difficult it is for someone to break the encryption. If you have the domestic version of WebLogic Server, choose 512-, 768-, or 1024-bit keys. The 1024-bit key is recommended.

4. Click Generate Request.

The Certificate Request Generator servlet displays messages informing you if any required attributes are empty or if any attributes contain invalid values. Click Back in your Web browser and correct any errors.

When all attributes have been accepted, the Certificate Request Generator servlet generates the following files in the startup directory of your WebLogic Server:

- `www__com-key.der`—The private key file.
- `www__com-request.dem`—The certificate request file, in binary format.
- `www__com-request.pem`—The CSR file that you submit to the certificate authority. It contains the same data as the `.dem` file but is encoded in ASCII so that it can be copied into e-mail or pasted it into a Web form.

5. Select a certificate authority and follow the instructions on that authority's Web site to purchase a digital certificate.

   - VeriSign, Inc. offers two options for WebLogic Server: Global Site Services, which features strong 128-bit encryption for domestic and export Web browsers, and Secure Site Services, which offers 128-bit encryption for domestic Web browsers and 40-bit encryption for export Web browsers.
   - Entrust.net certificates offer 128-bit encryption for domestic browser versions and 40-bit encryption for export browser versions.

   When you are instructed to select a server type, choose `BEA WebLogic Server` to ensure that you receive a digital certificate that is compatible with WebLogic Server.

Converting a Microsoft p7b Format to PEM Format

Microsoft can also be used as a certificate authority. The digital certificates issued by Microsoft are in a format (p7b) that cannot be used by WebLogic Server. To convert a digital certificate in p7b format to PEM format:

1. In Windows Explorer on Windows 2000, click on the file (*`filename`*`.p7b`) you want to convert.

   A Certificates window appears.

2. In the left pane of the Certificates window, expand the file you want to convert.
3. Select the Certificates option.

A list of certificates in the p7b file appear.

4. Select the certificate to convert tp PEM format.

   The Certificate Export wizard appears.

5. Click Next.
6. Select the `Base64 Encoded Cert` option (exports PEM formats).
7. Click Next.
8. Enter a name for the converted digital certificate.
9. Click Finish.

Note: For p7b certificate files that contain certificate chains, you need to concatenate the issuer PEM digital certificates to the certificate file. The resulting certificate file can be used by WebLogic Server.

Storing Private Keys, Digital Certificates, and Trusted CAs

Once you have obtained private keys, digital certificates, and trusted CAs, you need to store them so that WebLogic Server can use them to verify identity. Digital certificates can only be stored in a file in the domain directory. Private keys and trusted CAs can either be stored in a file in the domain directory or in a keystore.

- If you store the private keys, digital certificates, and trusted CAs in files in the domain directory, you need to set the SSL Server Key File Name, Server Certificate File Name, and Trusted CA File Name attributes in the Administration Console. For information about setting SSL attributes, see "Setting Attributes for One-Way SSL" on page 5-16.
- If you store the private keys and trusted CAs in a keystore, you need to create a keystore and load the private keys and trusted CAs into the keystore, configure the WebLogic Keystore provider to point to the keystore, and set the SSL Server Private Key Alias, Server Private Key Passphrase, Server Certificate File Name, and Trusted CA File Name in the Administration Console.

For more information, see:

- "Creating a Keystore and Loading Private Keys and Trusted CAs into the Keystore" on page 5-12
- "Configuring the WebLogic Server Keystore Provider to Locate a Keystore" on page 5-13
- "Setting Attributes for One-Way SSL" on page 5-16

Creating a Keystore and Loading Private Keys and Trusted CAs into the Keystore

A keystore is a mechanism designed to create and manage files that store private keys and trusted CAs. The WebLogic Keystore provider locates instances of keystores. Use the following utilities to create a keystore and load private keys and trusted CAs into the keystore:

- The WebLogic `ImportPrivateKey` utility. For more information, see utils.ImportPrivateKey.
- Sun Microsystem's keytool utility. For more information, see keytool—Key and Certificate Management Tool.

Note: The `keytool` utility requires a digital certificate to be presented when importing a private key into the keystore. This digital certificate is not used by WebLogic Server.

While you can use the `keytool` utility to load new private keys and trusted CAs into a keystore, the utility does not allow you to take an existing private key from a file and import it into the keystore.

Either create one keystore for both private keys and trusted CAs or two keystores: one for private keys and one for trusted CAs. BEA recommends creating two keystores.

By default, WebLogic Server looks for a private key keystore named `wlDefaultKeyStore.jks` in the domain directory. The keystore for trusted CAs can use either the demonstration trusted CA (`weblogic700/server/lib/cacerts`) or a trusted CA available from the JDK (`.../jre/lib/security/cacerts`).

Note: The `weblogic700/server/lib/cacerts` file should be used for demonstration or testing purposes and not in a production environment.

All keystore entries (private key and trusted CAs) are accessed via unique aliases. You specify the alias when loading the private key or trusted CA into the keystore. Aliases are case-insensitive; the aliases `Hugo` and `hugo` would refer to the same keystore entry. Aliases for private keys are specified in the Server Private Key Alias attribute when configuring the SSL protocol.

Configuring the WebLogic Server Keystore Provider to Locate a Keystore

Note: The WebLogic Keystore provider is deprecated in WebLogic Server 7.0 SP01.

The WebLogic Keystore provider has been enhanced in the following ways:

- You can specify a type of keystore. In WebLogic Server 7.0, only Java 2 Enterprise Edition keystores (JKS) were supported.
- You can protect each private key with an individual password. This password provides an additional level of security not available when storing private keys and trusted CAs in files.

Configuring a WebLogic Keystore provider is an optional step. If private keys and digital certificates are stored in files, you do not need to configure the WebLogic Keystore provider.

To configure the WebLogic Keystore provider, you need to create a keystore and set attributes in the Administration Console that point to the keystore.

Note: Weblogic Keystore providers are configured on a domain basis. However, keystores are configured on a per machine basis. If you have multiple security realms or multiple servers running on the same machine, they can all use the same keystore.

To configure the WebLogic Keystore provider:

1. Create a keystore. For more information, see "Creating a Keystore and Loading Private Keys and Trusted CAs into the Keystore" on page 5-12.

2. In the Administration Console, expand the Security-->Realms nodes.
3. Click the name of the realm you are configuring (for example, myrealm).
4. Expand the Providers node.
5. Click Key Stores.

   The Keystore tab appears. This tab displays the name keystore configured for the security realm. By default, the WebLogic Keystore provider is configured.

   Note: The Administration Console refers to the WebLogic Keystore provider as the DefaultKeystore.
6. Click DefaultKeystore.
7. On the General tab, enter the directory location of the keystore in the Private Key Store Location attribute. The default is `wlDefaultKeyStore.jks`.

   This attribute requires both a directory and filename location that is either absolute or relative to the root directory of the server.
8. Enter the password you specified when you created in the keystore in the Private Keystore Pass Phrase attribute.
9. Enter the directory location of the keystore that contains trusted CAs in the Root CA Key Store Location attribute.

   This attribute requires both a directory and filename location that is either absolute or relative to the root directory of the server. This step is not required if both private keys and trusted CAs are stored in the same keystore.
10. Enter the password you specified when you created in the trusted CA keystore in the Private Keystore Pass Phrase attribute.
11. Enter the type of keystore you are using in the Type attribute. The valid types are defined in JavaSoft Cryptography Architecture specification. Set this attribute to null or an empty string to use the default keystoretype configured in the `java.security` file.
12. Click Apply to save your changes.
13. Reboot WebLogic Server.

Note: If you don't want to use the WebLogic Keystore provider to store trusted CAs, you have the option of specifying another keystore location as a command line argument when starting WebLogic Server.

Use the `-Dweblogic.security.SSL.trustedCAKeyStore` command line argument to specify the location of a keystore that contains certificate authorities that the server or client trusts. The path value must be a relative or qualified name to a Sun JKS keystore file. If you do not specify this argument, WebLogic Server trusts all of CA stored in `JAVA_HOME\jre\lib\security\cacerts`.

For more information, see the weblogic.Admin Command-Line Reference.

Weblogic Server looks for trusted CAs in the following sequence:

- Uses the JKS keystore specified in the `-Dweblogic.security.ssl.trustedCAkeystore` command-line argument.
- Uses the keystore specified in the Root CA Key Store Location attribute for the WebLogic Keystore provider. For more information, see "Configuring a WebLogic Role Mapping Provider" on page 3-24.
- Uses a trusted CA file from a 6.x `config.xml` file (if one exists in the domain directory).
- Uses *`WL_HOME`*`\weblogic700\server\lib\cacerts`.

Enabling the SSL Protocol

To enable the SSL protocol, perform the following steps:

1. Expand the Server node.
2. Click the Configure New Server... link.
3. Select the Connections-->SSL Ports tab and set the following attributes
   - The Enabled attribute enables the use of the SSL protocol.
   - The SSL Listen Port attribute is the number of the dedicated port on which WebLogic Server listens for SSL connections. The default is 7002.

4. Click Apply to save your changes.
5. Set attributes for One-way or Two-way SSL. For more information, see "Setting Attributes for One-Way SSL" on page 5-16 or "Setting Attributes for Two-Way SSL" on page 5-17.
6. Reboot WebLogic Server.

Setting Attributes for One-Way SSL

To define attributes for one-way SSL:

1. Enable the SSL protocol. For more information, see "Enabling the SSL Protocol" on page 5-15.
2. Select the Connections-->SSL tab and set the attributes described in Table 5-2.

Table 5-2 SSL Attributes

SSL Attribute	Description
Server Private Key Alias	The alias specified when loading the private key for WebLogic Server into the keystore. Define this attribute only if you stored the private key for WebLogic Server in a keystore.You must have the WebLogic Keystore provider configured to use this attribute. All keystore entries (private key and trusted CAs) are accessed via unique aliases. You specify the alias when loading the private key or trusted CA into the keystore. Aliases are case-insensitive; the aliases `Hugo` and `hugo` would refer to the same keystore entry.

Table 5-2 SSL Attributes

SSL Attribute	Description
Server Private Key Passphrase	The password specified when loading the private key for WebLogic Server into the keystore. Define this attribute only if you stored the private key for WebLogic Server in a keystore.You must have the WebLogic Keystore provider configured to use this attribute.
Server Certificate File Name	The directory location of the digital certificate for WebLogic Server.
Server Key File Name	The directory location of the private key for WebLogic Server. Specify this attribute only if you stored the private key for WebLogic Server in a file.
Trusted CA File Name	The name of the file containing the PEM-encoded trusted certificate authorities.

3. Click Apply to save your changes.
4. Reboot WebLogic Server.

Setting Attributes for Two-Way SSL

Two-way SSL requires clients to present a digital certificate to WebLogic Server in order to establish an SSL connection. The client first authenticates the server's digital certificate and then the server authenticates the client's digital certificate.

To define attributes for two-way SSL:

1. Enable the SSL protocol.
2. Select the Connections-->SSL tab and set attributes for one-way SSL as described in Step 3 of "Setting Attributes for One-Way SSL" on page 5-16.

3. Enable one of the following two-way SSL attributes on the SSL tab:
   - Check the Client Certificate Enforced attribute to require a client to present a certificate. If a certificate is not presented, the SSL connection is terminated.
   - Check the Client Certificate Requested But Not Enforced attribute to ask for a certificate from the client. If this option is enabled, the connection continues if the client does not present a certificate.
4. Click Apply.
5. Reboot WebLogic Server.

Command-Line Arguments for the SSL Protocol

To use a private key stored in a file with WebLogic Server, use the following command-line argument:

`-Dweblogic.management.pkpassword=`*`pkpassword`*

where *`password`* is the password for the private key.

To specify a trusted CA keystore other than the keystore specified in the Root CA Key Store Location attribute in the WebLogic Keystore provider, use the following command line argument:

`-Dweblogic.security.SSL.trustedCAkeystore=`*`path`*

The path value must be a relative or qualified name to a Sun JKS keystore file. If you do not specify this argument, the server or client trusts all of the CA certificates that are in `shipped in the JDK (JAVA_HOME\jre\lib\security\cacerts)`.

Note: The `weblogic.security.SSL.sessionCache.size` and `weblogic.security.SSL.sessionCache.ttl` command-line arguments are not supported in WebLogic Server 7.0. For more information, see "SSL Session Behavior" on page 5-19.

SSL Session Behavior

WebLogic Server allows SSL sessions to be cached. Those sessions live for the life of the server.

not change this behavior in WebLogic Server 7.0. The `weblogic.security.SSL.sessionCache.size` and `weblogic.security.SSL.sessionCache.ttl` command-line arguments are ignored.

Clients that use HTTPS URLs get a new SSL session for each URL because each URL uses a different SSL context and therefore SSL sessions can not be shared or reused. The SSL session can be retrieved using the `weblogic.net.http.HttpsClient` class or the `weblogic.net.http.HttpsURLConnection` class.

Clients that use SSL sockets directly can control the SSL session cache behavior. The SSL session cache is specific to each SSL context. All SSL sockets created by SSL socket factory instances returned by a particular SSL context can share the SSL sessions.

Clients default to resuming sessions at the same IP address and port. Multiple SSL sockets use the same host and port share SSL sessions by default assuming the SSL sockets are using the same underlying SSL context.

Clients that don't want SSL sessions to be used at all need to explicitly call `setEnableSessionCreation(false)` on the SSL socket to ensure that no SSL sessions are cached. This setting only controls whether an SSL session is added to the cache, it does not stop an SSL socket from finding an SSL session that was already cached (for example, SSL socket 1 caches the session, SSL socket 2 sets `setEnableSessionCreation` to `false` but it can still reuse the SSL session but can still reuse the SSL session from SSL socket 1 since that session was put in the cache.)

SSL sessions exist for the lifetime of the SSL context, they are not controlled by the lifetime of the SSL socket. Therefore, creating a new SSL socket and connecting to the same host and port can resume a previous session as long as the SSL socket is created using an SSL socket factory from the SSL context that has the SSL session in its cache.

In WebLogic Server 6.x, there is one SSL session cached in the thread of execution. In WebLogic Server 7.0, session caching is maintained by the SSL context which can be shared by threads. A single thread has access to the entire session cache not just one SSL session so multiple SSL sessions can be used and shared in a single (or multiple) thread.

Using the SSL Protocol in a Cluster

When using the SSL protocol in a cluster, perform the following steps:

1. For each machine in the cluster, create a new domain using the Configuration Wizard. For more information, see Creating Domains and Servers.

   This domain must have the same name on all the machines in the cluster.

2. Obtain a private key and digital certificate for each machine in the cluster. For more information, see "Obtaining Private Keys, Digital Certificates and Trusted CAs" on page 5-5.

3. On each Managed server in the cluster, store the digital certificate for each machine in the cluster in the following locations:

   - *WL_HOME*`/NodeManagerHome/`
   - The directory specified by the `-Dweblogic.RootDirectory=path` command-line argument

   Note: You must use the same directory location and filenames for the digital certificate for each machine in the cluster.

4. Store the private key in a keystore that can be accessed by the WebLogic Keystore provider. For more information, see "Storing Private Keys, Digital Certificates, and Trusted CAs" on page 5-11

Note: If you choose to store the private keys in a keystore, you must use the same directory location and filenames for the keystore and the private key on each machine in the cluster.

5. On each Managed server in the cluster, you need to copy the keystore into the following locations:

- *WL_HOME*/NodeManagerHome/
- The directory specified by the `-Dweblogic.RootDirectory=path` command-line argument

6. Configure the WebLogic Keystore provider for the domain. For more information, see "Configuring the WebLogic Server Keystore Provider to Locate a Keystore" on page 5-13. In the Private Key Store Location attribute, specify the name and directory location of the keystore you created.
7. Enable the SSL protocol. For more information, see "Enabling the SSL Protocol" on page 5-15.
8. Set the Server Private Key Alias, Server Private Key Passphrase, and Server Certificate File Name attributes on the SSL tab. For more information, see "Setting Attributes for One-Way SSL" on page 5-16.
9. Reboot WebLogic Server.

Using a Hostname Verifier

A Hostname Verifier ensures the host to which an SSL connection is made is the intended or authorized party. A Hostname Verifier is useful when WebLogic Server or a WebLogic client is acting as an SSL client to another application server. It prevents man-in-the-middle attacks.

By default, WebLogic Server, as a function of the SSL handshake, compares the common name in the SubjectDN in the SSL server's digital certificate with the host name of the SSL server used to initiate the SSL connection. If these names do not match, the SSL connection is dropped. The SSL client actually drops the SSL connection if the names do not match.

If anything other than the default behavior is desired, either turn off host name verification or register a custom host name verifier. Turning off host name verification leaves WebLogic Server vulnerable to man-in-the-middle attacks.

Turn off host name verification in the following ways:

- In the Administration Console, check the Hostname Verification Ignored attribute under the SSL tab on the Server node.

- On the command line of the SSL client, enter the following argument:

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

You can also write a custom Hostname Verifier. For more information, see *Programming WebLogic Security*. If you write a custom Hostname Verifier, the name of the class that implements the Hostname Verifier must be specified in the CLASSPATH of WebLogic Server.

To use a custom Hostname Verifier:

1. Expand the Server node.
2. Click the Configure New Server... link.
3. Click the Connections-->SSL Ports tab and ensure the SSL protocol is enabled.
4. Enter the name of the Java class used to load your custom Hostname Verifier in the Hostname Verifier attribute on the SSL tab.
5. Click Apply to save your changes.
6. Reboot WebLogic Server.

Configuring RMI over IIOP with SSL

Use the SSL protocol to protect IIOP connections to RMI remote objects. The SSL protocol secures connections through authentication and encrypts the data exchanged between objects.

To use the SSL protocol to protect RMI over IIOP connections, do the following:

1. Configure WebLogic Server to use the SSL protocol.
2. Configure the client Object Request Broker (ORB) to use the SSL protocol. Refer to the product documentation for your client ORB for information about configuring the SSL protocol.
3. Use the `host2ior` utility to print the WebLogic Server IOR to the console. The `host2ior` utility prints two versions of the interoperable object reference (IOR), one for SSL connections and one for non-SSL connections. The header of the IOR specifies whether or not the IOR can be used for SSL connections.

4. Use the SSL IOR when obtaining the initial reference to the CosNaming service that accesses the WebLogic Server JNDI tree.

For more information about using RMI over IIOP, see *Programming WebLogic RMI* and *Programming WebLogic RMI over IIOP*.

6 Configuring Security for a WebLogic Domain

The following sections describe how to set security attributes on a WebLogic domain:

Note: This chapter applies to WebLogic Server deployments using the security features in WebLogic Server 7.0 as well as deployments using Compatibility Security.

Enabling Trust Between WebLogic Domains

A trust relationship is established when principals in a Subject from one WebLogic Server domain (referred to as the domain) are accepted as principals in the local domain.

In WebLogic Server 6.x, the trust relationship between two WebLogic Server domains was established if the system password was the same in both domains. The user identity was not checked after initial authentication. So you could authenticate to the trusted remote domain as `joeuser` and then perform operations locally as `joeuser` even though you had not authenticated to the local domain as `joeuser`.

WebLogic Server 7.0 adds more restrictions to the trust relationship between domains. In WebLogic Server 7.0, a trust relationship is established when the Credential attribute for one domain matches the Credential attribute for another domain.

By default, when you boot an Administration Server for the first time, the Credential attribute is not set. As the Administration Server boots, it notices that the Credential attribute is not set and generates a random credential. The Administration Server uses that credential to sign Principals in Subjects created in that domain. After some period of time (immediately if the Credential attribute on the embedded LDAP server is not set), the `config.xml` file is persisted to disk. Managed servers in that domain download the credential from the Administration Server when booting.

WebLogic Server performs a validation (comparing how the Principal was signed with how a local Principal would be signed) whenever the code is asked to create a new Subject.

If a WebLogic Server 7.0 deployment is acting as a client and authenticates to another domain, the Credential attributes of the two domains must match.

Note: Any credentials in clear text are encrypted the next time the `config.xml` file is persisted to disk.

If you want a WebLogic Server 6.x domain to interoperate with a WebLogic Server 7.0 domain, change the Credential attribute in both domains to the password of the `system` user in the WebLogic Server 6.x domain.

If you want two 7.0 domains to interoperate, perform the following procedure in both domains.

To establish a trust relationship between WebLogic Server domains:

1. Expand the Domains node.
2. Select the Security-->Advanced tab.
3. Click the Change... link in the Credential attribute.
4. Enter a password for the domain. Choose the password carefully. BEA Systems recommends using a combination of upper and lower case letters and numbers.
5. Confirm the password.
6. Click Apply.
7. Reboot WebLogic Server.

Configuring Connection Filtering

Create connection filters that allow you to reject or accept client connections based on the client's origin and protocol thus adding an additional level of security. For example, write a connection filter that rejects any non-SSL connections originating outside of your corporate network. After a client connects, and before any work is performed on its behalf, WebLogic Server passes the client's IP number and port, protocol (HTTP, HTTPS, T3, T3S, or IIOP), and WebLogic Server port number to the connection filter. By examining this information, you can choose to allow the connection or throw a `Filter Exception` to terminate it.

For information about writing a connection filter, see Writing a Network Connection Filter in *Programming WebLogic Security*.

To configure a connection filter:

1. Expand the Domains node.
2. Select the Security-->Filter tab.
3. Enter the class that implements the network connection filter in the Connection Filter attribute. This class must also be specified in the CLASSPATH for WebLogic Server.
4. Enter the syntax for the connection filter rules. For more information about connection filter rules, see Writing a Network Connection Filter in *Programming WebLogic Security.*
5. Click Apply.
6. Reboot WebLogic Server.
7. Expand the Domains node.
8. Click the Security tab.
9. Click the Advanced tab.
10. Click the Connection Logger Enabled attribute to enable the logging of accepted messages.
11. Click Apply.

When WebLogic Server acts a client to an untrusted WebLogic Server (for example, a server in a different domain), the remote credentials cannot be used for accessing locally protected resources. When running WebLogic Server 6.x code on WebLogic Server 7.0, this behavior is seen after using a JNDI InitialContext to perform remote operations and then attempting to use the same JNDI InitialContext to perform local operations.

When local and remote operations are required at the same time on the same server, the following options are available:

1. Use to the Java Authentication and Authorization Service (JAAS) Authentication available in WebLogic Server 7.0. In JAAS, remote credentials are supplied via a JAAS LoginModule. Use the following method to access the resulting Subject:

   ```
   weblogic.security.Security.runAs(Subject, PrivilegedAction)
   ```

   where `PrivilegedAction` represents the remote operation that is to be performed with the remote credentials.

2. After establishing the JNDI InitialContext, use the following method to obtain the remote credentials as a JAAS Subject:

   ```
   weblogic.security.Security.getCurrentSubject()
   ```

   After using the remote credentials, close the JNDI InitialContext and use the Subject as described in Option 1.

Note: The JNDI method of pass security credentials in an InitialContext was deprecated in WebLogic Server 6.x.

CHAPTER

7 Using the Java Security Manager

The Java security manager can be used with WebLogic Server to provide additional protection for resources running in a Java Virtual Machine (JVM). Using a Java Security Manager is an optional security step. The following sections describe how to use the Java security manager with WebLogic Server:

Setting Up the Java Security Manager

When you run WebLogic Server under Java 2 (JDK 1.2 or 1.3), WebLogic Server can use the Java security manager in Java 2 which prevents untrusted code from performing actions that are restricted by the Java security policy file.

The JVM has security mechanisms built into it that allow you to define restrictions to code through a Java security policy file. The Java security manager uses the Java security policy file to enforce a set of permissions granted to classes. The permissions allow specified classes running in that instance of the JVM to permit or not permit certain runtime operations. In many cases, where the threat model does not include malicious code being run in the JVM, the Java security manager is unnecessary. However, when third-parties use WebLogic Server and unknown classes are being run, the Java security manager may be useful.

To use the Java security manager with WebLogic Server, specify the `-Djava.security.policy` argument when starting WebLogic Server. The `-Djava.security.policy` argument specifies a filename (using a relative or fully-qualified pathname) that contains Java 2 security policies.

WebLogic Server provides sample Java security policy file, which you can edit and use. The file is located in *`WL_HOME`*`\server\lib\weblogic.policy`.

If you enable the Java security manager but do not specify a security policy file, the Java security manager uses the default security policies defined in the `java.security` and `java.policy` files in the `$JAVA_HOME/jre/lib/security` directory.

Define security policies for the Java security manager in one of the following ways:

- "Setting Application-Type Security Policies" on page 7-3
- "Setting Application-Specific Security Policies" on page 7-4

Modifying the weblogic.policy file for General Use

To use the Java security manager security policy file with your WebLogic Server deployment, set the following arguments on the Java command line when you start WebLogic Server:

- `java.security.manager` tells the JVM to use a Java security policy file.
- `java.security.policy` tells the JVM the location of the Java security policy file to use. The argument is the fully qualified name of the Java security policy, in this case, weblogic.policy.

For example:

```
$ java...-Djava.security.manager
-Djava.security.policy==c:/weblogic/weblogic.policy
```

Be sure to use == instead of = when specifying the `java.security.policy` argument so that only the `weblogic.policy` file is used by the Java security manager. The == causes the `weblogic.policy` file to override any default security policy. A single equal sign (=) causes the `weblogic.policy` file to be appended to an existing security policy.

If you have extra directories in your `CLASSPATH` or if you are deploying applications in extra directories, add specific permissions for those directories to your `weblogic.policy` file.

BEA recommends taking the following precautions:

- Make a backup copy of the `weblogic.policy` file and put the backup copy in a secure location.
- Set the permissions on the `weblogic.policy` file for the operating system file such that the administrator of the WebLogic Server deployment has write and read privileges and no other users have access to the file.

Caution: The Java security manager is partially disabled during the booting of Administration and Managed Servers. During the boot sequence, the current Java security manager is disabled and replaced with a variation of the Java security manager that has the `checkRead()` method disabled. While disabling this method greatly improves the performance of the boot sequence, it also minimally diminishes security. The startup classes for WebLogic Server are run with this partially disabled Java security manager and therefore the classes need to be carefully scrutinized for security considerations involving the reading of files.

For more information about the Java security manager, see the Javadoc shipped with the JDK.

Setting Application-Type Security Policies

Note: You cannot modify security policies for Web applications.

Set default security policies for EJBs, and J2EE Connector Resource Adapters in the Java security policy. The default security policies for EJBs, and Resource Adapters are defined in the Java security policy file under the following codebases:

- EJBs—`"file:/weblogic/application/defaults/EJB"`
- Resource Adapters—`"file:/weblogic/application/defaults/Connectors"`

Note: These security policies apply to all EJBs, and Resource Adapters deployed in that particular instance of WebLogic Server.

Setting Application-Specific Security Policies

Note: You cannot modify security policies for Web applications.

Set security policies for a specific EJB, or Resource adapter by adding policies to their deployment descriptors. Deployment descriptors are defined in the following files:

- EJBs—`weblogic-ejb-jar.xml`
- Resource adapters—`rar.xml`

Note: The security policies for Resource Adapters follow the J2EE standard while the security policies for EJBs follow the WebLogic Server extension to the J2EE standard.

Use the following syntax to add a security policy to a deployment descriptor:

```
<security-permission>
  <description>
    //Protect foo operation
  </description>
  <security-permission-spec>
    //Grant statements
    grant {
      permission java.lang.RuntimePermission(“foo”);
      permission otherPermission;
           }
  </security-permission-spec>
<security-permission>
```

Note: The `security-permission-spec` attribute cannot currently be added to a `weblogic-application.xml` file, you are limited to using this attribute within a `weblogic-ejb-jar.xml` or `weblogic.xml` file.

Variables are not supported in the `security-permission-spec` attribute.

Using the Recording Security Manager Utility

The Recording Security Manager utility can be used to detect permission problems that occur when starting and running WebLogic Server. The utility outputs permissions that can be added to your Java security policy file to resolve the permission problems that the utility finds. The Recording Security Manager is available at the BEA dev2dev Online.