BEA WebLogic Server™

Securing WebLogic Resources

Release 7.0
Document Revised: July 18, 2003

Copyright

Restricted Rights Legend

Trademarks or Service Marks

Securing WebLogic Resources

Part Number	Document Revised	Software Version
N/A	July 18, 2003	BEA WebLogic Server Version 7.0

Contents

About This Document

1. Introduction to Securing WebLogic Resources

2. Types of WebLogic Resources

3. Users and Groups

4. Security Roles

5. Security Policies

6. Example: Securing URL (Web) Resources Using the Administration Console

7. Example: Securing Enterprise JavaBean (EJB) Resources

8. Examples: Copying and Reinitializing Security Configurations for the basicauth Web Application

Index

About This Document

This document introduces the various types of WebLogic resources, and provides information that allows you to secure these resources using WebLogic Server.

The document is organized as follows:

- Chapter 1, "Introduction to Securing WebLogic Resources," which provides introductory information about this document (such as its intended audience), an overview of securing WebLogic resources, and a "main steps" section for those wanting to skip directly to instructional sections.
- Chapter 2, "Types of WebLogic Resources," which describes each of the different types of WebLogic resources and provides important information about securing some of the more complex and common WebLogic resources.
- Chapter 3, "Users and Groups," which provides descriptions of users and groups, and includes information about WebLogic Server's default groups. This section also includes step-by-step instructions that tell you how to work with users and groups in the WebLogic Server Administration Console.
- Chapter 4, "Security Roles," which provides a description of a security role, and includes information about WebLogic Server's default global roles. This section also explains the difference between global and scoped roles and describes the components of a security role. Last, this section includes step-by-step instructions that tell you how to work with global and scoped roles in the Administration Console.
- Chapter 5, "Security Policies," which provides a description of a security policy, and includes information about WebLogic Server's default security policies. This section also describes the components of a security policy, and provides step-by-step instructions that tell you how to work with security policies in the Administration Console.

- Chapter 6, "Example: Securing URL (Web) Resources Using the Administration Console," which steps you through securing various URL (Web) resources using the Administration Console.
- Chapter 7, "Example: Securing Enterprise JavaBean (EJB) Resources," which steps you through securing various Enterprise JavaBean (EJB) resources using the Administration Console.
- Chapter 8, "Examples: Copying and Reinitializing Security Configurations for the basicauth Web Application," which steps you through copying a Web application's security configuration from existing deployment descriptors, modifying a security policy using the Administration Console, then reinitializing the security configuration back to what it had been in the deployment descriptors.

Audience

This document is written primarily for Server Administrators. **Server Administrators** work closely with ApplicationArchitects to design a security scheme for the server and the applications running on the server, to identify potential security risks, and to propose configurations that prevent security problems. Related responsibilities may include maintaining critical production systems, configuring and managing security realms, implementing authentication and authorization schemes for server and application resources, upgrading security features, and maintaining security provider databases. Server Administrators have in-depth knowledge of the Java security architecture, includingWeb application and EJB security, Public Key security, and SSL.

This document is targeted toward Server Administrators who use the WebLogic Server Administration Console, and should be used in conjunction with *Managing WebLogic Security* to ensure that security is completely configured for a WebLogic Server deployment.

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. Other WebLogic Server documents that may be of interest to Server Administrators wanting to secure WebLogic resources are:

- *Managing WebLogic Security*
- "Securing Web Applications," "Securing Enterprise JavaBeans (EJBs)," and "Using Java Security to Protect WebLogic Resources" in *Programming WebLogic Security.*

- "Protecting System Administration Operations" in the *WebLogic Server Administration Guide*.
- "Configure Access Control" in *Programming WebLogic jCOM* (COM resources).
- "Security" in *Programming WebLogic J2EE Connectors* (EIS resources).
- "Configuring Security" in *Programming WebLogic Web Services* (Web Service resources).

Additional security documents are listed on the Security page.

Contact Us!

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

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

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

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

Documentation Conventions

The following documentation conventions are used throughout this document.

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

`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.
	*Example*s:
	`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 Introduction to Securing WebLogic Resources

The following sections prepare you to learn more about securing WebLogic resources:

Audience for This Guide

This document is written primarily for Server Administrators. **Server Administrators** work closely with Application Architects to design a security scheme for the server and the applications running on the server, to identify potential security risks, and to propose security configurations that prevent security problems. Related responsibilities may include maintaining critical production systems, configuring and managing security realms, implementing authentication and authorization schemes for server and application resources, upgrading security features, and maintaining security provider databases. Server Administrators have in-depth knowledge of the Java security architecture, including Web application and EJB security, Public Key security, and SSL.

This document is targeted toward Server Administrators who use the WebLogic Server Administration Console, and should be used in conjunction with *Managing WebLogic Security* to ensure that security is completely configured for a WebLogic Server deployment.

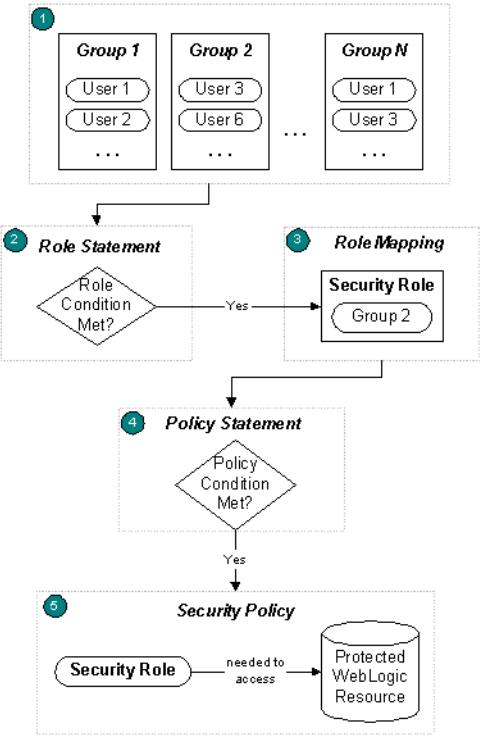
Terms and Concepts

WebLogic Server security includes many unique terms and concepts that you need to understand. These terms and concepts—which you will encounter throughout the WebLogic Server security documentation—are defined in the Terminology section and the Security Fundamentals section of *Introduction to WebLogic Security*, respectively.

Overview of Securing WebLogic Resources

Figure 1-1 illustrates the overall process for securing WebLogic resources, and a brief explanation follows.

Figure 1-1 Securing WebLogic Resources



1. Administrators statically assign users to groups, which can represent organizational boundaries. The same user can be a member of multiple groups. Figure 1-1 shows three groups with two users each. User 1 and User 3 are members of multiple groups.

   Note: BEA recommends assigning users to groups because doing so is more efficient for administrators who work with large numbers of users.

2. Administrators create a role statement based on their organization's established business procedures. The role statement includes a role condition that specifies when a particular group should be granted the security role.

3. At runtime, the WebLogic Security Service compares the groups against the role condition to determine whether they should be dynamically granted a security role. This process is referred to as **role mapping**. In Figure 1-1, Group 2 is the only group that is granted a security role.

Note: Individual users could also be granted a security role, but this is a less typical practice.

4. Administrators create a policy statement based on their organization's established business procedures. The policy statement includes a policy condition that specifies when a particular security role should be granted access to a protected WebLogic resource.
5. At runtime, the WebLogic Security Service uses the security policy and the WebLogic resource itself to determine whether access to the protected WebLogic resource should be granted. Only users who are members of the group that is granted the security role can access the WebLogic resource. In Figure 1-1, User 3 and User 6 can access the protected WebLogic resource because they are members of Group 2.

Securing WebLogic Resources: Main Steps

The main steps for securing a WebLogic resource are as follows:

1. Determine which WebLogic resource to secure. See "Types of WebLogic Resources" for more information.
2. If you want to secure a URL (Web) or Enterprise JavaBean (EJB) resource:
   a. Decide which technique you will use. See "Techniques for Securing URL and EJB Resources" on page 2-7 for more information.
   b. Review important information about securing URL and EJB resources to prevent overriding security configurations. See "Prerequisites for Securing URL and EJB Resources" on page 2-9 for more information.
   c. If you want to use the WebLogic Server Administration Console to secure your URL or EJB resource, follow the instructions in step 3.
   d. If you want to use deployment descriptors to secure your URL or EJB resource, see Using Declarative Security with Web Applications or Using Declarative Security with EJBs in *Programming WebLogic Security*, respectively.

e. If you want to copy security configurations from existing deployment descriptors upon the initial deployment of URL or EJB resources, or reinitialize the security configuration for URL or EJB resources to their original state (as specified in the deployment descriptors), follow the instructions in "Using the Combined Technique to Secure Your URL and EJB Resources" on page 2-17.

3. Use the Administration Console to secure your WebLogic resource. Follow these steps:

   a. Create users and groups—representations of individuals and collections of individuals—who may be granted a security role. See "Creating Users" on page 3-2 and "Creating Groups" on page 3-7 for step-by-step instructions.

   b. Create security roles—dynamically computed privileges granted to users or groups based on specific conditions—which are used to restrict access to WebLogic resources. See "Security Roles" for step-by-step instructions.

      Note: BEA recommends creating security roles and using them (rather than users or groups) to secure WebLogic resources, because doing so makes it more efficient for administrators who work with large numbers of users.

   c. Create a security policy—an association between the WebLogic resource and a user, group, or security role—that specifies who has access to the WebLogic resource. See "Security Policies" for step-by-step instructions.

CHAPTER

2 Types of WebLogic Resources

A **WebLogic resource** represents an underlying WebLogic Server entity that can be protected from unauthorized access using security roles and security policies.

WebLogic resources are hierarchical. Therefore, the level at which you define these security roles and security policies is up to you. For example, you can define security roles and security policies on an entire enterprise application (EAR), an Enterprise JavaBean (EJB) JAR containing multiple EJBs, a particular EJB within that JAR, or a single method within that EJB.

WebLogic Server defines the resources described in the following sections:

Administrative Resources

An **Administrative resource** is a type of WebLogic resource that allows users to perform administrative tasks. You secure Administrative resources when you want to protect the WebLogic Server Administration Console, the weblogic.Admin tool, and MBean APIs.

Administrative resources are limited in scope. Currently, you can only secure the User Lockout operation on an Administrative resource using the WebLogic Server Administration Console. This operation provides compatibility with WebLogic Server 6.x., and allows users who meet the security requirements to unlock users who have been locked out of their accounts. For more information about user lockout, see "Protecting User Accounts" in *Managing WebLogic Security*.

Application Resources

An **Application resource** is a type of WebLogic resource that represents an enterprise application, packaged as an EAR (Enterprise Application aRchive) file. Unlike the other types of WebLogic resources, the hierarchy of an Application resource is a mechanism for containment, rather than a type hierarchy. You secure an Application resource when you want to protect multiple WebLogic resources that *comprise* the enterprise application (for example, EJB resources, URL resources, and Web Service resources) *at one time*. In other words, securing an enterprise application will cause all the WebLogic resources within that application to inherit its security configuration.

You can also secure the WebLogic resources that comprise an enterprise application (EAR) individually. Doing both will cause the security configuration inherited from the enterprise application to be overridden for that WebLogic resource.

EIS (Enterprise Information System) Resources

A J2EE Connector is a system-level software driver used by an application server such as WebLogic Server to connect to an EIS (Enterprise Information System). BEA supports Connectors developed by EIS vendors and third-party application developers that can be deployed in any application server supporting the Sun Microsystems J2EE Platform Specification, Version 1.3. Connectors, also known as **Resource Adapters**, contain the Java, and if necessary, the native components required to interact with the EIS.

An **EIS (Enterprise Information System) resource** is a specific type of WebLogic resource that is designed as a Connector. To secure access to an EIS, you can create security policies and security roles for all Connectors as a group, or individual Connectors.

Notes: Information about securing EIS resources can be found both in this document, and in the "Security" section of *Programming WebLogic J2EE Connectors.*

Instructions for creating the credential maps for use with EIS resources are available in the "Single Sign-On with Enterprise Information Systems" section of *Managing WebLogic Security*.

COM Resources

WebLogic jCOM is a software bridge that allows bidirectional access between Java/J2EE objects deployed in WebLogic Server, and Microsoft ActiveX components available within Microsoft Office family of products, Visual Basic and C++ objects, and other Component Object Model/Distributed Component Object Model (COM/DCOM) environments.

A **COM resource** is a specific type of WebLogic resource that is designed as a program component object according to Microsoft's framework. To secure COM components accessed through BEA's bi-directional COM-Java (jCOM) bridging tool, you can create security policies and security roles for packages containing multiple COM classes, or individual COM classes.

Note: Information about securing COM resources can be found both in this document and in the "Configuring Access Control" section of *Programming WebLogic jCOM*.

JDBC (Java DataBase Connectivity) Resources

A **JDBC (Java DataBase Connectivity) resource** is a specific type of WebLogic resource that is related to JDBC. To secure JDBC database access, you can create security policies and security roles for all connection pools as a group, individual connection pools, and MultiPools. When you secure individual connection pools, you can choose whether to protect all operations on the connection pool, or specify one of the following operations:

- *admin*—the following methods on the `JDBCConnectionPoolRuntimeMBean` are invoked as admin operations: `clearStatementCache`, `destroy`, `disableDroppingUsers`, `disableFreezingUsers`, `enable`, `forceDestroy`, `forceShutdown`, `forceSuspend`, `getProperties`, `poolExists`, `resume`, `shutdown`, `shutdownHard`, `shutdownSoft`, and `suspend`.
- *reserve*—Applications reserve a connection in the connection pool by looking up the data source that points to the connection pool and then calling getConnection.

  Note: Giving a user the reserve permission enables them to execute vendor-specific operations on the connection. Depending on the database vendor, some of these operations may have database security implications.
- *shrink*—Shrinks the connection pool to the maximum of the currently reserved connections or the initial size.

- *reset*—Resets the database connection pool by shutting down and re-establishing all physical database connections. This also clears the statement cache for each connection in the connection pool. You can only reset a normally running connection pool.

Note: If a security policy controls access to a connection pool that is in a MultiPool, access checks will be performed at both levels of the JDBC resource hierarchy (once at the MultiPool level, and again at the individual connection pool level). As with all types of WebLogic resources, this double checking ensures that the most restrictive security policy controls access.

JMS (Java Messaging Service) Resources

A **JMS (Java Messaging Service) resource** is a specific type of WebLogic resource that is related to JMS. To secure JMS destinations, you can create security policies and security roles for all destinations (JMS queues and JMS topics) as a group, or an individual destination (JMS queue or JMS topic) on a JMS server. When you secure a particular destination on a JMS server, you can choose whether to protect all operations on the destination, or specify the send, browse, or receive operations (for a JMS queue) and the send and receive operations (for a JMS topic).

JNDI (Java Naming and Directory Interface) Resources

JNDI provides a common-denominator interface to many existing naming services, such as LDAP (Lightweight Directory Access Protocol) and DNS (Domain Name System). These naming services maintain a set of bindings, which relate names to objects and provide the ability to look up objects by name. JNDI allows the components in distributed applications to locate each other.

JNDI is defined to be independent of any specific naming or directory service implementation. It supports the use of a number of methods for accessing various new and existing services. This support allows any service-provider implementation to be plugged into the JNDI framework using the standard service provider interface (SPI) conventions.

A **JNDI (Java Naming and Directory Interface) resource** is a specific type of WebLogic resource that uses the industry-standard JNDI SPI to enable connectivity to heterogeneous enterprise naming and directory services. To secure access to the JNDI tree, you can create security policies and security roles for the entire JNDI tree, or an individual branch of that tree. Regardless, you can choose whether to protect all operations, or specify the lookup, modify, or list operations.

Server Resources

Many engineering teams divide administration responsibilities into distinct roles. Each project might give only one or two team members permission to deploy applications or modules, but allow all team members to view the server configuration. As described in Security Policies, WebLogic Server supports this division of responsibility by allowing administrators to secure WebLogic resources with security policies. Typically, these security policies are based on whether users or groups of users are granted a particular security role.

A **Server resource** is a specific type of WebLogic resource that is related to WebLogic Server instances. This type of WebLogic resource includes operations that start, shut down, lock, or unlock servers, and is therefore the type of WebLogic resource with which most administrators interact. Like other types of WebLogic resources, a Server resource and its operations are secured with security policies. However, because the configuration of a server is exposed through a set of MBeans, a Server resource also has additional protections that affect how administrators access MBean operations.

Note: Information about securing Server resources can be found both in this document and in the "Protecting System Administration Operations" section of the *WebLogic Server Administration Guide*.

URL (Web) and EJB (Enterprise JavaBean) Resources

A **URL (Web) resource** is a specific type of WebLogic resource that is related to Web applications. To secure Web applications, you can create security policies and security roles for a WAR (Web Application aRchive) file or individual components of a Web application (such as servlets and JSPs). An **EJB (Enterprise JavaBean) resource** is a specific type of WebLogic resource that is related to EJBs. To secure EJBs, you can create security policies and security roles for EJB JARs, individual EJBs within an EJB JAR, or individual methods on an EJB.

Because the Java 2 Enterprise Edition (J2EE) platform standardizes Web application and EJB security in deployment descriptors, WebLogic Server integrates this standard mechanism with its Security Service to give you a choice of two techniques for securing URL and EJB resources. The technique you choose will affect the procedure you will follow, and will require different prerequisite settings in the WebLogic Server Administration Console. For more information, see "Techniques for Securing URL and EJB Resources" on page 2-7 and "Prerequisites for Securing URL and EJB Resources" on page 2-9, respectively.

Note: The instructions for EJB resources provided in this document also apply to Message-driven Beans (MDBs).

Techniques for Securing URL and EJB Resources

The following sections describe the different techniques for securing URL (Web) and EJB (Enterprise JavaBean) resources in more detail:

- "Using the WebLogic Server Administration Console" on page 2-8
- "Using Deployment Descriptors" on page 2-8
- "Combining the Two Techniques" on page 2-9

Using the WebLogic Server Administration Console

The first technique for securing your URL and EJB resources is to use the WebLogic Server Administration Console. The primary benefit of this technique is unified security management. Instead of requiring developers to modify multiple deployment descriptors when organizational security requirements change, administrators can modify all security configurations from a centralized, graphical user interface. Users, groups, security roles, and security policies can all be defined using the Administration Console. As a result, the process of making changes based on updated security requirements becomes more efficient.

With the exception of Web Service resources, you can secure all types of WebLogic resources using this technique. Therefore, the instructions for securing WebLogic resources contained within this document are written specifically for users of the Administration Console.

Using Deployment Descriptors

The second technique for securing your URL and EJB resources is to use the Java 2 Enterprise Edition (J2EE) and WebLogic deployment descriptors. The primary benefit of this technique is that it is a widely-known, standard technique for adding declarative security to URLs and EJBs. It may also be the technique with which most organizations are familiar. This technique implies that users and groups may be defined in the WebLogic Server Administration Console, but that security roles and security policies may be specified in the `web.xml`, `weblogic.xml`, `ejb-jar.xml`, and `weblogic-ejb-jar.xml` deployment descriptors.

Notes: When using this technique, you can edit deployment descriptors using a text editor of your choice, or by using the Administration Console. (See the "Web Application Deployment Descriptor Editor Help" and "Using the EJB Deployment Descriptor Editor" in *Programming WebLogic Enterprise JavaBeans* for more information on using the Administration Console.)

Starting in WebLogic Server 7.0 SP02, there is a special tag called `<global-role/>` that allows you to specify on a role-by-role basis whether a security role mapping is defined in the deployment descriptors or in the Administration Console. For more information about using this tag with URL or EJB resources, see "Using the <global-role/> Tag with Web Applications" or "Using the <global-role/> Tag with EJBs" in *Programming WebLogic Security*, respectively.

You can secure only URL and EJB resources using this technique. The instructions for using deployment descriptors are available in the "Using Declarative Security with Web Applications" and "Using Declarative Security with EJBs" sections of *Programming WebLogic Security*, respectively.

Combining the Two Techniques

Some organizations currently using deployment descriptors to secure their URL and EJB resources may want to take advantage of the unified security management capabilities that the WebLogic Server Administration Console provides. In a situation like this, the Administration Console can be instructed to copy security configurations from existing deployment descriptors upon the initial deployment of a Web application or EJB module. Once these security configurations are copied, the Administration Console can be used for subsequent updates. It is also possible to use this combined technique to reinitialize security configurations for URL and EJB resources to the state specified in the deployment descriptors.

Warning: When using the combined technique, it is possible to override security configurations for URL and EJB resources. Therefore, administrators using the combined technique need to take extra care to ensure that the appropriate security configuration is in place for their Web applications and EJBs. Read "Prerequisites for Securing URL and EJB Resources" on page 2-9 for important information.

For more information about using the combined technique, see Chapter 8, "Examples: Copying and Reinitializing Security Configurations for the basicauth Web Application."

Prerequisites for Securing URL and EJB Resources

Whether using the WebLogic Server Administration Console or deployment descriptors to secure your URL and EJB resources, there are two important settings in WebLogic Server that you need to understand. Failure to understand these settings could result in incorrect or lost security configurations.

Before attempting to secure URL or EJB resources, read the following sections:

- "Understanding the fullyDelegateAuthorization Flag" on page 2-10
- "How to Change the fullyDelegateAuthorization Flag" on page 2-11

Understanding the fullyDelegateAuthorization Flag

To give you control over performance, the WebLogic Server Administration Console requires you to specify how the WebLogic Security Service should perform security checks. You specify this preference using the `fullyDelegateAuthorization` flag, a command-line argument that you set when you start WebLogic Server.

Note: Starting in WebLogic Server 7.0 SP3, you can choose to specify this preference in the WebLogic Server Administration Console *instead* of using the command-line argument. For more information, see "The Check Roles and Policies Setting" on page 2-14.

When the value of the `fullyDelegateAuthorization` flag is `false`, the WebLogic Security Service *only* performs security checks on URL and EJB resources that have security specified in their associated deployment descriptors (DDs). This is the default setting.

When the value of the `fullyDelegateAuthorization` flag is `true`, the WebLogic Security Service performs security checks on *all* URL (Web) and EJB resources, regardless of whether there are any security settings in the deployment descriptors (DDs) for these WebLogic resources. If you change the value of the `fullyDelegateAuthorization` flag to `true`, you also need to specify what the WebLogic Security Service should do when the Web application or EJB module is redeployed. For more information, see "Understanding the Ignore Security Data in Deployment Descriptors Check Box" on page 2-14.

Note: The `fullyDelegateAuthorization` flag only has an effect on the WebLogic Server instances (servers) in which it is set. Therefore, you should ensure that the `fullyDelegateAuthorization` flag is set the same way for both your Administration and Managed Servers.

How to Change the fullyDelegateAuthorization Flag

You can set the `fullyDelegateAuthorization` flag in one of three ways:

- Type:

  `-Dweblogic.security.fullyDelegateAuthorization=`*`boolean_value`*

  where *`boolean_value`* is `true` or `false` on the command line each time you start a WebLogic Server instance.

- Edit the `startWLS` script (located in the *`WL_HOME`*`\server\bin` directory) to include the following:

  `-Dweblogic.security.fullyDelegateAuthorization=`*`boolean_value`*

  where *`boolean_value`* is `true` or `false`. This is more efficient because it will save you from having to set the flag each time you start a WebLogic Server instance. However, keep in mind that this setting will apply to all WebLogic Server domains in the installation.

  Listing 2-1 shows the appropriate portion of the `startWLS` file with this flag set to true (in bold).

Listing 2-1 Sample startWLS Script

```
@rem Start Server

@echo off
if "%ADMIN_URL%" == "" goto runAdmin
@echo on
"%JAVA_HOME%\bin\java" %JAVA_VM% %MEM_ARGS% %JAVA_OPTIONS%
-Dweblogic.Name=%SERVER_NAME% -Dbea.home="d:\bea"
-Dweblogic.management.username=%WLS_USER%
-Dweblogic.management.password=%WLS_PW%
-Dweblogic.management.server=%ADMIN_URL%
-Dweblogic.ProductionModeEnabled=%STARTMODE%
-Djava.security.policy="%WL_HOME%\server\lib\weblogic.policy"
-Dweblogic.security.fullyDelegateAuthorization=true
weblogic.Server
goto finish

:runAdmin
@echo on
"%JAVA_HOME%\bin\java" %JAVA_VM% %MEM_ARGS% %JAVA_OPTIONS%
-Dweblogic.Name=%SERVER_NAME% -Dbea.home="d:\bea"
```

```
-Dweblogic.management.username=%WLS_USER%
-Dweblogic.management.password=%WLS_PW%
-Dweblogic.ProductionModeEnabled=%STARTMODE%
-Djava.security.policy="%WL_HOME%\server\lib\weblogic.policy"
-Dweblogic.security.fullyDelegateAuthorization=true
weblogic.Server

:finish

ENDLOCAL
```

Note: Add the lines shown in bold in both the main and `runAdmin` sections of the `startWLS` file, unless the script no longer contains both sections because of modifications.

- Edit the `startWebLogic` script (located in the *`WL_HOME`*`\user_projects\`*`domain`* directory, where *`domain`* is the name of a WebLogic Server domain you created) to include the following in the `JAVA_OPTIONS` section:

  ```
  set JAVA_OPTIONS=...
  -Dweblogic.security.fullyDelegateAuthorization=true
  ```

  This method allows you to set the `fullyDelegateAuthorization` flag for each WebLogic Server domain, rather than all the domains in the installation.

  Listing 2-2 shows the appropriate portion of the `startWebLogic` file with this flag set to true (in bold).

Listing 2-2 Sample startWebLogic Script Excerpt

```
@rem Set JAVA_OPTIONS to the java flags you want to pass to the vm. i.e.:
@rem set JAVA_OPTIONS=-Dweblogic.attribute=value -Djava.attribute=value

set JAVA_OPTIONS=-Dweblogic.security.SSL.trustedCAKeyStore=C:\bea_sp02_7a\
     weblogic700\server\lib\cacerts
     -Dweblogic.security.fullyDelegateAuthorization=true
```

Using Node Manager

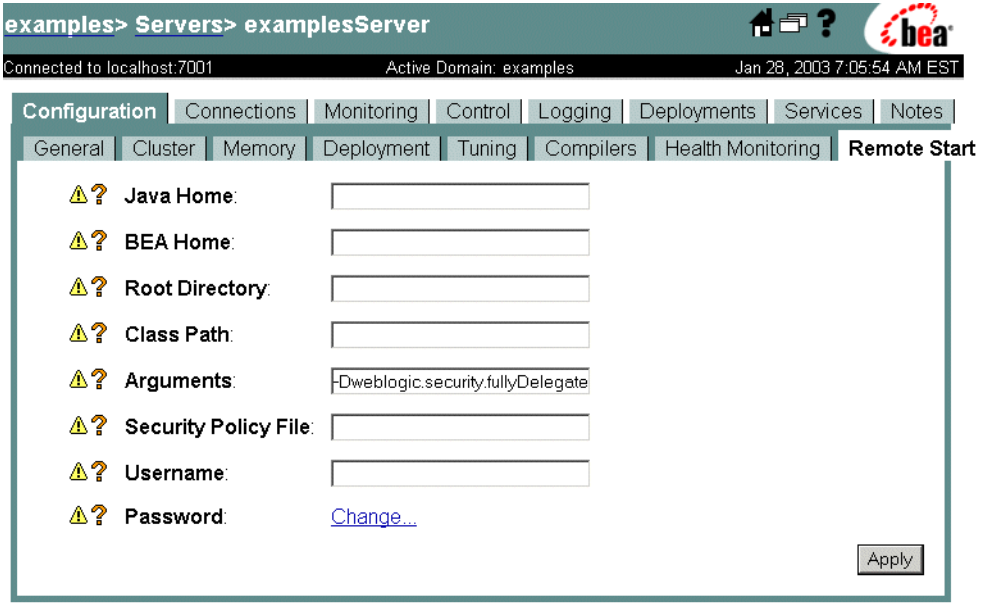
If you are using Node Manager to start your Managed Servers, the start scripts previously described are not used. Therefore, you will need to set the `fullyDelegatedAuthorization` flag using the WebLogic Server Administration Console.

To do this, click the Configuration →Remote Start tab that corresponds to your server, and specify the following in the Arguments field:

```
-Dweblogic.security.fullyDelegatedAuthorization=true
```

Figure 2-1 shows the appropriate field in the Administration Console for the `examplesServer`, with this flag set to true.

Figure 2-1 Configuration > Remote Start Tab for the ExamplesServer



Be sure to click the Apply button and restart your server to save your changes.

The Check Roles and Policies Setting

Prior to WebLogic Server 7.0 SP3, you had to specify how the WebLogic Security Service would perform security checks using the `fullyDelegateAuthorization` command-line argument, as described in "How to Change the fullyDelegateAuthorization Flag" on page 2-11. In WebLogic Server 7.0 SP3, the Check Roles and Policies setting was introduced to allow you to specify the *same* preference in the WebLogic Server Administration Console. The Check Roles and Policies drop-down menu is located on the General tab after clicking Security → Realms →*`myrealm`* (or the name of a security realm you created).

Note: The Check Roles and Policies setting affects all the WebLogic Server instances (servers) in the WebLogic Server domain in which it is set.

Understanding the Ignore Security Data in Deployment Descriptors Check Box

If you decide that the WebLogic Security Service should perform security checks on all Web applications and EJBs by setting the `fullyDelegateAuthorization` flag to `true`, you also need to tell WebLogic Server which technique you want to use to secure these URL (Web) and EJB resources. (See "Techniques for Securing URL and EJB Resources" on page 2-7 for more information.) You specify this preference using the Ignore Security Data in Deployment Descriptors check box.

Note: Starting in WebLogic Server 7.0 SP3, this setting appears as the Deployment Descriptor Security Behavior drop-down menu in the WebLogic Server Administration Console. The Seed Security From Deployment Descriptor value is equivalent to an *unchecked* Ignore Security Data in Deployment Descriptors check box; The Ignore Security Data in Deployment Descriptor is equivalent to a *checked* Ignore Security Data in Deployment Descriptors check box.

You should set the value of the Ignore Security Data in Deployment Descriptors check box as follows:

- To secure your URL (Web) and EJB resources using *only* the WebLogic Server Administration Console, *check* the Ignore Security Data in Deployment Descriptors check box and follow steps 3a - 3c in "Securing WebLogic Resources: Main Steps" on page 1-4.
- To secure your URL (Web) and EJB resources using *only* the deployment descriptors (that is, the `ejb-jar.xml`, `weblogic-ejb-jar.xml`, `web.xml`, and

`weblogic.xml` files), *uncheck* the Ignore Security Data in Deployment Descriptors check box and refer to the "Using Declarative Security with Web Applications" and "Using Declarative Security with EJBs" sections of *Programming WebLogic Security*, respectively.

Warning: Switching the value of the Ignore Security Data in Deployment Descriptors check box is risky and can lead to incorrect or lost security configurations. If you need to switch between these settings (specifically for the situations described in "Combining the Two Techniques" on page 2-9), you can *carefully* follow the instructions in "Using the Combined Technique to Secure Your URL and EJB Resources" on page 2-17

How to Change the Ignore Security Data in Deployment Descriptors Check Box

To change the value of the Ignore Security Data in Deployment Descriptors check box, follow these steps:

1. Using the navigation tree at the left side of the WebLogic Server Administration Console, expand Security, then Realms.
2. Click the name of a security realm for which you are setting this option (for example, `myrealm`).
3. On the General tab, click the Ignore Security Data in Deployment Descriptors check box to check or uncheck the box.
4. Click Apply to save your changes.

Understanding How These Settings Interact

Table 2-1 shows how to achieve the behavior you want from the WebLogic Security Service using different combinations of the `fullyDelegateAuthorization` and Ignore Security Data in Deployment Descriptors settings.

Table 2-1 How These Settings Interact

If you want to perform security checks on...	and set security for URL (Web) and EJB resources...	Then set fullyDelegateAuthorization to...	and set Ignore Security Data in Deployment Descriptors to...
All URL (Web) and EJB resources	using *only* the Administration Console	true	checked
All URL (Web) and EJB resources	by *copying* or *reinitializing* security data from the deployment descriptors into the configured Authorization and Role Mapping providers' databases when the Web application or EJB module is deployed, then using one of the other techniques **Note:** Security data will be copied/reinitialized *each time* the Web application or EJB module is deployed.	true	unchecked
Only on URIs and EJB methods that are specified in the deployment descriptors (default configuration)	using *only* the deployment descriptors	false	unchecked

Using the Combined Technique to Secure Your URL and EJB Resources

As described in "Techniques for Securing URL and EJB Resources" on page 2-7, you can combine the use of the WebLogic Server Administration Console and J2EE/WebLogic deployment descriptor techniques, and would typically do so for two reasons:

- To copy security configurations from deployment descriptors into the configured Authorization and Role Mapping providers' databases, upon initial deployment of Web application and EJB modules. This process enables you to use the Administration Console for subsequent modifications to security roles and security policies.
- To reinitialize security configurations for URL and EJB resources to their original state, as specified in the deployment descriptors.

Notes: Use of the combined technique for other purposes is not recommended. Before continuing, be sure you have read "Prerequisites for Securing URL and EJB Resources" on page 2-9.

The following sections provide step-by-step instructions for using the combined technique to secure your URL and EJB resources:

- "Copying Security Configurations" on page 2-17
- "Reinitializing Security Configurations" on page 2-25

Note: You may also want to review Chapter 8, "Examples: Copying and Reinitializing Security Configurations for the basicauth Web Application," before performing these tasks.

Copying Security Configurations

These instructions are intended for administrators who presently secure URL (Web) and Enterprise JavaBean (EJB) resources using J2EE and WebLogic deployment descriptors, but want to exclusively use the WebLogic Server Administration Console from this point foward. Note that BEA does **not** recommend maintaining security configurations in both the deployment descriptors and the Administration Console.

Caution: When using the combined technique, it is possible to override security configurations for URL and EJB resources. Therefore, you must take extra care to ensure that the appropriate security configuration is in place. Follow these instructions carefully to prevent data loss and to ensure that your URL and EJB resources are secured properly.

To copy security configurations for a URL or EJB resource so that you can use the WebLogic Server Administration Console for subsequent modifications, follow these steps:

Step 1: Modify the Prerequisite Settings and Deploy the Resource

1. Set the `fullyDelegateAuthorization` flag equal to `true`, using the instructions in "How to Change the fullyDelegateAuthorization Flag" on page 2-11.

   Notes: Recall what this setting means: you are telling WebLogic Server that you want the WebLogic Security Service to perform security checks on *all* URL (Web) and EJB resources. For more information, see "Understanding the fullyDelegateAuthorization Flag" on page 2-10.

   If the `fullyDelegateAuthorization` flag was already set to `true`, just continue to step 2.

2. Start the server and sign in to the WebLogic Server Administration Console. (For help, see "Starting and Stopping WebLogic Servers" in the *WebLogic Server Administration Guide*.)

   The `fullyDelegateAuthorization` flag appears in the console as the server starts.

3. Using the navigation tree at the left side of the Administration Console, click expand Security, then Realms.

4. Click the name of your security realm (for example, `myrealm`).
5. On the General tab, *uncheck* the Ignore Security Data in Deployment Descriptors check box. (The check box may already be unchecked, as this is the default setting.)

   Note: Recall what this setting means: you are telling WebLogic Server to *copy* security for URL (Web) and EJB resources from the deployment descriptors into the configured Authorization and Role Mapping providers' databases *each time you deploy the resource.* For more information, see "Understanding the Ignore Security Data in Deployment Descriptors Check Box" on page 2-14.

6. Click Apply to save your changes.
7. If you had to set the `fullyDelegateAuthorization` flag to `true` in step 1 (that is, it was **not** already set this way), restart the server. (For help, see "Starting and Stopping WebLogic Servers" in the *WebLogic Server Administration Guide*.)

   If you did not have to modify the value of the `fullyDelegateAuthorization` flag in step 1, continue to step 8 **without restarting the server**.

8. Deploy the Web application or EJB module whose security configuration you want to copy, targeting it to the appropriate server.

   Note: For instructions about how to deploy Web applications, see "Deploying J2EE Applications with the Administration Console" in *Developing WebLogic Server Applications*.

Step 2: Verify the Copied Security Policies (Optional)

To verify the copied security policies, follow the instructions shown in the appropriate column of Table 2-2.

Table 2-2 Verifying Copied Security Policies, Based on Resource Type

Step	URL (Web) Resources	EJB Resources
1	Open the `web.xml` deployment descriptor for the Web application, and record the content of any `<url-pattern>` and `<http-method>` elements, as well as any `<role-name>` subelements of the `<auth-constraint>` element.	Open the `ejb-jar.xml` deployment descriptor for the EJB, and record the content of any `<method-permission>` elements, specifically focusing on the `<role-name>`, `<ejb-name>`, and `<method-name>` subelements. **Note:** If the deployment descriptor uses the `<unchecked />` element where you would normally find a `<role-name>` element, security checks will not be performed on that method; therefore, no security data for that method will be copied.
2	Using the navigation tree at the left side of the Administration Console, right-click the name of the deployed Web application module.	Using the navigation tree at the left side of the Administration Console, right-click the name of the deployed EJB module.
3	Select the Define Policy... option from the menu.	Select the Define Roles and Policies for Individual Beans... option from the menu. A table listing all the EJBs that are in the JAR file appears.
4	Enter a URL pattern in the URL Pattern text field that corresponds to the content of a single `<url-pattern>` element you recorded in step 1. Then, click the Define Policy... button to proceed.	Click the [Define Policies] link for the EJB that corresponds to the `<ejb-name>` element you recorded in step 1.

Table 2-2 Verifying Copied Security Policies, Based on Resource Type (Continued)

Step	URL (Web) Resources	EJB Resources
5	On the Policy Editor page that appears, select a method from the Methods drop-down menu that corresponds to the content of a `<http-method>` element you recorded in step 1. The `Caller is Granted the Role` condition in the Policy Condition list box is highlighted, and the content of the Policy Statement list box corresponds to the content of the appropriate `<role-name>` element that you recorded in step 1.	On the Policy Editor page that appears, select a method from the Methods drop-down menu that corresponds to the content of a `<method-name>` element you recorded in step 1. The `Caller is Granted the Role` condition in the Policy Condition list box is highlighted, and the content of the Policy Statement list box corresponds to the content of the corresponding `<role-name>` element that you recorded in step 1.
6	Repeat steps 1 - 5 to verify multiple security policies.	Repeat steps 1 - 5 to verify multiple security policies.

Step 3: Verify the Copied Security Roles (Optional)

To verify the copied security roles, follow the instructions shown in the appropriate column of Table 2-3.

Table 2-3 Verifying Copied Security Roles, Based on Resource Type

Step	URL (Web) Resources	EJB Resources
1	Open the `weblogic.xml` deployment descriptor for the Web application, and record the content of any `<security-role-assignment>` elements, specifically focusing on the `<role-name>` and `<principal-name>` subelements. **Note:** If the deployment descriptor uses the `<global-role/>` element for an Web application, no scoped roles are actually defined; therefore no scoped roles for the Web application can be copied.	Open the `weblogic-ejb-jar.xml` deployment descriptor for the EJB, and record the content of any `<security-role-assignment>` elements, specifically focusing on the `<role-name>` and `<principal-name>` subelements. **Note:** If the deployment descriptor uses the `<global-role/>` element for an EJB, no scoped roles are actually defined; therefore no scoped roles for the EJB can be copied.
2	Using the navigation tree at the left side of the Administration Console, right-click the name of the deployed Web application module.	Using the navigation tree at the left side of the Administration Console, right-click the name of the deployed EJB module.

Table 2-3 Verifying Copied Security Roles, Based on Resource Type

Step	URL (Web) Resources	EJB Resources
3	Select the Define Role... option from the menu. The General tab appears.	Select the Define Role... option from the menu. The Select Roles page displays all the scoped roles for this EJB that are currently defined in the WebLogic Role Mapping provider's database, including the ones from your deployment descriptor's `<role-name>` element.
4	In the URL Pattern text field, enter `/*`, then click the Define Role... button to proceed. **Note:** Security roles obtained from deployment descriptors are always copied into the configured Role Mapping provider's database as scoped roles, with an URL pattern of `/*`. The Select Roles displays all the scoped roles for this Web application that are currently defined in the WebLogic Role Mapping provider's database, including the one from your deployment descriptor's `<role-name>` element.	Click the hyperlinked name of the scoped role.

Table 2-3 Verifying Copied Security Roles, Based on Resource Type

Step	URL (Web) Resources	EJB Resources
5	Click the hyperlinked name of the scoped role.	Click the Conditions tab. The Role Statement list box contains a Role Statement based on the content of your deployment descriptor's corresponding `<principal-name>` element. **Note:** Because principals can be users or groups, the Role Statement list box will show two expressions: one using the contents of the `<principal-name>` element in the `User Name of the Caller` Role Condition, the other using it in a `Caller is a Member of the Group` Role Condition, linked by an `or` statement. The Administration Console presumes that a user or group of the name used in the deployment descriptor already exists. If they do not, you will need to create them.

Table 2-3 Verifying Copied Security Roles, Based on Resource Type

Step	URL (Web) Resources	EJB Resources
6	Click the Conditions tab. The Role Statement list box contains a Role Statement based on the content of your deployment descriptor's corresponding `<principal-name>` element. **Note:** Because principals can be users or groups, the Role Statement list box will show two expressions: one using the contents of the `<principal-name>` element in the `User Name of the Caller` Role Condition, the other using it in a `Caller is a Member of the Group` Role Condition, linked by an `or` statement. The Administration Console presumes that a user or group of the name used in the deployment descriptor already exists. If they do not, you will need to create them.	Repeat steps 1 - 5 to verify multiple scoped roles.
7	Repeat steps 1 - 6 to verify multiple scoped roles.	--

Step 4: Revert the Ignore Security Data in Deployment Descriptors Setting

Caution: You must perform this step. Failure to revert this setting may result in inconsistent security configurations when your Web application and EJB modules are redeployed. Therefore, be sure to perform this step **before** you restart your server.

1. Using the navigation tree at the left side of the Administration Console, expand Security, then Realms.
2. Click the name of your security realm (for example, `myrealm`).

3. On the General tab, click the Ignore Security Data in Deployment Descriptors check box. (That is, you should be putting a check mark in the box.)

   Note: Recall what this setting means: you are telling WebLogic Server that you will set security for Web application and EJB resources using the Administration Console, not deployment descriptors. For more information, see "Understanding the Ignore Security Data in Deployment Descriptors Check Box" on page 2-14.

4. Click Apply to save your changes.

Step 5: Modify Security Roles and Security Policies Using the Administration Console (Optional)

Follow the instructions in "Modifying Global Roles" on page 4-17 and "Modifying Security Policies" on page 5-20 to modify your URL (Web) resource's security roles and security policies.

Reinitializing Security Configurations

To reinitialize security configurations for URL (Web) and Enterprise JavaBean (EJB) resources to their original state as specified in their deployment descriptors, follow these steps:

- "Step 1: Modify the Prerequisite Settings and Redeploy the WebLogic Resource" on page 2-25
- "Step 2: Verify the Reinitialized Security Policies and Security Roles (Optional)" on page 2-27
- "Step 3: Revert the Ignore Security Data in Deployment Descriptors Setting" on page 2-27
- "Step 4: Modify Security Roles and Security Policies Using the Administration Console (Optional)" on page 2-27

Step 1: Modify the Prerequisite Settings and Redeploy the WebLogic Resource

1. Set the `fullyDelegateAuthorization` flag equal to `true`, using the instructions in "How to Change the fullyDelegateAuthorization Flag" on page 2-11.

   Notes: Recall what this setting means: you are telling WebLogic Server that you want the WebLogic Security Service to perform security checks on *all* URL (Web) and EJB

resources. For more information, see "Understanding the fullyDelegateAuthorization Flag" on page 2-10.

If the `fullyDelegateAuthorization` flag was already set to `true`, just continue to step 2.

2. Start the server and sign in to the WebLogic Server Administration Console. (For help, see "Starting and Stopping WebLogic Servers" in the *WebLogic Server Administration Guide*.)

   The `fullyDelegateAuthorization` flag appears in the console as the server starts.

3. Using the navigation tree at the left side of the Administration Console, expand Security, then Realms.

4. Click the name of your security realm (for example, `myrealm`).

5. On the General tab, *uncheck* the Ignore Security Data in Deployment Descriptors check box.

   Note: Recall what this setting means: you are telling WebLogic Server to *copy* security for URL (Web) and EJB resources from the deployment descriptors into the configured Authorization and Role Mapping providers' databases *each time you deploy the resource.* For more information, see "Understanding the Ignore Security Data in Deployment Descriptors Check Box" on page 2-14.

6. Click Apply to save your changes.

7. Using the navigation tree at the left side of the Administration Console, expand Deployments, then click either:

   - Web Applications—for URL (Web) resources, or
   - EJB—for Enterprise JavaBean (EJB) resources.

8. Click the name of the Web application or EJB.

   A table that lists all the Web applications or EJBs appears in the right pane.

9. Click the trash can icon that is located in the same row as the Web application or EJB for which you want to reinitialize a security configuration.

10. Click Yes, then the Continue link to delete the Web application or EJB.

    The deleted Web application or EJB no longer appears in the table.

11. Redeploy the Web application or EJB whose security configuration you want to initialize, targeting it to the appropriate server.

    Note: For instructions about how to deploy Web applications and EJBs, see "Deployment Tools and Procedures" in *Developing WebLogic Server Applications*.

Step 2: Verify the Reinitialized Security Policies and Security Roles (Optional)

To verify the reinitialized security policies and security roles, follow the instructions shown in the appropriate column of Table 2-2 and Table 2-3, respectively.

Step 3: Revert the Ignore Security Data in Deployment Descriptors Setting

Caution: You must perform this step. Failure to revert this setting may result in inconsistent security configurations when your Web applications and EJBs are redeployed. Therefore, be sure to perform this step **before** you restart your server.

1. Using the navigation tree at the left side of the Administration Console, expand Security, then Realms.
2. Click the name of your security realm (for example, `myrealm`).
3. On the General tab, click the Ignore Security Data in Deployment Descriptors check box. (That is, you should be putting a check mark in the box.)

   Note: Recall what this setting means: you are telling WebLogic Server that you will set security for Web application and EJB resources using the Administration Console, not deployment descriptors. For more information, see "Understanding the Ignore Security Data in Deployment Descriptors Check Box" on page 2-14.

4. Click Apply to save your changes.

Step 4: Modify Security Roles and Security Policies Using the Administration Console (Optional)

Follow the instructions in "Modifying Global Roles" on page 4-17 and "Modifying Security Policies" on page 5-20 to modify your URL (Web) or EJB resource's security roles and security policies.

Web Service Resources

A WebLogic Web Service is typically packaged as an enterprise application that contains a special type of Web application that includes an additional deployment descriptor called `web-services.xml`. If your Web Service is implemented with a Java class, then the Web application WAR file contains the Java class files. If the Web Service is implemented with a stateless session EJB, then the enterprise application EAR file contains the corresponding EJB JAR file.

Note: A Web Service can also be packaged as a stand-alone Web application WAR file if the Web Service is implemented with just a Java class. This type of packaging for a Web Service is uncommon, however; typically Web Services are packaged as EAR files.

A **Web Service resource** is a specific type of WebLogic resource that is related to Web Services. To secure Web Services, you can create security policies and security roles for:

- The entire Web Service
- A subset of the operations of the Web Service
- The Web Service URL
- The stateless session EJB that implements the Web Service
- A subset of the methods of the stateless session EJB
- The WSDL and Home Page of the Web Service

Note: For more information about securing WebLogic Web Services, see "Configuring Security" in *Programming WebLogic Web Services*.

CHAPTER

3 Users and Groups

A **user** is an entity that can be authenticated. A user can be a person or a software entity, such as a Java client. Each user is given a unique identity within a security realm. For more efficient security management, BEA recommends adding users to groups. A **group** is a collection of users who usually have something in common, such as working in the same department in a company.

The following sections provide more information about users:

The following sections provide more information about groups:

Creating Users

Notes: The instructions in this section apply to the WebLogic Authentication provider only. If you customize the default security configuration to use a custom Authentication provider, you must use the administration tools supplied by that security provider to create a user.

When upgrading to the WebLogic Authentication provider, there is no automatic way to load existing users into the WebLogic Authentication provider's database. For this release of WebLogic Server, adding existing users is a manual step. If you have a large number of existing users, consider using the Realm Adapter Authentication provider. For more information about the Realm Adapter Authentication provider, see "Configuring a Realm Adapter Authentication Provider" in *Managing WebLogic Security*.

To create a new user, follow these steps:

1. In the left pane of the WebLogic Server Administration Console, expand Security, then Realms.
2. Expand the security realm for which you are creating a user (for example, `myrealm`).
3. Click Users.

   If available, a table of currently defined users appears in the right pane.
4. Click the Configure a New User... link to display the Create User page.

   Note: If multiple WebLogic Authentication providers are configured in the security realm, you will need to select which WebLogic Authentication provider's database should store information for the new user.
5. On General tab, enter the name of the user in the Name field.

   Notes: Do not use blank spaces, commas, hyphens, or any characters in this comma-seperated list: \t, < >, #, |, &, ~, ?, ( ), { }, `*`, `/`. User names are case sensitive.
6. If desired, enter a description of the user (such as their full name) in the Description field.
7. Enter a password for the user in the Password field.

Notes: The minimum password length for a user defined in the WebLogic Authentication provider is 8 characters. Do not use the user name/password combination `weblogic/weblogic` in a production environment.

8. Re-enter the password for the user in the Confirm Password field.
9. Click Apply to save your changes.

Adding Users to Groups

BEA recommends adding users to groups because groups allow you to manage a number of users at the same time. This is generally more efficient than managing each user individually.

Note: The instructions in this section assume that you have already created groups as described in "Creating Groups" on page 3-7, *or* that you will use the default groups described in "Default Groups" on page 3-5.

To add a user to a group, follow these steps:

1. In the left pane of the WebLogic Server Administration Console, expand Security, then Realms.
2. Expand the security realm for which you are adding a user to a group (for example, `myrealm`).
3. Click Users.

   If available, a table of currently defined users appears in the right pane.
4. Click the hyperlinked name of the user that you want to add to a group.

   Note: If you have a large number of users, use the Filter By field to retrieve and list only the users that match your search criteria. The Filter By field uses the asterisk (`*`) as the wildcard character.
5. Click the Groups tab.

   All the available groups appear in the Possible Groups list box. All the groups to which the user belongs appear in the Current Groups list box.

6. In the Possible Groups list box, highlight the name of a group.
7. Click the highlighted arrow to move the group from the Possible Groups list box to the Current Groups list box.
8. If desired, repeat steps 6 and 7 to add the user to multiple groups.
9. Click Apply to save your changes.

Modifying Users

To modify an existing user, follow these steps:

1. In the left pane of the WebLogic Server Administration Console, expand Security, then Realms.
2. Expand the security realm for which you are modifying a user (for example, `myrealm`).
3. Click Users.

   A table of currently defined users appears in the right pane.
4. Click the hyperlinked name of the user that you want to modify.

   Note: If you have a large number of users, use the Filter By field to retrieve and list only the users that match your search criteria. The Filter By field uses the asterisk (`*`) as the wildcard character.
5. Use the General tab to modify the user's description or password, and the Groups tab to modify the user's membership in one or more groups. (See "Creating Users" on page 3-2 and "Adding Users to Groups" on page 3-3 for specific instructions.)

   Note: On both of these tabs, be sure to click Apply to save your changes.

Deleting Users

To delete an existing user, follow these steps:

1. In the left pane of the WebLogic Server Administration Console, expand Security, then Realms.
2. Expand the security realm from which you are deleting a user (for example, `myrealm`).
3. Click Users.

   A table of currently defined users appears in the right pane.
4. Click the trash can icon that is located in the same row as the user you want to delete.

   Note: If you have a large number of users, use the Filter By field to retrieve and list only the users that match your search criteria. The Filter By field uses the asterisk (*) as the wildcard character.
5. Click Yes to confirm the deletion.
6. Click Continue.

   The Select Users page no longer shows the deleted user in the table.

Default Groups

By default, WebLogic Server defines the groups shown in Table 3-1.

Table 3-1 Default Groups

Group Name	Membership
users	If a user identifies himself or herself when they log in (for example, through a Web page), the user is a member of this group. **Note:** The users group includes all users except the <anonymous> user. For more information about the <anonymous> user, see "Guest User" in the *WebLogic Server 7.0 Upgrade Guide*.
everyone	Regardless of whether a user identifies himself or herself when they log in, the user is a member of this group. **Note:** The everyone group includes (that is, is nested within) the users group.
Administrators	By default, this group contains the user information entered as part of the installation process, and the system user if the WebLogic Server instance is running Compatibility security. Any user assigned to the Administrators group is granted the Admin security role by default.
Deployers	By default, this group is empty. Any user assigned to the Deployers group is granted the Deployer security role by default.
Operators	By default, this group is empty. Any user assigned to the Operators group is granted the Operator security role by default.
Monitors	By default, this group is empty. Any user assigned to the Monitors group is granted the Monitor security role by default.

Note: For more information about the default security roles, see "Default Global Roles" on page 4-5.

You can add to the default groups by creating your own, as described in "Creating Groups" on page 3-7.

Creating Groups

Notes: The instructions in this section apply to the WebLogic Authentication provider only. If you customize the default security configuration to use a custom Authentication provider, you must use the administration tools supplied by that security provider to create a group.

When upgrading to the WebLogic Authentication provider, there is no automatic way to load existing groups into the WebLogic Authentication provider's database. For this release of WebLogic Server, adding existing groups is a manual step. If you have a large number of existing groups, consider using the Realm Adapter Authentication provider. For more information about the Realm Adapter Authentication provider, see "Configuring a Realm Adapter Authentication Provider" in *Managing WebLogic Security*.

To create a new group, follow these steps:

1. In the left pane of the WebLogic Server Administration Console, expand Security, then Realms.
2. Expand the security realm for which you are creating a group (for example, `myrealm`).
3. Click Groups.

   If available, a table of currently defined groups appears in the right pane.
4. Click the Configure a New Group... link to display the Create Group page.

   Note: If multiple WebLogic Authentication providers are configured in the security realm, you will need to select which WebLogic Authentication provider's database should store information for the new group.
5. On General tab, enter the name of the group in the Name field.

   Notes: Do not use blank spaces, commas, hyphens, or any characters in this comma-seperated list: \t, < >, #, |, &, ~, ?, ( ), { }, `*, /`. Group names are case sensitive. The BEA convention is that group names are plural.
6. If desired, enter a description of the group in the Description field.
7. Click Apply to save your changes.

Nesting Groups

If desired, you can nest groups within other groups.

Note: The instructions in this section assume that you have already created groups as described in "Creating Groups" on page 3-7 *or* that you will use the default groups described in "Default Groups" on page 3-5.

To nest a group within another group, follow these steps:

1. In the left pane of the WebLogic Server Administration Console, expand Security, then Realms.
2. Expand the security realm for which you are nesting a group (for example, `myrealm`).
3. Click Groups.

   A table of currently defined groups appears in the right pane.
4. Click the hyperlinked name of the group that you want to nest within another group.

   Note: If you have a large number of groups, use the Filter By field to retrieve and list only the groups that match your search criteria. The Filter By field uses the asterisk (`*`) as the wildcard character.
5. Click the Membership tab.

   All the available groups appear in the Possible Groups list box. All the groups in which the group is nested appear in the Current Groups list box.
6. In the Possible Groups list box, highlight the name of a group.
7. Click the highlighted arrow to move the group from the Possible Groups list box to the Current Groups list box.
8. If desired, repeat steps 6 and 7 to nest the group within multiple groups.
9. Click Apply to save your changes.

Modifying Groups

To modify an existing group, follow these steps:

1. In the left pane of the WebLogic Server Administration Console, expand Security, then Realms.
2. Expand the security realm for which you are modifying a group (for example, `myrealm`).
3. Click Groups.

   A table of currently defined groups appears in the right pane.
4. Click the hyperlinked name of the group that you want to modify.

   Note: If you have a large number of groups, use the Filter By field to retrieve and list only the groups that match your search criteria. The Filter By field uses the asterisk (`*`) as the wildcard character.
5. Use the General tab to modify the group's description, and the Membership tab to modify the group's membership in one or more other groups. (See "Creating Groups" on page 3-7 and "Nesting Groups" on page 3-8 for specific instructions.)

   Note: On both of these tabs, be sure to click Apply to save your changes.

Deleting Groups

To delete an existing group, follow these steps:

1. In the left pane of the WebLogic Server Administration Console, expand Security, then Realms.
2. Expand the name of the security realm from which you are deleting a group (for example, `myrealm`).
3. Click Groups.

   A table of currently defined groups appears in the right pane.

4. Click the trash can icon that is located in the same row as the group you want to delete.

   Note: If you have a large number of groups, use the Filter By field to retrieve and list only the users that match your search criteria. The Filter By field uses the asterisk (*) as the wildcard character.

5. Click Yes to confirm the deletion.

6. Click Continue link.

   The Select Groups page no longer shows the deleted group in the table.

CHAPTER

4 Security Roles

A **security role** is a privilege granted to users or groups based on specific conditions. Like groups, security roles allow you to restrict access to WebLogic resources for several users at once. However, security roles:

- Are computed and granted to users or groups dynamically, based on conditions such as user name, group membership, or the time of day.
- Can be scoped to specific WebLogic resources within a single application in a WebLogic Server domain (unlike groups, which are always scoped to an entire WebLogic Server domain).

Granting a security role to a user or a group confers the defined access privileges to that user or group, as long as the user or group is "in" the security role. For example, an administrator may define a security role called `AppAdmin`, which has write access to a Web application's resources. Any user or group granted the `AppAdmin` security role would then have write access to that URL (Web) resource. Multiple users or groups can be granted a single security role. (For more information about users and groups, see Chapter 3, "Users and Groups.")

Note: In WebLogic Server 6.x, security roles applied to Web applications and Enterprise JavaBeans (EJBs) only. This version of WebLogic Server expands the use of security roles to all of the defined WebLogic resources. For more information, see Chapter 2, "Types of WebLogic Resources."

The following sections provide more information about security roles:

Dynamic Role Mapping

At runtime, the WebLogic Security Service compares users or groups against a role condition to determine whether they should be dynamically granted a security role. This process is referred to as **role mapping**, and occurs just prior to when the WebLogic Security Service renders an access decision for a protected WebLogic resource.

Note: For more information about role conditions and access decisions, see "Components of a Security Role: Role Conditions, Expressions, and Role Statements" on page 4-12 and "Access Decisions" in *Developing Security Providers for WebLogic Server*, respectively.

This dynamic mapping of security roles to users or groups provides a very important benefit: users or groups can be granted a security role based on business rules, or the context of the request. For example, a user may be allowed to be in a `Manager` security role only while the actual manager is away. Dynamically granting this security role means that you do not need to change or redeploy your application to allow for such a temporarily arrangement. You could simply specify the hours between which the temporary manager should have special privileges. Further, you would not need to remember to revoke these special privileges when the actual manager returns, as you would if you temporarily added the user to a management group.

Types of Security Roles: Global Roles and Scoped Roles

There are two types of security roles in WebLogic Server: global roles and scoped roles. A security role that applies to all WebLogic resources deployed within a security realm (and thus the entire WebLogic Server domain) is called a **global role**. A security role that applies to a specific instance of a WebLogic resource deployed in a security realm (such as a method on an EJB or a branch of a JNDI tree) is called a **scoped role**. Multiple roles (either global or scoped) can be used to create a security policy for a WebLogic resource. (For more information, see Chapter 5, "Security Policies.")

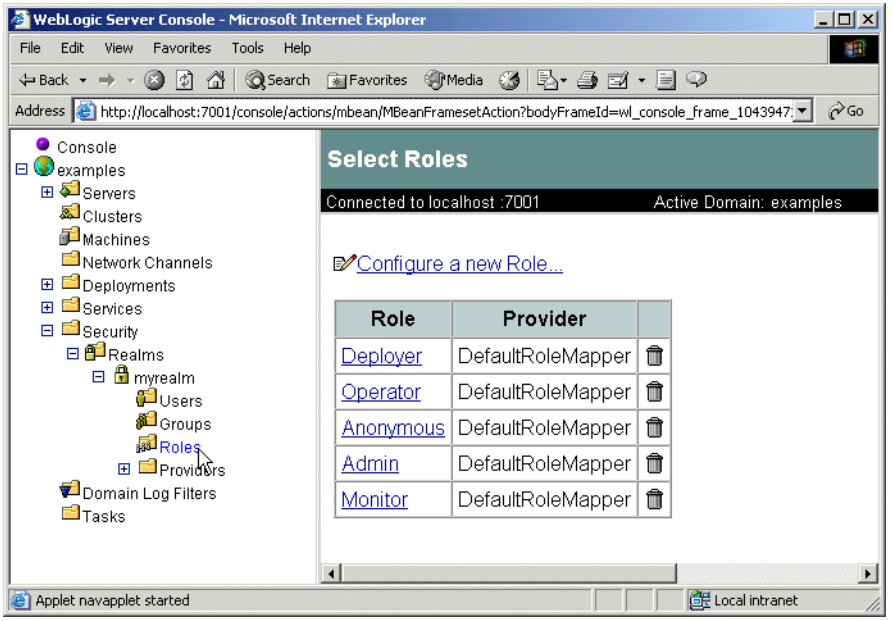
While BEA recommends creating security roles and using them (rather than users or groups) to secure WebLogic resources, you do not need to use a particular type of security role. BEA provides several default global roles that you can use out of the box to secure your WebLogic resources; these are described in "Default Global Roles" on page 4-5. If you do not have a need for them, you never need to use scoped roles. (Scoped roles are provided for their flexibility and are an extra feature for advanced customers.)

Ways to Create Security Roles in the Administration Console

The way you use the WebLogic Server Administration Console to create security roles differs, depending on whether you want to create a global role or a scoped role.

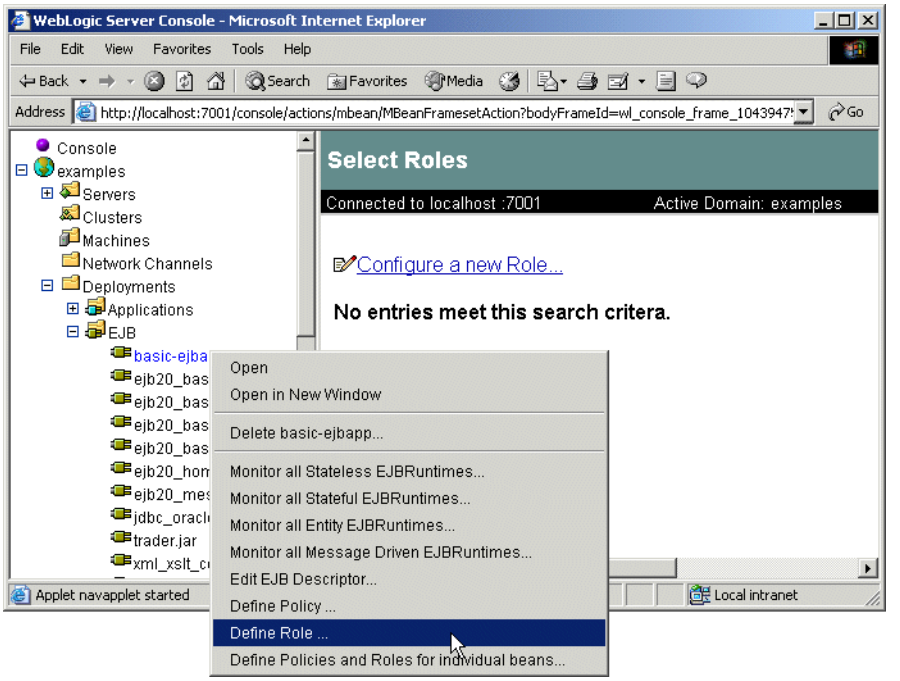
Since global roles apply to all WebLogic resources in a security realm, you create global roles at the security realm level. Using the Administration Console, expand Security, then Realms, then `myrealm` (or the name of a security realm you created). Then click Roles to display the page that allows you to create a global role. This navigation path is shown on the left side of Figure 4-1, and the resulting page is shown on the right.

Figure 4-1 Creating a Global Role



Because they apply only to a particular WebLogic resource in a security realm, you create scoped roles at the WebLogic resource level. Deployed components (that is, Web applications, EJBs, and so on) for which you can create scoped roles show a Define Role... option when you right-click on them in the Administration Console's navigation tree. Select the Define Role... option to display the page that allows you to create a scoped role. This navigation path—using the `basic-ejbapp` JAR as the WebLogic resource—is shown on the left side of Figure 4-2, and the resulting page is shown on the right.

Figure 4-2 Creating a Scoped Role



Default Global Roles

By default, WebLogic Server defines the global roles shown in Table 4-1. The table also lists the privileges that users or groups in these security roles are granted.

Note: The default global roles are used in the default security policies that protect most types of WebLogic resources. In addition, the default global roles are used to provide additional security for Server resources that are exposed as MBeans. For more information, see Chapter 5, "Security Policies," and "Protecting System Administration Operations" in the *WebLogic Server Administration Guide*.

Table 4-1 Default Global Roles and Their Privileges

Global Role	Privileges
`Anonymous`	All users (the group `everyone`) are granted this global role. **Note:** This global role is provided as a convenience, and can be specified in the `weblogic.xml` and `weblogic-ejb-jar.xml` deployment descriptors.
`Admin`	■ View the server configuration, *including* the encrypted value of encrypted attributes. ■ Modify the entire server configuration. ■ Deploy enterprise applications, startup and shutdown classes, and Web application, EJB, J2EE Connector, Web Service, and WebLogic Tuxedo Connector components. If applicable, edit deployment descriptors. ■ Start, resume, and stop servers by default.
`Deployer`	■ View the server configuration, *except* for encrypted attributes. ■ Change startup and shutdown classes, Web applications, JDBC data pool connections, EJB, J2EE Connector, Web Service, and WebLogic Tuxedo Connector components. If applicable, edit deployment descriptors.
`Operator`	■ View the server configuration, *except* for encrypted attributes. ■ Start, resume, and stop servers by default.
`Monitor`	View the server configuration, *except* for encrypted attributes. **Note:** This security role effectively provides read-only access to the Administration Console, `weblogic.Admin` utility and MBean APIs.

Note: If you are working directly with WebLogic Server MBeans and want more detailed information about the global roles and their privileges than is shown in Table 4-1, see "Protected MBean Attributes and Operations" on page 4-7.

You can add to the default global roles by creating your own security roles (global or scoped) as described in "Creating Global Roles" on page 4-15 and "Creating Scoped Roles" on page 4-19, respectively.

Protected MBean Attributes and Operations

Table 4-2 lists the immutable privileges given to users or groups who are granted the `Admin` default global role, for various WebLogic Server MBeans. In other words, users or groups who are granted the `Admin` default global role have permission to access the MBean attributes listed in Table 4-2.

Note: Users or groups who are granted the `Admin` default global role are also given the privileges described in Table 4-3 through Table 4-5.

Table 4-2 MBean Privileges for the Admin Default Global Role

MBeans	Attributes
BridgeDestinationCommonMBean	UserPassword
BridgeDestinationMBean	UserPassword
JDBCConnectionPoolMBean	Password, XAPassword
JDBCDataSourceFactoryMBean	Password
JMSBridgeDestinationMBean	UserPassword
NetworkChannelMBean	DefaultIIOPPassword
NodeManagerMBean	CertificatePassword
SecurityConfigurationMBean	Credential, EncryptedSecretKey, Salt
SecurityMBean	Salt, EncryptedSecretKey
ServerMBean	SystemPassword,DefaultIIOPPassword, DefaultTGIOPPassword, CustomIdentityKeyStorePassPhrase, CustomTrustKeyStorePassPhrase, JavaStandardTrustKeyStorePassPhrase
ServerStartMBean	Password
SSLMBean	ServerPrivateKeyPassPhrase
WLECConnectionPoolMBean	UserPassword, ApplicationPassword

Notes: The MBeans shown in Table 4-2 are all in the `weblogic.management.configuration` package. For more information on MBeans used to configure WebLogic Server, see "System Administration Infrastructure" in the *WebLogic Server Administration Guide*.

Table 4-3 lists the immutable privileges given to users or groups who are granted the `Admin` or `Deployer` default global roles, for various WebLogic Server MBeans. In other words, users or groups who are granted the `Admin` or `Deployer` default global roles have permission to access the MBean operations listed in Table 4-3.

Table 4-3 Privileges for the Admin or Deployer Default Global Roles

MBeans	Operations
Application, ApplicationConfig	All
ConnectorComponent, ConnectorComponentConfig	All
DeployerRuntime, DeploymentTaskRuntime	All
EJBComponent, EJBComponentConfig	All
WebAppComponent, WebAppComponentConfig	All
WebServiceComponent, WebServiceComponentConfig	All
WebServer, WebServerConfig	All
JDBCConnectionPool, JDBCConnectionPoolConfig	All
JDBCDataSourceFactory, JDBCDataSourceFactoryConfig	All
JDBCMultiPool, JDBCMultipoolConfig	All
JDBCDataSource, JDBCDataSourceConfig	All
JDBCTxDataSource, JDBCTxDataSourceConfig	All

Table 4-3 Privileges for the Admin or Deployer Default Global Roles (Continued)

MBeans	Operations
JDBCPoolComponent, JDBCPoolComponentConfig	All
JMSBridgeDestination, JMSBridgeDestinationConfig	All
JMSConnectionConsumer, JMSConnectionConsumerConfig	All
JMSConnectionFactory, JMSConnectionFactoryConfig	All
JMSDestination, JMSDestinationConfig	All
JMSDistributedDestination, JMSDistributedDestinationConfig	All
JMSDistributedDestinationMember, JMSDistributedDestinationMemberConfig	All
JMSDistributedTopic, JMSDistributedTopicConfig	All
JMSDistributedTopicMember, JMSDistributedTopicMemberConfig	All
JMSDistributedQueue, JMSDistributedQueueConfig	All
JMSDistributedQueueMember, JMSDistributedQueueMemberConfig	All
JMSFileStore, JMSFileStoreConfig	All
JMSDestinationKey, JMSDestinationKeyConfig	All
JMSServer, JMSServerConfig	All
JMSStore, JMSStoreConfig	All
JMSSessionPool, JMSSessionPoolConfig	All
JMSTemplate, JMSTemplateConfig	All

Table 4-3 Privileges for the Admin or Deployer Default Global Roles (Continued)

MBeans	Operations
JMSQueue, JMSQueueConfig	All
JMSTopic, JMSTopicConfig	All
JMSJDBCStore, JMSJDBCStoreConfig	All
WTCServer, WTCServerConfig	All
WTCBridgeGlobal, WTCBridgeGlobalConfig	All
WTCResources, WTCResourcesConfig	All
WTCExport, WTCExportConfig	All
WTCImport, WTCImportConfig	All
WTCLocalTuxDom, WTCLocalTuxDomConfig	All
WTCRemoteTuxDom, WTCRemoteTuxDomConfig	All
WTCPassword, WTCPasswordConfig	All
WTCtBridgeGlobal, WTCtBridgeGlobalConfig	All
WTCtBridgeRedirect, WTCtBridgeRedirectConfig	All
EJBDescriptor, ConnectorDescriptor, WebDescriptor	All
Server	addDeployment, lookupServerLifeCycleRuntime, lookupServerRuntime, removeDeployment, sendNotification
ServerConfig	addDeployment, lookupServerLifeCycleRuntime, removeDeployment, sendNotification

Table 4-4 lists the immutable privileges given to users or groups who are granted the `Admin` or `Monitor` default global roles, for various WebLogic Server MBeans. In other words, users or groups who are granted the `Admin` or `Monitor` default global roles have permission to access the MBean operations listed in Table 4-4.

Table 4-4 Privileges for the Admin or Monitor Default Global Roles

MBeans	Operations
Machine	lookupNodeManagerRuntime
NodeManagerRuntime	getStateForAll, register
Server	lookupServerLifeCycleRuntime, lookupServerRuntime

Table 4-5 lists the immutable privileges given to users or groups who are granted the `Admin` or `Operator` default global roles, for various WebLogic Server MBeans. In other words, users or groups who are granted the `Admin` or `Operator` default global roles have permission to access the MBean operations listed in Table 4-5.

Table 4-5 Privileges for the Admin or Operator Default Global Roles

MBeans	Operations
ServerLifeCycleRuntime	All
ServerLifeCycleTaskRuntime	All
ServerStart	All
Server	ExpectedToRun, lookupServerLifeCycleRuntime, lookupServerRuntime, sendNotification, start, suspend
ServerConfig	ExpectedToRun, lookupServerLifeCycleRuntime, sendNotification
ServerRuntime	forceShutdown, resume, shutdown, start, stop

Default Group Associations

By default, WebLogic Server grants four of the default global roles to four of the default groups. By adding a user to one of these groups, the user is automatically granted the global role. These default group associations are shown in Table 4-6.

Table 4-6 Default Group Associations

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

Components of a Security Role: Role Conditions, Expressions, and Role Statements

A **role condition** is a condition under which a security role (global or scoped) will be granted to a user or group. The role conditions that are available in this release of WebLogic Server are:

- `User Name of the Caller`—Creates a condition for a security role based on a user name. For example, you might create a condition indicating that only the user `John` can be granted the `BankTeller` security role.
- `Caller is a Member of the Group`—Creates a condition for a security role based on a group. For example, you might create a condition indicating that only users in the group `FullTimeBankEmployees` can be granted the `BankTeller`

security role. BEA recommends this role condition for more efficient security management.

- `Hours of Access are Between`—Creates a condition for a security role based on a specified time period. For example, you might create a condition indicating that the `BankTeller` security role can only be granted to users when the bank is open.

  When you use the `Hours of Access are Between` role condition, the security role will be granted to all users during the hours you specify, unless you further restrict the users by adding one of the other role conditions.

These role conditions, along with the specific information you supply for the condition (such as an actual user name, group, or start/stop times), are called **expressions**. An example of an expression that you may see in the WebLogic Server Administration Console is shown in Figure 4-3.

Figure 4-3 Expression Example

In this expression example, the first line is the role condition, the second line is the specific information you supply for the condition—in this case, a group called `FullTimeBankEmployees`.

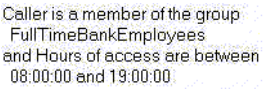
A **role statement** is a collection of expressions that define how a security role is granted, and is therefore the main part of any security role you create. The ability to use multiple expressions means that you can create complex security roles that meet your organization's security requirements. The use of `and` and `or` between these expressions, as well as the ordering of the expressions, is also an important feature:

- `And` is used to specify that all the expressions must be true in order for the security role to be granted.
- `Or` is used to specify that at least one of the expressions must be true in order for the security role to be granted.

Notes: The entire role statement must be true in order for a user or group to be granted the security role. More restrictive expressions should come later in a role statement. WebLogic Server evaluates expressions in a role statement from left to right.

An example of a role statement that you may see in the Administration Console is shown in Figure 4-4.

Figure 4-4 Role Statement Example



In this role statement example, there are two expressions: the first and second lines contain an expression based on the `Caller is a Member of the Group` role condition, and the third and fourth lines contain another expression based on the `Hours of Access are Between` role condition.

Working with Global Roles

The following sections provide instructions for working with global roles:

- "Creating Global Roles" on page 4-15
- "Modifying Global Roles" on page 4-17
- "Deleting Global Roles" on page 4-18

Note: This section describes how to create, modify, and delete global roles. Because they are always scoped to a WebLogic resource, instructions for creating, modifying, and deleting scoped roles are provided under "Working with Scoped Roles" on page 4-18.

Creating Global Roles

Notes: The section "Ways to Create Security Roles in the Administration Console" on page 4-3 may be helpful to review before creating security roles. If you are creating global roles that will be used to secure Server resources, be sure to adhere to the advice given in the "Maintaining a Consistent Security Scheme" section in *Protecting System Administration Operations*.

To create a new global role, follow these steps:

1. In the left pane of the WebLogic Server Administration Console, expand Security, then Realms.
2. Expand the security realm for which you are creating a global role (for example, `myrealm`).
3. Click Roles to display the Select Roles page.

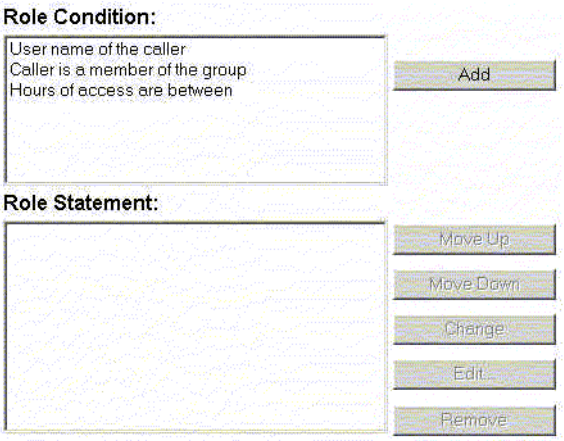
   If available, a table of currently defined global roles appears in the right pane.
4. Click Configure a New Role....

   Note: If multiple WebLogic Role Mapping providers are configured in the security realm, you will need to select which WebLogic Role Mapping provider's database should store information for the new global role.
5. On General tab, enter the name of the global role in the Name field.

   Notes: Do not use blank spaces, commas, hyphens, or any characters in this comma-seperated list: \t, < >, #, |, &, ~, ?, ( ), { }. Security role names are case sensitive. The BEA convention is that all security role names are singular and the first letter is capitalized..

   The proper syntax for a security role name is as defined for an `Nmtoken` in the *Extensible Markup Language (XML) recommendation*.
6. Click Apply to save your changes.
7. Click the Conditions tab to display the Role Editor page (see Figure 4-5).

Figure 4-5 Role Editor Page



8. In the Role Condition list box, click one of the conditions. (For more information about the different role conditions, see "Components of a Security Role: Role Conditions, Expressions, and Role Statements" on page 4-12.)

   Note: BEA recommends that you create expressions using the `Caller is a Member of the Group` condition. When a group is used to create a security role, the security role can be granted to all members of the group (that is, multiple users).

9. Click Add to display a customized window.

10. If you selected the `Hours of Access are Between` condition, use the Time Constraint window to select start and end times, and then click the OK button. The window closes and an expression appears in the Role Statement list box.

    If you selected one of the other conditions, follow these steps:

    a. Use the Users or Groups window to enter the name of a user or group, then click Add. An expression appears in the list box.

       Note: You can repeat this step multiple times to add more than one user or group.

    b. If necessary, use the buttons located to the right of the list box to modify the expressions.

Move Up and Move Down change the ordering of the highlighted user or group name. Change switches the highlighted `and` and `or` statements between expressions. Remove deletes the highlighted user or group name.

c. Click OK to add the expression to the role statement. The window closes and the expression appears in the Role Statement list box.

11. If desired, repeat steps 8 - 10 to add expressions based on different role conditions.
12. If necessary, use the buttons located to the right of the Role Statement list box to modify the expressions:
    - Move Up and Move Down change the ordering of the highlighted expression.
    - Change switches the highlighted `and` and `or` statements between expressions.
    - Edit... reopens the customized window for the highlighted expression and allows you to modify the expression.
    - Remove deletes the highlighted expression.
13. When all the expressions in the Role Statement list box are correct, click Apply.

    Note: You can also click Reset to restore the Role Editor page to the state it was in when the page was first loaded (that is, to undo any of your changes).

Modifying Global Roles

The procedure for modifying global roles is, for the most part, the same as the procedure for creating a new global role. Follow these steps:

1. In the left pane of the WebLogic Server Administration Console, expand Security, then Realms.
2. Expand the security realm from which you are modifying a global role (for example, `myrealm`).
3. Click Roles.

   A table of currently defined global roles appears in the right pane.
4. Select the global role that you want to modify from the table.

5. Select the Conditions tab to display the Role Editor page.
6. Make your changes, using steps 8 - 12 in "Creating Global Roles" on page 4-15 as a guide.
7. Click Apply to save your changes.

Deleting Global Roles

To delete a global role, follow these steps:

1. In the left pane of the WebLogic Server Administration Console, expand Security, then Realms.
2. Expand the security realm from which you are deleting a global role (for example, `myrealm`).
3. Click Roles.

   A table of currently defined global roles appears in the right pane.
4. Click the trash can icon that is located in the same row as the global role you want to delete.
5. Click Yes to confirm the deletion.
6. Click Continue.

   The Select Roles page no longer shows the deleted global role in the table.

Working with Scoped Roles

The following sections provide instructions for working with scoped roles for the various types of WebLogic resources:

- "Creating Scoped Roles" on page 4-19
- "Modifying Scoped Roles" on page 4-29
- "Deleting Scoped Roles" on page 4-30

Creating Scoped Roles

To create a scoped role for a WebLogic resource, follow these steps:

- "Step 1: Select the WebLogic Resource" on page 4-19
- "Step 2: Create the Scoped Role" on page 4-27
- "Step 3: Create the Role Conditions" on page 4-27

Note: The instructions for working with scoped roles vary slightly from WebLogic resource to WebLogic resource. Be sure to follow any variations noted in this procedure that pertain to the type of WebLogic resource with which you are working. For more information, see Chapter 2, "Types of WebLogic Resources."

Step 1: Select the WebLogic Resource

Follow the instructions in the appropriate section to select the type of WebLogic resource:

- "Administrative Resources" on page 4-20
- "Application Resources" on page 4-20
- "COM Resources" on page 4-20
- "EIS Resources" on page 4-21
- "EJB Resources" on page 4-22
- "JDBC Resources" on page 4-23
- "JMS Resources" on page 4-24
- "JNDI Resources" on page 4-24
- "Server Resources" on page 4-25
- "URL Resources" on page 4-26

Administrative Resources

In the left pane of the WebLogic Server Administration Console, right-click the name of the WebLogic Server domain (for example, `examples`), and choose Define Role... to display the Select Roles page.

If available, a table of currently defined scoped roles appears in the right pane.

Application Resources

1. In the left pane of the WebLogic Server Administration Console, expand Deployments, then Applications.

   Note: You can optionally expand the enterprise application (EAR) for which you are creating a scoped role to see the different types of WebLogic resources it contains.

2. Right-click the name of an enterprise application (EAR) and choose Define Role... to display the Select Roles page.

   If available, a table of currently defined scoped roles appears in the right pane.

COM Resources

If a package of EJB classes (such as `ejb20.basic.beanManaged.*`) will be accessed by a COM client, follow these steps to create the scoped role:

1. In the left pane of the WebLogic Server Administration Console, expand Deployments, then EJB.

   The EJB node expands to show the EJB JARs that are currently deployed.

2. Right-click the name of an EJB JAR containing the EJB that will be used to access the package, and choose Define Policies and Roles for Individual Beans... to display a list of EJBs.

3. Click the [Define JCOM Roles] link that is located in the same row as the EJB that will be used to access the package.

   The General tab's COM Class field already shows the name of the package to which you want to scope the security role.

   Note: The value in the COM class field is a Java class or package name that is exposed to COM via the jCOM bridge.

4. Click the Define Role... button to display the Select Roles page.

   If available, a table of currently defined scoped roles appears in the right pane.

If a package of Java classes (such as `java.util.*`) or individual classes (such as `java.util.Collection`) will be accessed by a COM client, follow these steps to create the scoped role:

1. In the left pane of the WebLogic Server Administration Console, expand Services.
2. Right-click the JCOM node and choose Define Role....
3. On the General tab, in the COM class field, enter the name of the Java class or package to which you want to scope the security role.

   Note: The value you enter in the COM class field is a Java class or package name that is exposed to COM via the jCOM bridge.

4. Click the Define Role... button to display the Select Roles page.

   If available, a table of currently defined scoped roles appears in the right pane.

EIS Resources

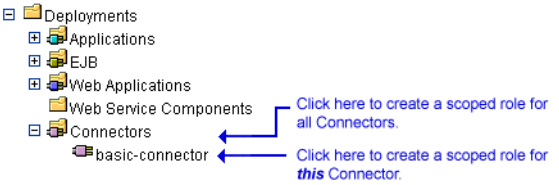
1. In the left pane of the WebLogic Server Administration Console, expand Deployments.

   The Deployments node expands to show the types of WebLogic resources that can be deployed.

2. Right-click at the level of the EIS resource for which you want to create the scoped role, and choose Define Role... to display the Select Roles page.

   To create a scoped role for *all* Connectors, right-click Connectors. To create a scoped role for a *particular* Connector, expand Connectors, then right-click the name of a Connector. Figure 4-6 illustrates where you might click, using the `basic-connector` as an example.

Figure 4-6 Deployments Portion of the Administration Console Navigation Tree



If available, a table of currently defined scoped roles appears in the right pane.

EJB Resources

Note: These instructions also apply to Message-driven Beans (MDBs).

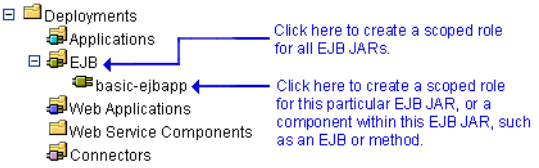
1. In the left pane of the WebLogic Server Administration Console, expand Deployments.

   The Deployments node expands to show the types of WebLogic resources that can be deployed.

2. Right-click at the level of the EJB resource for which you want to create the scoped role.

   To create a scoped role for *all* EJB JARs, right-click EJB. To create a scoped role for a *particular* EJB JAR, or for an EJB within a JAR, expand EJB, then right-click the name of an EJB JAR. Figure 4-7 illustrates where you might click, using the `basic-ejbapp` JAR as an example.

Figure 4-7 Deployments Portion of the Administration Console Navigation Tree



3. If you are creating the scoped role for all EJB JARs or for a particular EJB JAR (that is, for *all* the EJBs in the JAR), choose Define Role... to display the Select Roles page.

   If you are creating the scoped for a *particular* EJB in an EJB JAR, follow these steps:

a. Choose Define Policies and Roles for Individual Beans... to display a list of EJBs.

b. Click the [Define Roles] link that is located in the same row as the EJB for which you want to create the scoped role.

If available, a table of currently defined scoped roles appears in the right pane.

JDBC Resources

1. In the left pane of the WebLogic Server Administration Console, expand Services, then JDBC.

   The JDBC node expands to show nodes for various JDBC components (connection pools, MultiPools, and data sources).

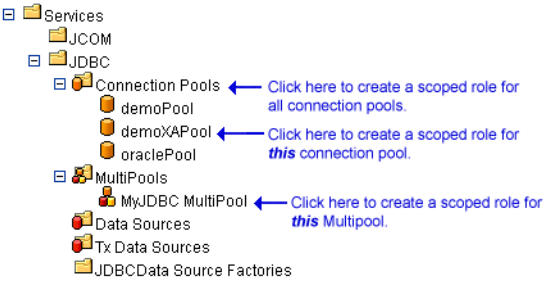
2. Right-click at the level of the JDBC resource for which you want to create the scoped role, and choose Define Role... to display the Scoped Roles page.

   To create a scoped role for *all* connection pools, right-click Connection Pools. To create a scoped role for a *particular* connection pool, expand Connection Pools, then right-click the name of a connection pool. To create a scoped role for an individual MultiPool, expand MultiPools, then right-click the name of the MultiPool.

   Note: You cannot create a scoped role that encompasses all MultiPools.

   Figure 4-8 illustrates where you might click, using various connection pools and a MultiPool as an example.

Figure 4-8 Services Portion of the Administration Console Navigation Tree



If available, a table of currently defined scoped roles appears in the right pane.

JMS Resources

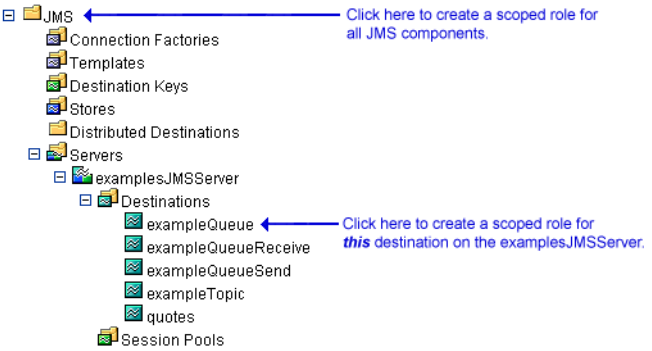
1. In the left pane of the WebLogic Server Administration Console, expand Services, then JMS.

   The JMS node expands to show nodes for various JMS components (connection factories, templates, destination keys, and so on).

2. Right-click at the level of the JMS resource for which you want to create the scoped role, and choose Define Role... to display the Select Roles page.

   To create a scoped role for *all* JMS components, right-click JMS. To create a scoped role for a *particular* destination on a JMS server, expand Servers, then the JMS server and the Destinations node, then right-click the name of a destination. Figure 4-9 illustrates where you might click, using various destinations on the `examplesJMSServer` as an example.

Figure 4-9 Services Portion of the Administration Console Navigation Tree



If available, a table of currently defined scoped roles appears in the right pane.

JNDI Resources

1. In the left pane of the WebLogic Server Administration Console, expand Servers.

   The Servers node expands to show the servers available in the current WebLogic Server domain.

2. Right-click the name of a server that contains the JNDI resource for which you want to create the scoped role. (For example, `myserver`.)

3. From the menu that appears, select the View JNDI Tree option.

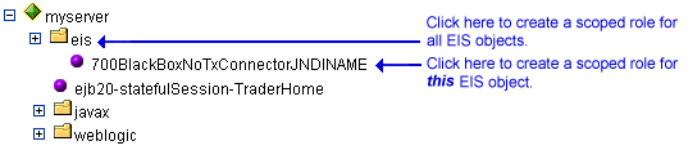
The JNDI tree for the server appears in a new Administration Console window.

4. In the new Administration Console window, right-click at the level of the JNDI tree at which you want to create the scoped role, and choose Define Role... to display the Select Roles page.

   To create a scoped role for *a group of* objects, right-click the node that represents that object type. To create a scoped role for a *particular* object, expand the node that represents that object, then right-click the name of an object.

   Figure 4-10 illustrates where you might click, using the examplesServer JNDI tree as an example.

Figure 4-10 New Administration Console Window for examplesServer JNDI Tree



If available, a table of currently defined scoped roles appears in the right pane.

Server Resources

In the left pane of the WebLogic Server Administration Console, expand Servers.

1. The Servers node expands to show the different Server resources for which a scoped role can be created.

2. Right-click at the level of the Server resource at which you want to create the scoped role, and choose Defined Role... to display the Select Roles page.

   To create a scoped role for *all* servers, right-click Servers. To create a scoped role for a *particular* server, expand Servers, then right-click the name of a server. Figure 4-11 illustrates where you might click, using the `examplesServer` as an example.

Figure 4-11 Servers Portion of the Administration Console Navigation Tree

If available, a table of currently defined scoped roles appears in the right pane.

URL Resources

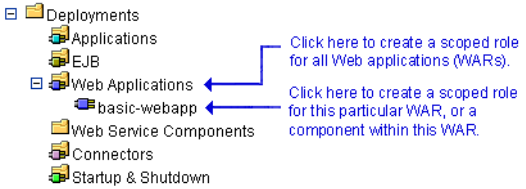
1. In the left pane of the WebLogic Server Administration Console, expand Deployments.

   The Deployments node expands to show the types of WebLogic resources that can be deployed.

2. Right-click at the level of the URL (Web) resource at which you want to create the scoped role.

   To create a scoped role for *all* Web applications (WARs), right-click Web Applications. To create a scoped role for a *particular* WAR or a component in a WAR (for example, a specific servlet or JSP), expand Web Applications, then right-click the name of a Web application (WAR). Figure 4-12 illustrates where you might click, using the `basic-webapp` WAR as an example.

Figure 4-12 Deployments Portion of the Administration Console Navigation Tree



3. If you are creating the scoped role for all Web applications (WARs), choose Define Role... to display the Select Roles page.

   If you are creating the scoped role for a particular WAR, or a component within a WAR, follow these steps:

   a. Choose Define Role... to display the General tab.

   b. Enter a URL pattern in the text field.

A URL pattern is a path to a specific component within a Web application. Or, you can use `/*` to associate the scoped role with all components (servlets, JSPs, and so on) within the Web application.

c. Click the Define Role... button to display the Select Roles page.

If available, a table of currently defined scoped roles appears in the right pane.

Step 2: Create the Scoped Role

1. Click Configure a new Role....

   Note: If multiple WebLogic Role Mapping providers are configured in the security realm, select which WebLogic Role Mapping provider's database should store information for the new scoped role.

2. On the General tab, enter the name of the scoped role in the Name field.

   Notes: Do not use blank spaces, commas, hyphens, or any characters in this comma-separated list: \t, < >, #, |, &, ~, ?, ( ), { }. Security role names are case sensitive. The BEA convention is that all security role names are singular and the first letter is capitalized.

   The proper syntax for a security role name is as defined for an `Nmtoken` in the *Extensible Markup Language (XML) Recommendation*.

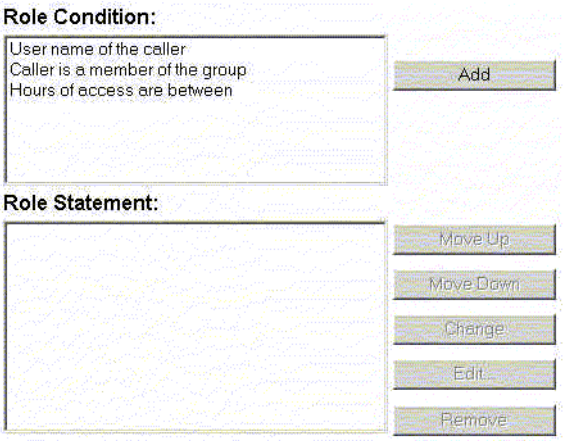
   Warning: If you create a scoped role with the same name as a global role, the scoped role takes precedence over the global role.

3. Click Apply to save your changes.

Step 3: Create the Role Conditions

1. Select the Conditions tab to display the Role Editor page (see Figure 4-13).

Figure 4-13 Role Editor Page



2. In the Role Condition list box, select one of the conditions. (For more information about the different role conditions, see "Components of a Security Role: Role Conditions, Expressions, and Role Statements" on page 4-12.)

   Notes: BEA recommends that you create expressions using the `Caller is a Member of the Group` condition where possible. When a group is used to create a security role, the security role can be granted to all members of the group (that is, multiple users).

   Because the JMS subsystem performs its security check only once and the `Hours of Access are Between` condition requires a subsequent security check, you should not use the `Hours of Access are Between` condition if you are creating a scoped role for a JMS resource.

3. Click Add to display a customized window.

4. If you selected the `Hours of Access are Between` condition, use the Time Constraint window to select start and end times, then click OK. The window closes and an expression appears in the Role Statement list box.

   If you selected one of the other conditions, follow these steps:

   a. Enter the name of a user or group in the Users or Groups window, then click Add. An expression appears in the list box.

      Note: You can repeat this step multiple times to add more than one user or group.

b. If necessary, use the buttons located to the right of the list box to modify the expressions.

   Move Up and Move Down change the ordering of the highlighted user or group name. Change switches the highlighted `and` and `or` statements between expressions. Remove deletes the highlighted user or group name.

c. Click OK to add the expression to the role statement. The window closes and the expression appears in the Role Statement list box.

5. If needed, repeat steps 2 - 4 to add expressions based on different role conditions.
6. If necessary, use the buttons located to the right of the Role Statement list box to modify the expressions:
   - Move Up and Move Down change the ordering of the highlighted expression.
   - Change switches the highlighted `and` and `or` statements between expressions.
   - Edit... reopens the customized window for the highlighted expression and allows you to modify the expression.
   - Remove deletes the highlighted expression.
7. When all the expressions in the Role Statement list box are correct, click Apply.

   Note: You can also click Reset to restore the Role Editor page to the state it was in when the page was first loaded (that is, to undo any of your changes).

Modifying Scoped Roles

To modify a scoped role for a WebLogic resource, follow these steps:

1. Navigate to the Select Roles page for the WebLogic resource, as described in "Step 1: Select the WebLogic Resource" on page 4-19.

   A table that lists all the scoped roles for the WebLogic resource appears in the right pane.

2. Select the scoped role that you want to modify from the table.
3. Select the Conditions tab.

4. Make your changes, using the instructions in "Step 3: Create the Role Conditions" on page 4-27 as a guide.
5. Click Apply to save your changes.

Deleting Scoped Roles

To delete a scoped role for a WebLogic resource, follow these steps:

1. Navigate to the Select Roles page for your WebLogic resource, as described in "Step 1: Select the WebLogic Resource" on page 4-19.

   A table that lists all the scoped roles for the WebLogic resource appears in the right pane.
2. Click the trash can icon that is located in the same row as the scoped role you want to delete.
3. Click Yes to confirm the deletion.
4. Click Continue.

   The Select Roles page no longer shows the deleted scoped role in the table.

CHAPTER

5 Security Policies

A **security policy** is an association between a WebLogic resource and one or more users, groups, or security roles that is designed to protect the WebLogic resource against unauthorized access.

Note: Security policies replace the access control lists (ACLs) and permissions that were used to protect WebLogic resources in previous releases of WebLogic Server.

The following sections provide more information about security policies:

Security Policy Granularity and Inheritance

Security policies are always scoped to a WebLogic resource, but because WebLogic resources are hierarchical, the level at which you define a security policy is up to you. For example, you can define security policies on: entire enterprise applications (EARs); an EJB (Enterprise JavaBean) JAR containing multiple EJBs; a particular EJB within that JAR; or a single method within that EJB.

If you create a security policy for a *type* of WebLogic resource (for example, EJB resources), all new instances of that WebLogic resource inherit the security policy. (For more information about the types of WebLogic resources, see Chapter 2, "Types of WebLogic Resources.") This inheritance of security policies means that you can secure multiple WebLogic resources efficiently. Out of the box, WebLogic Server secures each WebLogic resource type with a default security policy that is inherited by all instances of that WebLogic resource. For more information, see "Default Security Policies" on page 5-3.

A security policy created for a specific instance of a WebLogic resource will override any security policy assigned to the WebLogic resource type. So, if you create a security policy for a particular EJB, this security policy (and not the one you created for the EJB resource type) will be used.

Security Policy Storage and Prerequisites for Use

Security policies are stored in the security provider database of the Authorization provider that is configured in the default (active) security realm. By default, the WebLogic Authorization provider is configured, and security policies are stored in the embedded LDAP server.

When creating a security policy with a user or group, the user or group must be defined in the security provider database of the Authentication provider that is configured in the default security realm. When creating a security policy with a security role, the security role (whether global or scoped) must be defined in the security provider database of the Role Mapping provider that is configured in the default security realm. By default, the WebLogic Authentication and Role Mapping providers are configured, and the default groups and default global roles are stored in the databases for these security providers (also the embedded LDAP server).

Note: For more information about the WebLogic Authentication, Authorization, and Role Mapping providers, see "The WebLogic Security Providers" in the *Introduction to WebLogic Security*.

Default Security Policies

By default, WebLogic Server defines the security policies shown in Table 5-1. These security policies are defined for each type of WebLogic resource described in Chapter 2, "Types of WebLogic Resources," and are based on the default global roles and default groups.

Table 5-1 Default Security Policies for WebLogic Resources

WebLogic Resource	Security Policy
Administrative resources	Default global role: `Admin`
Application resources	None
COM resources	None
EIS resources	Default group: `Everyone`
EJB resources	Default group: `Everyone`
JDBC resources	Default group: `Everyone`
JNDI resources	Default group: `Everyone`
JMS resources	Default group: `Everyone`
Server resources	Default global roles: ■ `Admin` ■ `Operator`
URL resources (previously Web resources, deprecated)	Default group: `Everyone`
Web Service resources	Default group: `Everyone`

Warning: Do not modify the default security policies for Administrative and Server resources to make them more restrictive. Eliminating some of the existing security roles might negatively impact the functioning of WebLogic

Server. However, if you like, you can make the default security policies more inclusive (for example, by adding new security roles).

Note: For more information about the WebLogic resources shown in Table 5-1, see Chapter 2, "Types of WebLogic Resources."

You can add to the default security policies by creating your own, as described in "Working With Security Policies" on page 5-7.

Protected Public Interfaces

The WebLogic Server Administration Console, the `weblogic.Admin` command, and MBean APIs are secured using the default security policies, which are based on the default global roles and default groups described in Table 4-1and Table 4-6. Therefore, to use the Administration Console, a user must belong to one of these default groups or be granted one of these global roles. Additionally, administrative operations that require interaction with MBeans are secured using the MBean protections described in the "Protecting System Administration Operations" section of the *WebLogic Server Administration Guide*. Therefore, interaction with the following protected public interfaces typically must satisfy both security schemes.

- *The WebLogic Server Administration Console*—The WebLogic Security Service verifies whether a particular user can access the Administration Console when the user attempts to log in. If a user attempts to invoke an operation for which they do not have access, they see an Access Denied error.

  For information about using this public interface, see the *Administration Console Online Help*.

- *The weblogic.Admin command*—The WebLogic Security Service verifies whether a particular user has permission to execute a command when the user attempts to invoke the command. If a user attempts to invoke an operation for which they do not have access, WebLogic Server throws a `weblogic.management.NoAccessRuntimeException`, which developers can explicitly catch in their programs. The server sends this exception to its log file, but you can also configure the server to send exceptions to standard out.

  For information about using this public interface, see "Protected MBean Attributes and Operations" on page 4-7 and "weblogic.Admin Command-Line Reference" in the *WebLogic Server Command Line Reference*.

Note: The `weblogic.Admin` command is a convenience utility that abstracts the interaction with the MBean APIs (described below). Therefore, for any administrative task you can perform using the weblogic.Admin command, you can also choose to write yourself using the MBean APIs.

- *MBean APIs*—The WebLogic Security Service verifies whether a particular user has permission to access the API when the user attempts to perform an operation on the MBean. If a user attempts to invoke an operation for which they do not have access, WebLogic Server throws a `weblogic.management.NoAccessRuntimeException`, which developers can explicitly catch in their programs. The server sends this exception to its log file, but you can also configure the server to send exceptions to standard out.

  For information about using these APIs, see "Protected MBean Attributes and Operations" on page 4-7 and *Programming WebLogic Management Services with JMX*.

Components of a Security Policy: Policy Conditions, Expressions, and Policy Statements

A **policy condition** is a condition under which a security policy will be created. The policy conditions that are available in this release of WebLogic Server are:

- `User Name of the Caller`—Creates a condition for a security policy based on a user name. For example, you might create a condition indicating that only the user `John` can access the `Deposit` EJB.
- `Caller is a Member of the Group`—Creates a condition for a security policy based on a group. When a group is used to create a security policy, the security policy is assigned to all members of the group. For example, you might create a condition indicating that only users in the group `FullTimeBankEmployees` can access the `Deposit` EJB.
- `Caller is Granted the Role`—Creates a condition for a security policy based on a security role. For example, you might create a condition indicating

that only users and groups in the `BankTeller` security role can access the `Deposit` EJB.

- `Hours of Access are Between`—Creates a condition for a security policy based on a specified time period. For example, you might create a condition indicating that the `BankTeller` security role can only access the `Deposit` EJB when the bank is open.

These policy conditions, along with the specific information you supply for the condition (such as an actual user name, group, or security role, or start/stop times), are called **expressions**. An example of an expression that you may see in the WebLogic Server Administration Console is shown in Figure 5-1.

Figure 5-1 Expression Example

In this expression example, the first line is the policy condition, the second line is the specific information you supply for the condition—in this case, a group called `FullTimeBankEmployees`.

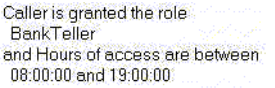
A **policy statement** is the collection of expressions that define who is granted access to a WebLogic resource, and is therefore the main part of any security policy you create. The ability to use multiple expressions means that you can create complex security policies that meet your organization's security requirements. The use of `and` and `or` between these expressions, as well as the ordering of the expressions, is also an important feature:

- `And` is used to specify that all the expressions must be true in order for the security policy to be applied.
- `Or` is used to specify that at least one of the expressions must be true in order for the security policy to be applied.

Notes: The entire policy statement must be true in order for security policy to be applied. More restrictive expressions should come later in a policy statement. WebLogic Server evaluates expressions in a policy statement from left to right.

An example of a policy statement that you may see in the Administration Console is shown in Figure 5-2.

Figure 5-2 Policy Statement Example



In this policy statement example, there are two expressions: the first and second lines contain an expression based on the `Caller is Granted the Role` policy condition, and the third and fourth lines contain another expression based on the `Hours of Access are Between` policy condition.

Working With Security Policies

The following sections provide instructions for working with security policies for the various types of WebLogic resources:

- "Creating Security Policies" on page 5-7
- "Modifying Security Policies" on page 5-20
- "Deleting Security Policies" on page 5-20

Creating Security Policies

Notes: The instructions for working with security policies vary slightly from WebLogic resource to WebLogic resource. Be sure to follow any variations noted in this procedure that pertain to the type of WebLogic resource with which you are working. For more information, see Chapter 2, "Types of WebLogic Resources."

In this release of WebLogic Server, you must keep track of the security policies you create. There is currently no listing mechanism for previously created security policies in the WebLogic Server Administration Console.

To create a security policy for a WebLogic resource, follow these steps:

- "Step 1: Select the WebLogic Resource" on page 5-8

- "Step 2: Create the Policy Conditions" on page 5-18

Step 1: Select the WebLogic Resource

Follow the instructions in the appropriate section to select the type of WebLogic resource:

- "Administrative Resources" on page 5-8
- "Application Resources" on page 5-9
- "COM Resources" on page 5-10
- "EIS Resources" on page 5-11
- "EJB Resources" on page 5-11
- "JDBC Resources" on page 5-13
- "JMS Resources" on page 5-14
- "JNDI Resources" on page 5-15
- "Server Resources" on page 5-16
- "URL Resources" on page 5-17

Administrative Resources

In the left pane of the WebLogic Server Administration Console, right-click the name of the WebLogic Server domain (for example, `examples`), and choose Define Policy... to display the Policy Editor page (see Figure 5-3, "Policy Editor Page," on page 5-9).

Notes: In this version of WebLogic Server, you can only secure the `unlockuser` method. For more information about user lockouts, see "Protecting User Accounts" in *Managing WebLogic Security*.

Notice the `Caller is Granted the Role: Admin` policy statement that the Administrative resource you selected has inherited from the default security policy associated with the Administrative resource type. If you proceed to "Step 2: Create the Policy Conditions" on page 5-18, you will be overriding this default security policy. For more information, see "Default Security Policies" on page 5-3 and "Security Policy Granularity and Inheritance" on page 5-1.

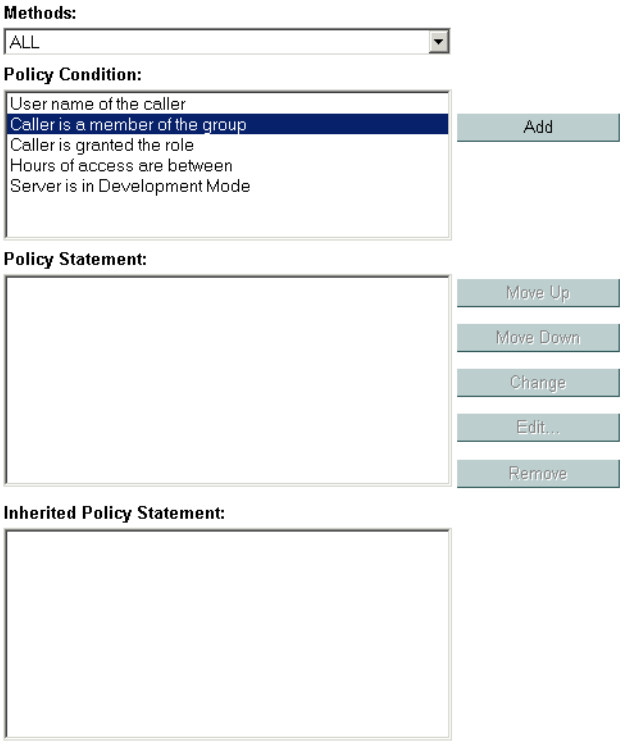
Application Resources

1. In the left pane of the WebLogic Server Administration Console, expand Deployments, then Applications.

   Note: You can optionally expand the enterprise application (EAR) for which you are creating a scoped role to see the different types of WebLogic resources it contains.

2. Right-click the name of the enterprise application and choose Define Policy... to display the Policy Editor page (see Figure 5-3).

Figure 5-3 Policy Editor Page



Note: Notice that there are no default policy statements for Application resources. (For more information, see "Default Security Policies" on page 5-3.)

COM Resources

If you want to create a security policy for a package of EJB classes (such as `ejb20.basic.beanManaged.*`) that will be accessed by a COM client, follow these steps:

1. In the left pane of the WebLogic Server Administration Console, expand Deployments, then EJB.

   The EJB node expands to show the EJB JARs that are currently deployed.

2. Right-click the name of an EJB JAR containing the EJB that will be used to access the package, and choose Define Policies and Roles for Individual Beans... to display a list of EJBs.

3. Click the [Define JCOM Policies] link that is located in the same row as the EJB that will be used to access the package.

   The General tab's COM Class field already shows the name of the package for which you want to create the security policy.

   Note: The value in the COM class field is a Java class or package name that is exposed to COM via the jCOM bridge.

4. Click the Define Policy... button to display the Policy Editor page (see Figure 5-3, "Policy Editor Page," on page 5-9).

Note: If you create a security policy for a package of EJB classes that will be accessed by a COM client and want to use scoped roles in the `Caller is Granted the Role` condition, be sure to use the scoped role you associated with the package of EJB classes (described in "COM Resources" on page 4-20).

If you want to create a security policy for a package of Java classes (such as `java.util.*`) or individual classes (such as `java.util.Collection`) that will be accessed by a COM client, follow these steps:

1. In the left pane of the WebLogic Server Administration Console, expand Services.

2. Right-click the JCOM node and choose Define Policy....

3. On the General tab, in the COM class field, enter the name of the Java class or package you want to protect, then click the Define Policy... button to display the Policy Editor page (see Figure 5-3, "Policy Editor Page," on page 5-9).

Notes: The value you enter in the COM class field is a Java class or package name that is exposed to COM via the jCOM bridge.

Notice that there are no default policy statements for COM resources. (For more information, see "Default Security Policies" on page 5-3.)

EIS Resources

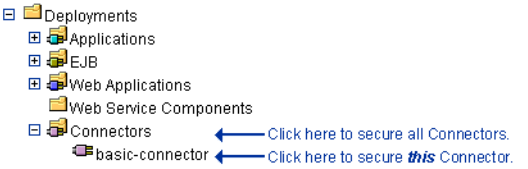
1. In the left pane of the WebLogic Server Administration Console, expand Deployments.

   The Deployments node expands to show the types of WebLogic resources that can be deployed.

2. Right-click at the level of the EIS resource at which you want to create the security policy, and choose Define Policy... to display the Policy Editor page (see Figure 5-3, "Policy Editor Page," on page 5-9).

   To secure *all* the Connectors with a single security policy, right-click Connectors. To secure a *particular* Connector, expand Connectors, then right-click the name of a Connector. Figure 5-4 illustrates where you might click, using the `basic-connector` Connector as an example.

Figure 5-4 Deployments Portion of the Administration Console Navigation Tree



Note: Notice the `Caller is Granted the Role: Everyone` policy statement that the EIS resource you selected has inherited from the default security policy associated with the EIS resource type. If you proceed to "Step 2: Create the Policy Conditions" on page 5-18, you will be overriding this default security policy. For more information, see "Default Security Policies" on page 5-3 and "Security Policy Granularity and Inheritance" on page 5-1.

EJB Resources

Note: These instructions also apply to Message-driven Beans (MDBs).

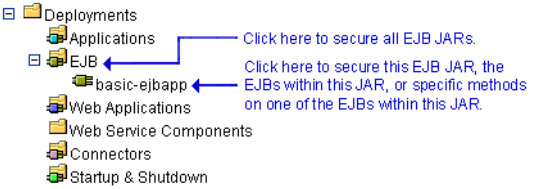
1. In the left pane of the WebLogic Server Administration Console, expand Deployments.

   The Deployments node expands to show the types of WebLogic resources that can be deployed.

2. Right-click at the level of the EJB resource at which you want to create the security policy.

   To secure *all* the EJB JARs with a single security policy, right-click EJB. To secure a *particular* EJB JAR, an EJB within a JAR, or a method on one of the EJBs within a JAR, expand EJB, then right-click the name of an EJB JAR. Figure 5-5 illustrates where you might click, using the `basic-ejbapp` JAR as an example.

Figure 5-5 Deployments Portion of the Administration Console Navigation Tree



3. If you are creating a security policy for all EJB JARs or for a particular EJB JAR, choose Define Policy... to display the Policy Editor page (see Figure 5-3, "Policy Editor Page," on page 5-9).

   If you are creating the security policy for a particular EJB within an EJB JAR, or a method on one of the EJBs within the JAR, follow these steps:

   a. Choose Define Policies and Roles for Individual Beans... to display a list of EJBs.

   b. Click the [Define Policies] link that corresponds to the particular EJB you want to secure (regardless of whether you want to secure the entire EJB or a particular method within the EJB) to display the Policy Editor page (see Figure 5-3, "Policy Editor Page," on page 5-9).

      Note: Notice the `Caller is Granted the Role: Everyone` policy statement that the EJB resource you selected has inherited from the default security policy associated with the EJB resource type. If you proceed to "Step 2: Create the Policy Conditions" on page 5-18, you will be overriding this default security policy. For more information,

see "Default Security Policies" on page 5-3 and "Security Policy Granularity and Inheritance" on page 5-1.

4. If you are securing a particular EJB within an EJB JAR, specify which EJB method you want to protect, or select ALL to protect all methods.

JDBC Resources

1. In the left pane of the WebLogic Server Administration Console, expand Services, then JDBC.

   The JDBC node expands to show nodes for various JDBC components (connection pools, Multipools, and data sources).

2. Right-click at the level of the JDBC resource at which you want to create the security policy, and choose Define Policy... to display the Policy Editor page (see Figure 5-3, "Policy Editor Page," on page 5-9).

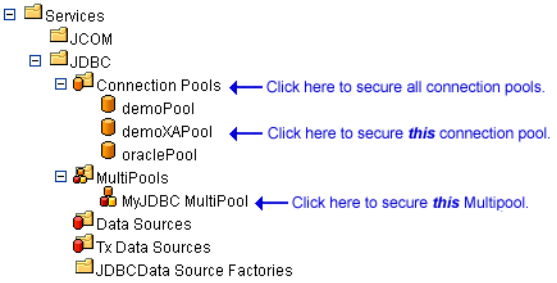
   To secure *all* the connection pools with a single security policy, right-click Connection Pools. To secure a *particular* connection pool, expand Connection Pools, then right-click the name of a connection pool. To secure individual MultiPools, expand MultiPools, then right-click the name of a MultiPool.

   Notes: You cannot secure all MultiPools with a single security policy.

   If a security policy controls access to a connection pool that is in a MultiPool, access checks will be performed at both levels of the JDBC resource hierarchy (once at the MultiPool level, and again at the individual connection pool level). As with all types of WebLogic resources, this double checking ensures that the most restrictive security policy controls access.

   Figure 5-6 illustrates where you might click, using various connection pools and a MultiPool as an example.

Figure 5-6 Services Portion of the Administration Console Navigation Tree



Note: Notice the `Caller is Granted the Role: Everyone` policy statement that the JDBC resource you selected has inherited from the default security policy associated with the JDBC resource type. If you proceed to "Step 2: Create the Policy Conditions" on page 5-18, you will be overriding this default security policy. For more information, see "Default Security Policies" on page 5-3 and "Security Policy Granularity and Inheritance" on page 5-1.

3. If you are securing a particular connection pool, use the Methods drop-down menu to specify a method that you want to protect, or select ALL to protect all methods.

JMS Resources

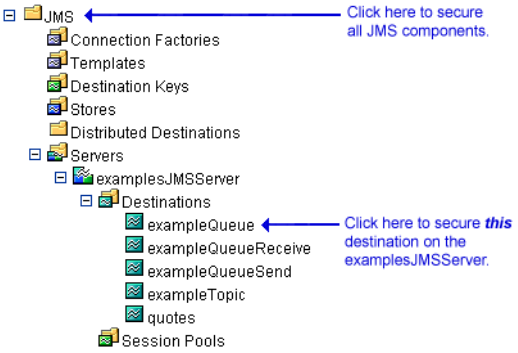
1. In the left pane of the WebLogic Server Administration Console, expand Services, then JMS.

   The JMS node expands to show nodes for various JMS components (connection factories, templates, destination keys, and so on).

2. Right-click at the level of the JMS resource for which you want to create the security policy, and choose Define Policy... to display the Policy Editor page (see Figure 5-3, "Policy Editor Page," on page 5-9).

   To create a security policy for *all* JMS components, right-click JMS. To create a security policy for a *particular* destination (JMS queue or JMS topic) on a JMS server, expand Servers, then the JMS server and the Destinations node, then right-click the name of a destination. Figure 5-7 illustrates where you might click, using various destinations on the `examplesJMSServer` as an example.

Figure 5-7 Services Portion of the Administration Console Navigation Tree



Note: Notice the `Caller is Granted the Role: Everyone` policy statement that the JMS resource you selected has inherited from the default security policy associated with the JMS resource type. If you proceed to "Step 2: Create the Policy Conditions" on page 5-18, you will be overriding this default security policy. For more information, see "Default Security Policies" on page 5-3 and "Security Policy Granularity and Inheritance" on page 5-1.

3. If you are securing a particular destination on a JMS server, use the Methods drop-down menu to specify a method that you want to protect, or select ALL to protect all methods.

JNDI Resources

1. In the left pane of the WebLogic Server Administration Console, expand Servers.

   The Servers node expands to show the servers available in the current WebLogic Server domain.

2. Right-click the name of the server that contains the JNDI resource for which you want to create the security policy. (For example, `myserver`.)

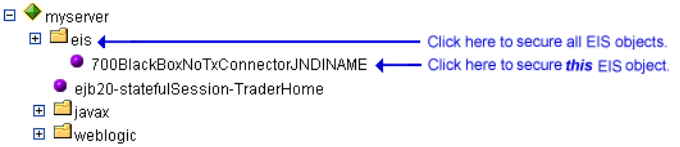
3. From the menu that appears, select the View JNDI Tree option.

   The JNDI tree for the server appears in a new Administration Console window.

4. In the new Administration Console window, right-click at the level of the JNDI tree at which you want to create the security policy, and choose Define Policy... to display the Policy Editor page (see Figure 5-3, "Policy Editor Page," on page 5-9).

   To secure *a group of* objects, right-click the node that represents that object type. To secure a *particular* object, expand the node that represents that object, then right-click the name of an object. Figure 5-8 illustrates where you might click, using the `examplesServer` JNDI tree as an example.

Figure 5-8 New Administration Console Window for examplesServer JNDI Tree



Note: Notice the `Caller is Granted the Role: Everyone` policy statement that the JNDI resource you selected has inherited from the default security policy associated with the JNDI resource type. If you proceed to "Step 2: Create the Policy Conditions" on page 5-18, you will be overriding this default security policy. For more information, see"Default Security Policies" on page 5-3 and "Security Policy Granularity and Inheritance" on page 5-1.

5. Using the Methods drop-down menu, specify which JNDI method you want to protect, or select ALL to protect all methods.

Server Resources

1. In the left pane of the WebLogic Server Administration Console, expand Servers.

   The Servers node expands to show the different server resources that can be secured.

2. Right-click at the level of the Server resource at which you want to create a security policy, and choose Define Policy... to display the Policy Editor page (see Figure 5-3, "Policy Editor Page," on page 5-9).

   To create a security policy for *all* servers, right-click Servers. To create a security policy for a *particular* server, expand Servers, then right-click the name of a server. Figure 5-9 illustrates where you might click, using the `examplesServer` as an example.

Figure 5-9 Servers Portion of the Administration Console Navigation Tree

Note: Notice the `Caller is Granted the Role: Admin or Caller is Granted the Role: Operator` policy statement that the server resource you selected has inherited from the default security policy associated with the server resource type. If you proceed to "Step 2: Create the Policy Conditions" on page 5-18, you will be overriding this default security policy. For more information, see "Default Security Policies" on page 5-3 and "Default Security Policies" on page 5-3.

3. Using the Methods drop-down menu, specify a method that you want to protect, or select ALL to protect all methods.

URL Resources

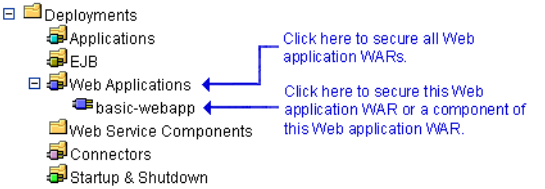
1. In the left pane of the WebLogic Server Administration Console, expand Deployments.

   The Deployments node expands to show the types of WebLogic resources that can be deployed.

2. Right-click at the level of the Web application resource at which you want to create the security policy.

   To secure *all* the Web applications (WARs) with a single security policy, right-click Web Applications. To secure a *particular* WAR or a component of a WAR (for example, a specific servlet or JSP), expand Web Applications, then right-click the name of a Web application (WAR). Figure 5-10 illustrates where you might click, using the `basic-webapp` WAR as an example.

Figure 5-10 Deployments Portion of the Administration Console Navigation Tree



3. If you are creating a security policy for *all* Web applications (WARs), choose Define Policy... to display the Policy Editor page (see Figure 5-3, "Policy Editor Page," on page 5-9).

   If you are creating the security policy for a particular WAR or component of the WAR, follow these steps:

   a. Choose Define Policy....

   b. On the General tab, enter a URL pattern in the text field.

      Note: A URL pattern is a path to a specific servlet within a Web application. Or, you can use `/*` to protect all servlets within the Web application.

   c. Click the Define Policy... button to display the Policy Editor page (see Figure 5-3, "Policy Editor Page," on page 5-9).

      Note: Notice the `Caller is Granted the Role: Everyone` policy statement that the URL resource you selected has inherited from the default security policy associated with the URL resource type. If you proceed to "Step 2: Create the Policy Conditions" on page 5-18, you will be overriding this default security policy. For more information, see "Default Security Policies" on page 5-3 and "Security Policy Granularity and Inheritance" on page 5-1.

4. If you are securing a particular WAR or component of a particular WAR, specify which method you want to protect, or select ALL to protect all methods.

Step 2: Create the Policy Conditions

1. In the Policy Condition list box, click one of the conditions. (For more information about the different policy conditions, see "Components of a Security Policy: Policy Conditions, Expressions, and Policy Statements" on page 5-5.)

   Note: BEA recommends that you create expressions using the `Caller is Granted the Role` condition. Basing expressions on security roles allows you to create one security policy that takes into account multiple users or groups, and is a more efficient method of management.

2. Click Add to display a customized window.

3. If you selected the `Hours of Access are Between` condition, use the Time Constraint window to select start and end times, then click OK. The window closes and an expression appears in the Policy Statement list box.

Note: Because the JMS subsystem performs its security check only once and this condition requires a subsequent security check, you should not use the `Hours of Access are Between` condition if you are securing a JMS resource.

If you selected one of the other conditions, follow these steps:

a. Use the Users, Groups, or Roles window to enter the name of a user, group, or security role, then click Add. An expression appears in the list box.

Note: You can repeat this step multiple times to add more than one user, group, or security role.

b. If necessary, use the buttons located to the right of the list box to modify the expressions:

Move Up and Move Down change the ordering of the highlighted user or group name. Change switches the highlighted `and` and `or` statements between expressions. Remove deletes the highlighted user or group name.

c. Click OK to add the expression to the policy statement. The window closes and the expression appears in the Policy Statement list box.

4. If desired, repeat steps 1 - 3 to add expressions based on different policy conditions.

5. If necessary, use the buttons located to the right of the Policy Statement list box to modify the expressions:

- Move Up and Move Down change the ordering of the highlighted expression.
- Change switches the highlighted `and` and `or` statements between expressions.
- Edit... reopens the customized window for the highlighted expression and allows you to modify the expression.
- Remove deletes the highlighted expression.

6. When all the expressions in the Policy Statement list box are correct, scroll down the page and click Apply.

Note: You can also click Reset to restore the Policy Editor page to the state it was in when the page was first loaded (that is, to undo any of your changes).

Modifying Security Policies

To modify a security policy for a WebLogic resource, follow these steps:

1. Navigate to the Policy Editor page for the WebLogic resource, as described in "Step 1: Select the WebLogic Resource" on page 5-8.

   Note: Pay special attention to the Inherited Policy Statement list box to ensure that you understand which security policies you may be overriding.

2. Make your changes, using "Step 2: Create the Policy Conditions" on page 5-18 as a guide.

3. Click Apply to save your changes.

Deleting Security Policies

To delete a security policy for a WebLogic resource, follow these steps:

1. Navigate to the Policy Editor page for the WebLogic resource, as described in "Step 1: Select the WebLogic Resource" on page 5-8.

2. Click Delete to delete the entire security policy.

3. Click Apply to save your changes.

6 Example: Securing URL (Web) Resources Using the Administration Console

In this example, you will restrict access to all the deployed Web applications to a user who has been granted a default global security role. Next, you will further restrict access to the `basicauth` Web application to a different user. Last, you will tighten security even further on a particular JSP within the Web application (`welcome.jsp`), using a scoped role.

Note: Prior to reviewing this example, you should read: "Techniques for Securing URL and EJB Resources" on page 2-7; "Prerequisites for Securing URL and EJB Resources" on page 2-9; and "Types of Security Roles: Global Roles and Scoped Roles" on page 4-3.

To secure these URL (Web) resources using the WebLogic Server Administration Console, follow these steps:

Step 1: Specify Server and Prerequisite Settings

1. Set the `fullyDelegateAuthorization` flag equal to `true`, using the instructions in "How to Change the fullyDelegateAuthorization Flag" on page 2-11.

   Note: Recall what this setting means: you are telling WebLogic Server that you want the WebLogic Security Service to perform security checks on *all* URL (Web) and EJB resources. For more information, see "Understanding the fullyDelegateAuthorization Flag" on page 2-10.

2. From the Windows Start menu, select Programs →BEA WebLogic Platform 7.0 →WebLogic Server 7.0 →Server Tour and Examples →Launch Examples Server to start the server called `examplesServer`.

   The `fullyDelegateAuthorization` flag appears in the console as the `examplesServer` starts, and the BEA WebLogic Server Out-of-the-Box Examples Index Page opens in your browser.

3. Click the Administration Console link, located at the top of the BEA WebLogic Server Out-of-the-Box Examples Index Page.

4. Click the Sign In button to sign in to the Administration Console for the `examplesServer`.

5. Using the navigation tree at the left side of the Administration Console, expand Security, then Realms.
6. Click the `myrealm` security realm.
7. On the General tab, click the Ignore Security Data in Deployment Descriptors check box. (That is, you should be putting a check mark in the box.)

   Note: Recall what this setting means: you are telling WebLogic Server that you will set security for Web application and EJB resources using the Administration Console, not deployment descriptors. For more information, see "Understanding the Ignore Security Data in Deployment Descriptors Check Box" on page 2-14.

8. Click Apply to save your changes.

Step 2: Create Users

1. Using the navigation tree at the left side of the Administration Console, expand Security, then Realms.
2. Expand the `myrealm` security realm.
3. Click Users.

   The Select Users displays all the users currently defined in the WebLogic Authentication provider's database.

4. Click the Configure a New User... link to display the Create User page.
5. On General tab, in the Name field, type: `Tom`
6. If desired, enter a description of the user in the Description field.
7. In the Password and Confirm Password fields, type: `webexample`
8. Click Apply to save your changes.
9. Repeat steps 4 - 8 to create a user named `Neil`.
10. Using the navigation tree, click Users, and confirm that users `Tom` and `Neil` have been added.

The Select Users page shows that Tom and Neil have been added to the WebLogic Authentication provider's database.

Step 3: Add a User to a Group

Note: Instead of creating a new group, this step uses one of WebLogic Server's default groups.

1. On the Select Users page, click the hyperlinked user name `Tom`.
2. Click the Groups tab.
3. In the Possible Groups list box, click the `Administrators` group to highlight it.
4. Click the highlighted arrow to move the `Administrators` group from the Possible Groups list box to the Current Groups list box.
5. Click Apply to save your changes.

Note: Do *not* add `Neil` to the `Administrators` group.

Step 4: Grant a Global Role to the Group

Note: Because the default group called `Administrators` is automatically granted the default global role called `Admin`, there is no need to create a global role nor grant that global role to the `Administrators` group.

If you want to verify that this is done, however, follow these steps:

1. Using the navigation tree, click Roles.

   The Select Roles page displays all the global roles currently defined in the WebLogic Role Mapping provider's database.

2. Click the hyperlinked global role name `Admin`.

3. Click the Conditions tab.

   The Role Statement list box reads:

   ```
   Caller is a Member of the Group
      Administrators
   ```

Step 5: Create a Security Policy for All URL (Web) Resources Using the Global Role

1. Using the navigation tree, expand Deployments, then right-click Web applications.
2. From the menu, select the Define Policy... option to display the Policy Editor page.

   Note: Recall what this option means: you are creating a security policy that will encompass *all* the deployed URL (Web) resources and their components.

3. In the Policy Condition list box, highlight: `Caller is Granted the Role`.
4. Click Add to open the Roles window.
5. In the Enter Role Name field, type: `Admin`
6. Click Add, then OK.

   The Roles window closes. The Policy Statement list box reads:

   ```
   Caller is a Member of the Group
      Everyone
   and Caller is Granted the Role
      Admin
   ```

   Note: The `Caller is a Member of the Group` policy condition that appears is part of the default security policy for URL resources. For more information, see "Default Security Policies" on page 5-3.

7. Highlight the `Caller is a Member of the Group` policy condition, then click Remove.

The Policy Statement list box reads:

```
Caller is Granted the Role
   Admin
```

8. Click Apply to save your changes.

Step 6: Attempt to Access a Web Application

Note: All instructions provided in this section assume that you are working in a Windows environment.

1. Obtain the "Basic Authentication Sample Web Application" (available under "Code Samples: WebLogic Server" on the *dev2dev Web site*).
2. Unzip the `basicauth.zip` file to a temporary directory (for example, `C:\basicauth`).
3. Deploy the `basicauth` Web application and target it to the `examplesServer`.

   Note: For instructions about how to deploy Web applications, see "Deploying J2EE Applications with the Administration Console" in *Developing WebLogic Server Applications*.
4. Open a new Web browser and type: `http://localhost:7001/basicauth`.

   The browser prompts you for a username and password.
5. In the username field, type: `Neil`, and in the password field, type: `webexample`, then click OK.

   The browser re-prompts you for a username and password.
6. In the username field, type: `Tom`, and in the password field, type: `webexample`, then click OK.

   The browser displays a page like the one shown in *Figure 6-1*.

Figure 6-1 Browser-Based Authentication Example Web Page

This result occurs because you just secured all URL (Web) resources (including the `basicauth` Web application) with a security policy based on the global security role `Admin`, which user `Tom` is granted but user `Neil` is not.

Note: If you accidentally close your Web browser after this step, you can get back into the Administration Console by typing: `http://localhost:7001/console` and clicking the Sign In button. If you accidentally close the console window that runs WebLogic Server, and attempt to use steps 1-3 in "Step 1: Specify Server and Prerequisite Settings" on page 6-2, you will notice that you first have to log in using `Tom/webexample`, because by securing all Web applications in this step, you also secured the `examplesWebapp`.

Step 7: Restrict Access to the basicauth Web Application

1. Using the navigation tree at the left side of the Administration Console, expand Web Applications, then right-click `basicauth`.
2. From the menu, select the Define Policy... option.

   Note: Recall what this option means: you can create a security policy for a particular Web application or a particular component within the Web application.
3. On the General tab, in the URL Pattern field, type: `/*`

   Note: The URL pattern of `/*` will secure all components within the `basicauth` Web application, including JSPs and servlets.
4. Click the Define Policy... button to proceed.

5. In the Policy Condition list box, highlight: `User Name of the Caller`.

   Note: Do *not* modify the value shown in the Methods drop-down menu. (It should read: `ALL`.)

6. Click Add to open the Users window.

7. In the Enter User Name field, type: `Neil`

8. Click Add, then OK.

   The Users window closes. The Policy Statement list box reads:

   ```
   User Name of the Caller
      Neil
   ```

   Note: Recall that by defining this security policy on the `basicauth` Web application, you are overriding the security policy that has already been defined for all URL (Web) resources in "Step 5: Create a Security Policy for All URL (Web) Resources Using the Global Role." Specifically, you are overriding the inherited policy statement of:

   ```
   Caller is Granted the Role
      Admin
   ```

   that is shown in the Inherited Policy Statements list box.

9. Click Apply to save your changes.

10. Repeat steps 4-6 in "Step 6: Attempt to Access a Web Application" on page 6-6.

    The `basicauth` Web application behaves in the opposite way. In other words, when you attempt to access the `basicauth` Web application with the user `Tom`, you are re-prompted for a username and password. When you attempt to gain access with user `Neil`, the browser displays the page shown in Figure 6-1, but says "welcome" to `Neil`.

    This result occurs because you just secured all components in the `basicauth` Web application with a security policy based on a particular user (in this case, the user `Neil`).

Step 8: Create a Scoped Role

1. Using the navigation tree at the left side of the Administration Console, right-click `basicauth`.
2. From the menu, select the Define Role... option.

   Note: Recall what this option means: you can create a security role that is scoped to a particular Web application. Thereafter, the scoped role can only be used in a security policy for this Web application.
3. On the General tab, in the URL Pattern field, type: `/*`

   Note: The URL pattern of `/*` will scope the security role to all components within the `basicauth` Web application, including JSPs and servlets.
4. Click the Define Role... button to proceed.
5. Click the Configure a New Role... link to display the Create Role page.
6. On General tab, in the Name field, type: `AppAdmin`
7. Click Apply to save your changes.

Step 9: Grant the Scoped Role to a Group

1. Click the Conditions tab.
2. In the Role Condition list box, highlight: `Caller is a Member of the Group`.
3. Click Add to open the Groups window.
4. In the Enter Group Name field, type: `Administrators`
5. Click Add, then OK.

   The Groups window closes. The Role Statement list box reads:

   ```
   Caller is a Member of the Group
   ```

```
Administrators
```

6. Click Apply to save your changes.

Step 10: Restrict Access to the welcome JSP Using the Scoped Role

1. Using the navigation tree at the left side of the Administration Console, right-click `basicauth`.
2. From the menu, select the Define Policy... option.

   Note: Recall what this option means: you can create a security policy for a particular Web application or a particular component within the Web application.
3. On the General tab, in the URL Pattern field, type: `/welcome.jsp`
4. Click the Define Policy... button to proceed.
5. In the Policy Condition list box, highlight: `Caller is Granted the Role`.

   Note: Do *not* modify the value shown in the Methods drop-down menu. (It should read: `ALL`.)
6. Click Add to open the Roles window.
7. In the Enter Role Name field, type: `AppAdmin`
8. Click Add, then OK.

   The Roles window closes. The Policy Statement list box reads:

   ```
   Caller is Granted the Role
      AppAdmin
   ```

   Note: Recall that by defining this security policy on the `welcome.jsp`, you are overriding the security policy that has already been defined for the `basicauth` Web application in "Step 7: Restrict Access to the basicauth Web Application." Specifically, you are overriding the inherited policy statement of:

   ```
   User Name of the Caller
   ```

```
Neil
```

that is shown in the Inherited Policy Statements list box.

9. Click Apply to save your changes.
10. Repeat steps 4-6 in "Step 6: Attempt to Access a Web Application" on page 6-6.

    The `basicauth` Web application behaves in the opposite way. In other words, when you attempt to access the `basicauth` Web application's `welcome.jsp` page with the user `Neil`, you are re-prompted for a username and password. When you attempt to gain access with user `Tom`, the browser displays the page shown in Figure 6-1.

    This result occurs because you just secured the `welcome.jsp` page with a security policy based on the scoped security role `AppAdmin`, which user `Tom` is granted but user `Neil` is not.

CHAPTER

7 Example: Securing Enterprise JavaBean (EJB) Resources

In this example, you will restrict access to all the EJBs in the `ejb20_basic_statelessSession` JAR to a user who has been granted a global security role you created. Next, you will further restrict access to the `statelessSession` EJB (contained within this EJB JAR) to a different user. Last, you will tighten security on a particular EJB methods (the `create()` and `buy()` methods) even further.

Note: Prior to reviewing this example, you should read: "Techniques for Securing URL and EJB Resources" on page 2-7; "Prerequisites for Securing URL and EJB Resources" on page 2-9; and "Types of Security Roles: Global Roles and Scoped Roles" on page 4-3.

To secure these Enterprise JavaBean (EJB) resources using the WebLogic Server Administration Console, follow these steps:

Step 1: Specify Server and Prerequisite Settings

1. Set the `fullyDelegateAuthorization` flag equal to `true`, using the instructions in "How to Change the fullyDelegateAuthorization Flag" on page 2-11.

   Note: Recall what this setting means: you are telling WebLogic Server that you want the WebLogic Security Service to perform security checks on *all* URL (Web) and EJB resources. For more information, see "Understanding the fullyDelegateAuthorization Flag" on page 2-10.

2. From the Windows Start menu, select Programs →BEA WebLogic Platform 7.0 →WebLogic Server 7.0 →Server Tour and Examples →Launch Examples Server to start the server called `examplesServer`.

   The `fullyDelegateAuthorization` flag appears in the console as the examplesServer starts, and the BEA WebLogic Server Out-of-the-Box Examples Index Page opens in your browser.

3. Click the Administration Console link, located at the top of the BEA WebLogic Server Out-of-the-Box Examples Index Page.

4. Click the Sign In button to sign in to the Administration Console for the `examplesServer`.

5. Using the navigation tree at the left side of the Administration Console, expand Security, then Realms.

6. Click the `myrealm` security realm.

7. On the General tab, click the Ignore Security Data in Deployment Descriptors check box. (That is, you should be putting a check mark in the box.)

   Note: Recall what this setting means: you are telling WebLogic Server that you will set security for Web application and EJB resources using the Administration Console, not deployment descriptors. For more information, see "Understanding the Ignore Security Data in Deployment Descriptors Check Box" on page 2-14.

8. Click Apply to save your changes.

Step 2: Create a Group

1. Using the navigation tree at the left side of the Administration Console, expand Security, then Realms.
2. Expand the `myrealm` security realm.
3. Click Groups.

   The Select Groups page displays all the groups currently defined in the WebLogic Authentication provider's database.

4. Click the Configure a New Group... link to display the Create Group page.
5. On General tab, in the Name field, type: `Testers`
6. If desired, enter a description of the group in the Description field.
7. Click Apply to save your changes.

Step 3: Create Users

1. Using the navigation tree, click Users.

   The Select Users page displays all the users currently defined in the WebLogic Authentication provider's database.

2. Click the Configure a New User... link to display the Create User page.
3. On General tab, in the Name field, type: `Stephanie`
4. If desired, enter a description of the user in the Description field.
5. In the Password and Confirm Password fields, type: `ejbexample`
6. Click Apply to save your changes.
7. Repeat steps 2 - 6 to create a user named `Jen`.
8. Using the navigation tree, click Users, and confirm that users `Stephanie` and `Jen` have been added.

   The Select Users page shows that Stephanie and Jen have been added to the WebLogic Authentication provider's database.

Step 4: Add a User to the Group

1. On the Select Users page, click the hyperlinked user name `Stephanie`.
2. Click the Groups tab.
3. In the Possible Groups list box, highlight the `Testers` group.
4. Click the highlighted arrow to move the `Testers` group from the Possible Groups list box to the Current Groups list box.
5. Click Apply to save your changes.

Note: Do *not* add `Jen` to the `Testers` group.

Step 5: Create a Global Role

1. Using the navigation tree, click Roles.

The Select Roles page displays all the global roles currently defined in the WebLogic Role Mapping provider's database.

2. Click the Configure a New Role... link to display the Create Role page.
3. On General tab, in the Name field, type: `QA`
4. Click Apply to save your changes.

Step 6: Grant the Global Role to the Group

1. Click the Conditions tab.
2. In the Role Condition list box, highlight: `Caller is a Member of the Group`.
3. Click Add to open the Groups window.
4. In the Enter Group Name field, type: `Testers`
5. Click Add, then OK.

   The Groups window closes. The Role Statement list box reads:

   ```
   Caller is a Member of the Group
      Testers
   ```

6. Click Apply to save your changes.

Step 7: Create a Security Policy for the statelessSession EJB JAR Using the Global Role

1. Using the navigation tree, expand Deployments, then EJB.

2. Right-click `ejb20_basic_statelessSession.jar`.
3. From the menu, select the Define Policy... option.

   Note: Recall what this option means: you are creating a security policy at the EJB JAR level, which includes all EJBs within the JAR, and all methods within those EJBs.
4. In the Policy Condition list box, highlight: `Caller is Granted the Role`.
5. Click Add to open the Roles window.
6. In the Enter Role Name field, type: `QA`
7. Click Add, then OK.

   The Roles window closes. The Policy Statement list box reads:

   ```
   Caller is Granted the Role
      QA
   ```

   Note: Recall that by defining this security policy on the `ejb20_basic_statelessSession.jar`, you are overriding any security policies that have already been defined for the EJB resource type. Specifically, you are overriding the inherited policy statement:

   ```
   Caller is a Member of the Group
      Everyone
   ```

   that is shown in the Inherited Policy Statements list box.

   This `Caller is a Member of the Group` policy condition is part of the default security policy for EJB resources. For more information, see "Default Security Policies" on page 5-3.
8. Click Apply to save your changes.

Step 8: Attempt to Access EJBs Through a Client Application

Note: All instructions provided in this section assume that you are working in a Windows environment.

1. Open a DOS shell and type: `cd WL_HOME\samples\server\config\examples`, where *WL_HOME* is the top-level installation directory for WebLogic Platform.
2. Type: `setExamplesEnv.cmd` to set your environment.
3. Type: `cd ..\..\src\examples\security\jaas`
4. Type: `ant` to build the example.
5. Copy the `sample_jaas.config` file from the *WL_HOME*`\samples\server\src\examples\security\jaas` directory to the *JAVA_HOME*`\jre\lib\security` directory, where *JAVA_HOME* is the location of your Java SDK installation.
6. Edit the `java.security` file (also located in the *JAVA_HOME*`\jre\lib\security` directory) and add the following line to the very end of the file (all on one line):

   ```
   login.config.url.1=file:${java.home}/lib/security/
   sample_jaas.config
   ```

7. Restart the `examplesServer`. (For help, see "Starting and Stopping WebLogic Servers" in the *WebLogic Server Administration Guide*.)
8. In the *WL_HOME*`\samples\server\src\examples\security\jaas` directory, edit the `build.xml` file as follows:
   a. Scroll to the end of the file and locate the line: `<target name="run">` (This is shown in bold in Listing 7-1.)
   b. In the `<arg line>` element, change the user name and password (currently shown as `weblogic weblogic`) to `Stephanie ejbexample`. (This is also shown in bold in Listing 7-1.)
   c. Save the `build.xml` file.

Listing 7-1 Relevant Portion of the build.xml File

```
<!-- Run the example -->
   <target name="run" >
      <java classname="examples.security.jaas.SampleClient"
      fork="yes" failonerror="true">
         <arg line="t3://localhost:${PORT} weblogic weblogic"/>
         <classpath>
            <pathelement path="${CLASSPATH};${CLIENT_CLASSES}/
```

```
            ejb20_basic_statelessSession_client.jar;
            ${CLIENT_CLASSES}/utils_common.jar"/>
         </classpath>
      </java>
   </target>
```

9. In the same directory (*WL_HOME*\samples\server\src\examples\security\jaas) type: `ant run`

   You should see the following output similar to the following:

```
Buildfile: build.xml

run:
   [java] username: Stephanie
   [java] password: **********
   [java] URL: t3://localhost:7001
   [java] Creating a trader
   [java] Buying 100 shares of BEAS.
   [java] Buying 200 shares of MSFT.
   [java] Buying 300 shares of AMZN.
   [java] Buying 400 shares of HWP.
   [java] Selling 100 shares of BEAS.
   [java] Selling 200 shares of MSFT.
   [java] Selling 300 shares of AMZN.
   [java] Selling 400 shares of HWP.
   [java] Removing the trader

BUILD SUCCESSFUL

Total time: 5 seconds
```

   This result occurs because the client application calls an EJB that is stored in the `ejb20_basic_statelessSession.jar` you just secured with a security policy.

10. Repeat steps 8 - 9, but use `Jen ejbexample` as the user name and password in the `build.xml` file.

    You should see output that starts with:

```
run:
   [java] username: Jen
   [java] password: **********
   [java] URL: t3://localhost:7001
   [java] Creating a trader
   [java] java.rmi.AccessException: Security violation: User
Jen has insufficient permission to access method; nested
```

```
exception is:
   [java] java.lang.SecurityException: Security violation:
User Jen has insufficient permission to access method
```

This result occurs because the client application calls an EJB that is stored in the `ejb20_basic_statelessSession.jar` you just secured with a security policy.

Step 9: Restrict Access to the statelessSession EJB

1. Using the navigation tree at the left side of the Administration Console, right-click `ejb20_basic_statelessSession.jar`.
2. From the menu, select the Define Policies and Roles for Individual Beans... option.

   A table listing all the EJBs that are in the JAR file appears (in this case, just the `statelessSession` EJB).

   Note: Recall what this option means: you can create a security policy at the EJB level (meaning the security policy will apply to all methods within the EJB), or a particular method within the EJB.
3. Click the [Define Policies] link for the `statelessSession` EJB.
4. In the Policy Condition list box, highlight: `User Name of the Caller`.

   Note: Do *not* modify the value shown in the Methods drop-down menu. (It should read: `ALL`.)
5. Click Add to open the Users window.
6. In the Enter User Name field, type: `Jen`
7. Click Add, then OK.

   The Users window closes. The Policy Statement list box reads:

   ```
   User Name of the Caller
      Jen
   ```

Note: Recall that by defining this security policy on the `statelessSession` EJB, you are overriding the security policy that has already been defined for the EJB JAR in "Step 7: Create a Security Policy for the statelessSession EJB JAR Using the Global Role." Specifically, you are overriding the inherited policy statement of:

```
Caller is Granted the Role
   QA
```

that is shown in the Inherited Policy Statements list box.

8. Click Apply to save your changes.
9. Repeat steps 8 - 10 in "Step 8: Attempt to Access EJBs Through a Client Application" on page 7-6.

   The output from the client application should be the reverse of what it was. In other words, `Stephanie` should be denied access to the `statelessSession` EJB, and `Jen` should be granted access.

   This result occurs because the client application calls the EJB that you just secured with a security policy.

Step 10: Restrict Access to the create() and buy() EJB Methods

1. Using the navigation tree at the left side of the Administration Console, right-click `ejb20_basic_statelessSession.jar`.
2. From the menu, select the Define Policies and Roles for Individual Beans... option.

   A table listing all the EJBs that are in the JAR file appears (in this case, just the `statelessSession` EJB).

   Note: Recall what this option means: you can create a security policy at the EJB level (meaning the security policy will apply to all methods within the EJB), or a particular method within the EJB.
3. Click the [Define Policies] link for the `statelessSession` EJB.

4. Using the Methods drop-down menu, select the `create() - HOME` method.
5. In the Policy Condition list box, highlight: `Caller is a Member of the Group`.
6. Click Add to open the Groups window.
7. In the Enter Group Name field, type: `Testers`
8. Click Add, then OK.

   The Groups window closes. The Policy Statement list box reads:

   ```
   Caller is a Member of the Group
      Testers
   ```

   Note: Recall that by defining this security policy on the `create()` method, you are overriding the security policy that has already been defined for the `statelessSession` EJB in "Step 9: Restrict Access to the statelessSession EJB." Specifically, you are overriding the inherited policy statement of:

   ```
   User Name of the Caller
      Jen
   ```

   that is shown in the Policy Statement list box when `ALL` is selected from the Methods drop-down menu.
9. Click Apply to save your changes.
10. Repeat steps 4 - 9 to secure the `buy(java.lang.String, int) - REMOTE` method, using the same Policy Statement.
11. Repeat steps 8 - 10 in "Step 8: Attempt to Access EJBs Through a Client Application" on page 7-6.

    The output from the client application should fail at different methods for users `Stephanie` and `Jen`. User `Stephanie` should be denied access at the `sell()` method because this method comes after the `create()` and `buy()` methods in the client application (see Listing 7-2 for sample output). User `Jen` should be denied access at the `create()` method (see Listing 7-3 for sample output).

Listing 7-2 Output for User Stephanie: Access Denied at sell() Method

```
Buildfile: build.xml
```

```
run:
   [java] username: Stephanie
   [java] password: **********
   [java] URL: t3://localhost:7001
   [java] Creating a trader
   [java] Buying 100 shares of BEAS.
   [java] Buying 200 shares of MSFT.
   [java] Buying 300 shares of AMZN.
   [java] Buying 400 shares of HWP.
   [java] Selling 100 shares of BEAS.

   [java] java.rmi.AccessException: Security Violation: User:
'Stephanie' has insufficient permission to access EJB: type=<ejb>,
application=_appsdir_ejb20_basic_statelessSession_ear,
module=ejb20_basic_statelessSession.jar, ejb=statelessSession,
method=sell, methodInterface=Remote,
signature={java.lang.String,int}.
```

Listing 7-3 Output for User Jen: Access Denied at create() Method

```
Buildfile: build.xml

run:
   [java] username: Jen
   [java] password: **********
   [java] URL: t3://localhost:7001
   [java] Creating a trader

   [java] java.rmi.AccessException: Security violation: User Jen
has insufficient permission to access method; nested exception is:
      [java] java.lang.SecurityException: Security violation: User
Jen has insufficient permission to access method
```

These results occur because the client application calls the EJB methods that you just secured with security policies.

8 Examples: Copying and Reinitializing Security Configurations for the basicauth Web Application

In this example, you will copy security configurations for the `basicauth` Web application into the configured Authorization and Role Mapping providers' databases so that you can use the Administration Console for subsequent modifications to security roles and security policies. After you make a modification to a security policy using the Administration Console, you will reinitialize the security configuration for the `basicauth` Web application using the original deployment descriptor. Therefore, this example consists of three main steps:

Note: Prior to reviewing this example, you should read: "Techniques for Securing URL and EJB Resources" on page 2-7; "Prerequisites for Securing URL and EJB Resources" on page 2-9; and "Using the Combined Technique to Secure Your URL and EJB Resources" on page 2-17.

Step 1: Copy Security Configurations for the basicauth Web Application

To copy security configurations for the `basicauth` Web application, follow these steps:

Step 1: Obtain the basicauth Web Application

Note: All instructions provided in this section assume that you are working in a Windows environment.

1. Obtain the "Basic Authentication Sample Web Application" (available under Code Samples: WebLogic Server on the *dev2dev Web site*).
2. Unzip the `basicauth.zip` file to a temporary directory (for example, `C:\basicauth`).

Step 2: Modify the Prerequisite Settings and Deploy the Web Application

Note: All instructions provided in this section assume that you are working in a Windows environment.

1. Set the `fullyDelegateAuthorization` flag equal to `true`, using the instructions in "How to Change the fullyDelegateAuthorization Flag" on page 2-11.

   Notes: Recall what this setting means: you are telling WebLogic Server that you want the WebLogic Security Service to perform security checks on *all* URL (Web) and EJB resources. For more information, see "Understanding the fullyDelegateAuthorization Flag" on page 2-10.

   If the `fullyDelegateAuthorization` flag was already set to `true`, just continue to step 2.

2. From the Windows Start menu, select Programs →BEA WebLogic Platform 7.0 →WebLogic Server 7.0 →Server Tour and Examples →Launch Examples Server to start the server called `examplesServer`.

   The `fullyDelegateAuthorization` flag appears in the console as the `examplesServer` starts, and the BEA WebLogic Server Out-of-the-Box Examples Index Page opens in your browser.

3. Using the navigation tree at the left side of the Administration Console, expand Security, then Realms.

4. Expand the `myrealm` security realm.

5. On the General tab, *uncheck* the Ignore Security Data in Deployment Descriptors check box. (The check box may already be unchecked, as this is the default setting.)

   Note: Recall what this setting means: you are telling WebLogic Server to *copy* security for URL (Web) and EJB resources from the deployment descriptors into the configured Authorization and Role Mapping providers' databases *each time you deploy the resource.* For more information, see "Understanding the Ignore Security Data in Deployment Descriptors Check Box" on page 2-14.

6. Click Apply to save your changes.

7. If you had to set the `fullyDelegateAuthorization` flag to `true` in step 1 (that is, it was **not** already set this way), restart the server. (For help, see "Starting and Stopping WebLogic Servers" in the *WebLogic Server Administration Guide*.)

   If you did not have to modify the value of the `fullyDelegateAuthorization` flag in step 1, continue to step 8 **without restarting the server**.

8. Using the Administration Console, deploy the `basicauth` Web application and target it to the `examplesServer`.

   Note: For instructions about how to deploy Web applications, see "Deploying J2EE Applications with the Administration Console" in *Developing WebLogic Server Applications*.

Step 3: Verify the Copied Security Policies (Optional)

1. Open the `web.xml` deployment descriptor for the `basicauth` Web application, and record the content of any `<url-pattern>` and `<http-method>` elements, as well as any `<role-name>` subelements of the `<auth-constraint>` element. Listing 8-1 shows the relevant portions of the `web.xml` deployment descriptor file in bold font.

Listing 8-1 The basicauth Web Application web.xml Deployment Descriptor

```
<!DOCTYPE web-app (View Source for full doctype...)>
<web-app>
   <welcome-file-list>
      <welcome-file>welcome.jsp</welcome-file>
   </welcome-file-list>
   <security-constraint>
      <web-resource-collection>
         <web-resource-name>Success</web-resource-name>
         <url-pattern>/welcome.jsp</url-pattern>
         <http-method>GET</http-method>
         <http-method>POST</http-method>
      </web-resource-collection>
      <auth-constraint>
         <role-name>developers</role-name>
      </auth-constraint>
   </security-constraint>
   <login-config>
      <auth-method>BASIC</auth-method>
```

```
      <realm-name>default</realm-name>
   </login-config>
   <security-role>
      <role-name>developers</role-name>
   </security-role>
</web-app>
```

2. Using the navigation tree at the left side of the Administration Console, expand Web applications, then right-click `basicauth`.
3. From the menu, select the Define Policy... option.
4. On the General tab, in the URL Pattern text field, type: `/welcome.jsp`
5. Click the Define Policy... button to proceed.
6. On the Policy Editor page, use the Methods drop-down menu to select the method: `POST`.

   The `Caller is Granted the Role` Policy Condition is highlighted and the Policy Statement list box reads:

   ```
   Caller is Granted the Role

      developers
   ```

7. Using the Methods drop-down menu, select the method: `GET`.

   The `Caller is Granted the Role` Policy Condition is highlighted and the Policy Statement list box reads:

   ```
   Caller is Granted the Role

      developers
   ```

Step 4: Verify the Copied Security Roles (Optional)

1. Open the `weblogic.xml` deployment descriptor for the `basicauth` Web application, and record the content of any `<security-role-assignment>` elements, specifically focusing on the `<role-name>` and `<principal-name>` subelements. Listing 8-2 shows the relevant portions of the `weblogic.xml` deployment descriptor file in bold font.

Listing 8-2 The basicauth Web Application weblogic.xml Deployment Descriptor

```
<!DOCTYPE weblogic-web-app (View Source for full doctype...)>
   <weblogic-web-app>
      <security-role-assignment>
         <role-name>developers</role-name>
         <principal-name>myGroup</principal-name>
      </security-role-assignment>
   </weblogic-web-app>
```

2. Using the navigation tree at the left side of the Administration Console, right-click `basicauth` Web application.
3. From the menu, select the Define Role... option.
4. On the General tab, in the URL Pattern text field, type: `/*`
5. Click the Define Role... button to proceed.

   The Select Roles page displays the scoped role called `developers`.
6. Click the hyperlinked name: `developers`
7. Click the Conditions tab.

   The Role Statement list box contains a Role Statement based on the content of the deployment descriptor's corresponding `<principal-name>` element, which in this case is the group called `myGroup`.

Step 5: Revert the Ignore Security Data in Deployment Descriptors Setting

Caution: You must perform this step. Failure to revert this setting may result in inconsistent security configurations when your URL (Web) resources are redeployed.

1. Using the navigation tree at the left side of the Administration Console, expand Security, then Realms.

2. Click the name of your security realm (for example, `myrealm`).
3. On the General tab, click the Ignore Security Data in Deployment Descriptors check box. (That is, you should be putting a check mark in the box.)

   Note: Recall what this setting means: you are telling WebLogic Server that you will set security for Web application and EJB resources using the Administration Console, not deployment descriptors. For more information, see "Understanding the Ignore Security Data in Deployment Descriptors Check Box" on page 2-14.

4. Click Apply to save your changes.

Step 2: Modify a Security Policy Using the Administration Console

1. Using the navigation tree at the left side of the Administration Console, right-click `basicauth`.
2. From the menu, select the Define Policy... option.
3. On the General tab, in the URL Pattern text field, type: `/welcome.jsp`
4. Click the Define Policy... button to proceed.
5. On the Policy Editor page, use the Methods drop-down menu to select the method: `POST`.
6. In the Policy Condition list box, highlight the `Hours of Access are Between` policy condition.
7. Click Add.
8. Click OK in the Time Constraint window to select the default start and end times.

   The Policy Statement list box reads as follows:

```
Caller is Granted the Role
   developers
and Hours of Access are Between
```

```
08:00:00 and 19:00:00
```

9. Click Apply to save your changes.
10. Using the Methods drop-down menu, select the method: `POST`, and verify that there are two expressions in the Policy Statement list box.

Step 3: Reinitialize Security Configurations for the basicauth Web Application

To reinitialize security configurations for the `basicauth` Web application from its deployment descriptor, follow these steps:

Step 1: Modify the Ignore Security Data in Deployment Descriptors Setting

1. Using the navigation tree at the left side of the Administration Console, expand Security, then Realms.
2. Expand the `myrealm` security realm.
3. On the General tab, *uncheck* the Ignore Security Data in Deployment Descriptors check box.

Note: Recall what this setting means: you are telling WebLogic Server to *copy* security for URL (Web) and EJB resources from the deployment descriptors into the configured Authorization and Role Mapping providers' databases *each time you deploy the resource.* For more information, see "Understanding the Ignore Security Data in Deployment Descriptors Check Box" on page 2-14.

4. Click Apply to save your changes.

Step 2: Redeploy the basicauth Web Application

1. Using the navigation tree at the left side of the Administration Console, expand Deployments, then click Web Applications.
2. Click the `basicauth` Web application.
3. Click the trash can icon that is located in the same row as the `basicauth` Web application.
4. Click Yes, then the Continue link to delete the `basicauth` Web application.

   The deleted Web application or EJB no longer appears in the table.
5. Redeploy the `basicauth` Web application, targeting it to the `examplesServer`.

   Note: For instructions about how to deploy Web applications and EJBs, see "Deployment Tools and Procedures" in *Developing WebLogic Server Applications*.

Step 3: Verify That the Security Configuration Has Been Reinitialized (Optional)

1. Using the navigation tree at the left side of the Administration Console, right-click `basicauth`..
2. From the menu, select the Define Policy... option.
3. On the General tab, in the URL Pattern text field, type: `/welcome.jsp`
4. Click the Define Policy... button to proceed.

5. On the Policy Editor page, use the Methods drop-down menu to select the method: `POST`.

   The Policy Statement list box reads as follows:

   ```
   Caller is Granted the Role
      developers
   ```

Step 4: Revert the Ignore Security Data in Deployment Descriptors Setting

Caution: You must perform this step. Failure to revert this setting may result in inconsistent security configurations when your URL (Web) resources are redeployed.

1. Using the navigation tree at the left side of the Administration Console, expand Security, then Realms.
2. Click the name of your security realm (for example, `myrealm`).
3. On the General tab, click the Ignore Security Data in Deployment Descriptors check box. (That is, you should be putting a check mark in the box.)

   Note: Recall what this setting means: you are telling WebLogic Server that you will set security for Web application and EJB resources using the Administration Console, not deployment descriptors. For more information, see "Understanding the Ignore Security Data in Deployment Descriptors Check Box" on page 2-14.

4. Click Apply to save your changes.

Index

F

G

I

J

M

N

S

U

W