BEA WebLogic Portal®

Security Guide

Version 10.2
Revised: February 2008

Contents

1. Introduction

Part I. Architecture

2. Planning a Security Strategy

Part II. Development

3. Securing Your Portal Deployment

Part III. Staging

6. Managing Security Providers

7. Configuring Delegated Administration

8. Configuring Visitor Entitlements

9. Deploying Security Components

Part IV. Production

10.Implementing Authorization Programmatically

CHAPTER 1

Introduction

This guide describes how to secure your portal applications. Use the security techniques described in this guide to assure the confidentiality, integrity, and availability of your portal applications and application resources.

BEA WebLogic Portal® security leverages BEA WebLogic Server security features. The WebLogic Server, in turn, supports and enhances J2EE security services. Like other J2EE components, the security services are based on standardized, modular components. WebLogic Server implements these Java security service methods according to the standard, and adds extensions that handle many details of application behavior automatically, without requiring additional programming. For more information about WebLogic Server security, see Security for BEA WebLogic Server 10.0.

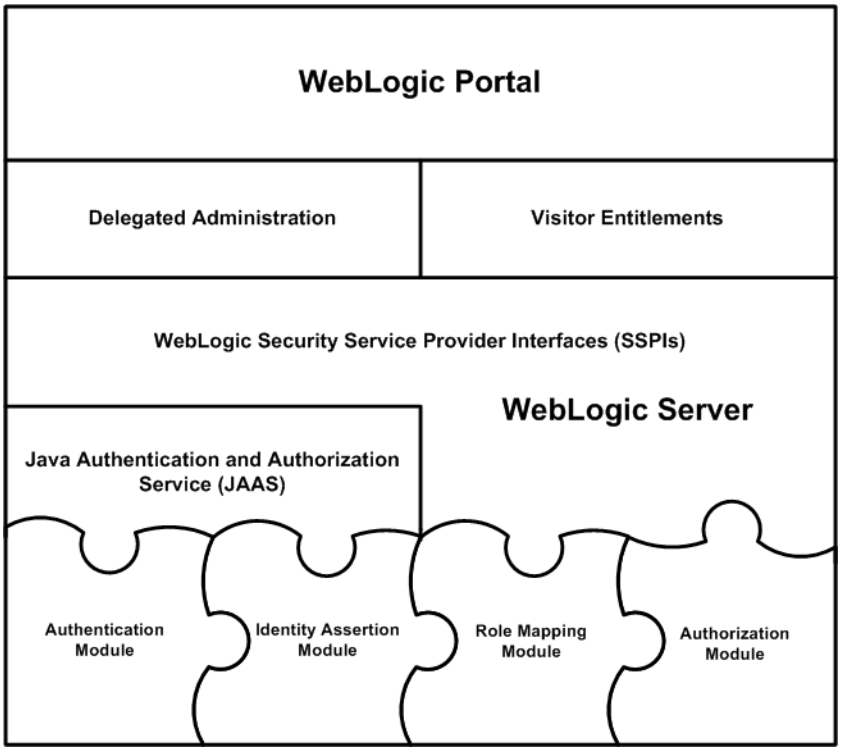
This chapter includes the following sections:

Foundations of WebLogic Portal Security

WebLogic Portal is part of the unified WebLogic security architecture. WebLogic Portal-specific security extensions, including visitor entitlements and delegated administration, use the WebLogic Security Service Provider Interfaces (SSPIs). All authorization checks are performed using the isAccesssAllowed method as part of the shared architecture.

Figure 1-1 shows WebLogic Portal in the unified WebLogic security architecture.

Figure 1-1 WebLogic Portal Security Architecture



This section describes the components of the WebLogic Portal security architecture.

J2EE Security Services

WebLogic Server utilizes the security services of J2EE. Like the other J2EE components, the security services are based on standardized, modular components. WebLogic Server implements these Java security service methods according to the standard, and adds extensions that handle many details of application behavior automatically, without requiring additional programming.You can use the security providers that the WebLogic Server supplies as defaults, or you can integrate custom providers.

WebLogic Security Service Provider Interfaces

WebLogic Portal uses the WebLogic Security Framework, which unifies security enforcement and presents security as a service to other WebLogic Server components. The WebLogic Security Framework calls methods in the WebLogic Security Service Provider Interfaces (SSPIs) to perform security operations. The WebLogic SSPIs are set of WebLogic packages that enable security providers to be developed and integrated with the WebLogic Server Security Service.

These interfaces are implemented by the WebLogic security providers and custom security providers. Security providers are software modules that can be plugged into a WebLogic Server security realm to offer security services, such as authentication, authorization, and auditing, to applications. Custom security providers (written by third-party security vendors or security developers) are implementations of the SSPIs and are not supplied with WebLogic Server.

Java Authentication and Authorization Service

WebLogic Server uses Java Authentication and Authorization Service (JAAS) classes to reliably and securely authenticate clients. JAAS is a standard extension to the security in the Java Software Development Kit version 1.4.1. WebLogic Server clients, including WebLogic Portal, use only the authentication portion of JAAS.

JAAS implements a Java version of the Pluggable Authentication Module (PAM) framework, which permits applications to remain independent from underlying authentication technologies. This enables you to use new or updated authentication technologies without making modifications to your application.

The JAAS LoginContext provides support for the ordered execution of all configured authentication providers and is responsible for the management of each configured provider. For more information, see JAAS and WebLogic Server.

Authentication

Authentication is the verification of identity, answering the question: Is the user who they say they are?

Authentication is typically performed using a login process, during which the user supplies a *credential*, such as a username and password combination. Once the user is authenticated, a set of *identities* (such as the username and the user's group membership) is associated with the user. These identities are also referred to as *principals*.

This section includes these topics:

- Authentication Providers
- Identity Assertion Providers and Single Sign-On
- Implementing Authentication Programmatically

Authentication Providers

Authentication providers verify the identity of users. Authentication providers also remember, transport, and make available that identity information to various components of a system through *subjects* (containers for authentication information) when needed. During the authentication process, a *principal validation provider* provides additional security protections for the principals contained in the subject by signing and verifying the authenticity of those principals.

WebLogic Server supplies its own authentication providers, which can access user and group data in its embedded LDAP server, external LDAP stores, and external Relational Database Management Systems (RDBMSs). While WebLogic Server uses the embedded LDAP server as its authentication provider by default, WebLogic Portal uses the SQL Authenticator by default.

Tip: Previous releases of WebLogic Portal included a WebLogic Portal-specific RDBMS Authenticator. This has been deprecated. If you are upgrading from a previous release, you can choose whether to upgrade to the WebLogic SQL Authenticator as your default authentication provider or continue to use your existing user store. For information about upgrading your user store, see the Upgrade Guide.

WebLogic Portal uses WebLogic Server for authentication, whether you are using one or more WebLogic authentication providers, one or more custom authentication providers configured for use with WebLogic Server, or a combination of WebLogic and custom providers.

Tip: WebLogic Server supplies RDBMS authentication providers including the SQL Authenticator, Read-only SQL Authenticator, and Custom RDBMS Authenticator. WebLogic Server also supplies authentication providers for external LDAP servers including Open LDAP, Sun iPlanet, Microsoft Active Directory, and Novell NDS LDAP servers, as well as its embedded LDAP server.

Each authentication provider requires a JAAS LoginModule. LoginModules are responsible for authenticating users within the policy domain and for populating a subject with the necessary principals (users and groups). LoginModules that are not used for perimeter authentication also verify the proof material submitted (for example, a password). For more information on creating LoginModules for a custom authentication provider, see Login Modules.

Identity Assertion Providers and Single Sign-On

An *identity assertion provider* is a specific type of authentication provider that allows users or system processes to assert their identity using *tokens* (representations of security-related information). Identity assertion providers enable perimeter authentication and support *single sign-on* (SSO). For example, an identity assertion provider can generate a token from a digital certificate, and that token can be passed around the system so that users are not asked to sign on more than once.

Note: You can use an identity assertion provider in place of an authentication provider if you create a LoginModule for the identity assertion provider, or in addition to an authentication provider if you want to use the authentication provider LoginModule.

Web applications often consist of many different components, each of which may have its own authentication scheme or user registry. SSO enables users of these applications to authenticate only once to obtain access to all components of the application. Once users are authenticated in one site that participates in a SSO configuration, they are automatically logged into the other sites in the SSO configuration.

The WebLogic Server SAML Identity Asserter can create Security Assertion Markup Language (SAML) assertions, which allow access to remote portlets in the federated portal architecture. For more information, see the Federated Portal Guide.

Tip: The WSRP Identity Asserter, which provided this functionality in previous releases, is still available for backward compatibility.

Implementing Authentication Programmatically

You can implement authentication programmatically using the appropriate WLP APIs. For detailed information and best practices, see "Implementing Authentication Programmatically" on page 3-9.

Authorization

Authorization is the control of access to resources, answering the question: Does the user have access to this protected resource? Interactions between users and resources are controlled based on user identity or other information.

In WebLogic Portal (as in WebLogic Server), *roles* control access to portal resources, J2EE resources, and administrative tools, so users can access only the resources and tools that their assigned roles allow. WebLogic Portal uses WebLogic Server roles to enable you to dynamically match users to roles at login.

Authorization Providers

An authorization provider controls the interactions between users and resources to ensure confidentiality. Like a LoginModule for an authentication provider, an *Access Decision* is the component of an authorization provider that determines if access to a resource is allowed. Specifically, an Access Decision determines whether a subject has permission to perform the specified action on a WebLogic resource. A runtime call to the isAccessAllowed method makes this determination, based on the subject's security roles.

It returns one of the following values:

- `PERMIT`—Indicates that the requested access is permitted
- `DENY`—Indicates that the requested access is explicitly denied
- `ABSTAIN`—Indicates that no explicit decision was rendered

Role Mapping Providers

Role mapping is the process by which principals (users or groups) are dynamically mapped to security roles at runtime. A *role mapping provider* determines which security roles apply to the subject when the subject is attempting to perform an operation on a resource. Because this operation usually involves gaining access to the resource, role mapping providers are typically used with authorization providers.

The default role mapping provider is the WebLogic XACML role mapping provider, which stores role policies separately in the embedded LDAP server. The WebLogic XACML role mapping provider is required for WebLogic Portal, and suitable for most needs. It is unlikely that you will need to configure a custom role mapping provider.

Roles and Role Policies

A security *role* is a privilege granted to users or groups based on specific conditions. Roles are used to determine whether to grant or deny access to resources, and to determine which capabilities on those resources are available to the user. Granting a role to a user or group confers the defined access privileges to that user or group, as long as the user or group is granted the role. Any number of users or groups can be granted a single role.

Roles are computed and granted to users or groups dynamically, based on *role policies*, which consist of a role name and a role definition. Role policies are dynamic, and can be based on username, group membership, user profile property values, session and request attributes, and date and time functions.

Roles 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). For more information on role scoping, see "Understanding Global and Scoped Roles" on page 2-7.

Security Policies

Security policies answer the question: Who has access to a WebLogic resource? A security policy is created when you define an association between a WebLogic resource and one or more users, groups, or roles. Hence, a role policy defines a role and a security policy defines an authorization constraint associated with that role.

BEA recommends basing security policies on roles rather than users or groups. Basing security policies on roles enables you to manage access based on a role that a user or group is granted, which is a more flexible method of management.

Security policies are kept in the authorization provider's store, which is the embedded LDAP server by default. For a list of the default security policies for the WebLogic resources, see Default Root Level Security Policies.

If a security policy is based on a user or group, the user or group must be defined in the user store for the authentication provider that is configured in the default security realm. If a security policy

is based on a role, the role must be defined in the store for the role mapping provider that is configured in the default security realm.

Deployment Descriptors

The J2EE platform defines a means of establishing security contracts in a document known as the deployment descriptor. Deployment descriptors restrict access to resources that can be accessed directly using a URL. You must use deployment descriptors to secure portlet resources including JSPs and page flows, which could otherwise be accessed directly by anyone that knows the URL to those resources. Roles defined in deployment descriptors are based on users and groups.

For information on securing portal application resources using deployment descriptors, see Chapter 4, "Preventing Direct Access to Portal Application Resources".

Portal application resources can also be protected based on runtime constraints, such as the time of day and user profile property values. For information on setting up these runtime constraints using role-based security policies, see "Setting Up Role-Based Authorization" on page 2-6.

WebLogic Portal-Specific Security Extensions

This section provides a high level overview of the WebLogic Portal-specific extensions to role-based access control. You can control access to portal resources for two categories of users:

- Portal visitors, whose access to portals and portal resources you control using *visitor entitlements*. Visitor access is determined based on visitor entitlement roles.
- Portal administrators, whose capabilities to manage portal resources you control using *delegated administration*. Administrative access is determined based on delegated administration roles.

For more information on role-based access control, see "Setting Up Role-Based Authorization" on page 2-6.

Visitor Entitlements

Use visitor entitlements to determine who may access the resources in a portal application and what they may do with those resources. Access is based on the role assigned to a portal visitor, allowing for flexible management of portal resources.

For example, if you have an Employee Review portlet that only managers should be able to access, you can create a `Managers` visitor entitlement role, and set entitlements so that only users who are assigned that role can view the portlet.

A portal visitor is assigned a role based on their username, group membership, user profile property values, session and request attributes, and date and time functions. For example, the `Gold Member` role could be assigned to visitors who are part of the frequent flyer program and have flown more than 50,000 miles in the previous year. Roles are assigned to visitors dynamically when they log in to the site. As this example highlights, visitor entitlements also enable personalization.

Note: If no visitor entitlement roles exist, the default behavior is to *allow* access to the portal and portal resources to all visitors. Content management entitlements are an exception to this policy. If there are no entitlements set on content management components, then those components are not accessible to visitors.

Delegated Administration

Delegated administration provides secure administrative access to the WebLogic Portal Administration Console. Using the WebLogic Portal Administration Console, administrators can create and manage portals, desktops, shells, books, pages, layouts, look and feels, and portlets. A delegated administration role is a classification of a portal administrator based on their username, group membership, user profile property values, session and request attributes, and date and time functions.

In your organization, you might want different administrators to have different access privileges to various administration tasks and resources. You can use delegated administration to propagate WebLogic Portal Administration Console access privileges within a hierarchy of roles.

For example, a system administrator might have access to every feature in the WebLogic Portal Administration Console. The system administrator might then create a portal administrator role for administrators who are allowed to manage instances of portal resources in specific desktop views of your portal. The system administrator might also create a library administrator role for administrators who are allowed to manage your portal resource library.

Note: WebLogic Portal has one predefined delegated administration role, `PortalSystemDelegator`. By default, all members of the `Administrators` group are assigned the `PortalSystemDelegator` role. Anyone assigned the `PortalSystemDelegator` role has unlimited access to administrative tasks anywhere in the enterprise portal application. Other delegated administration roles only have access to resources if that access has been explicitly granted.

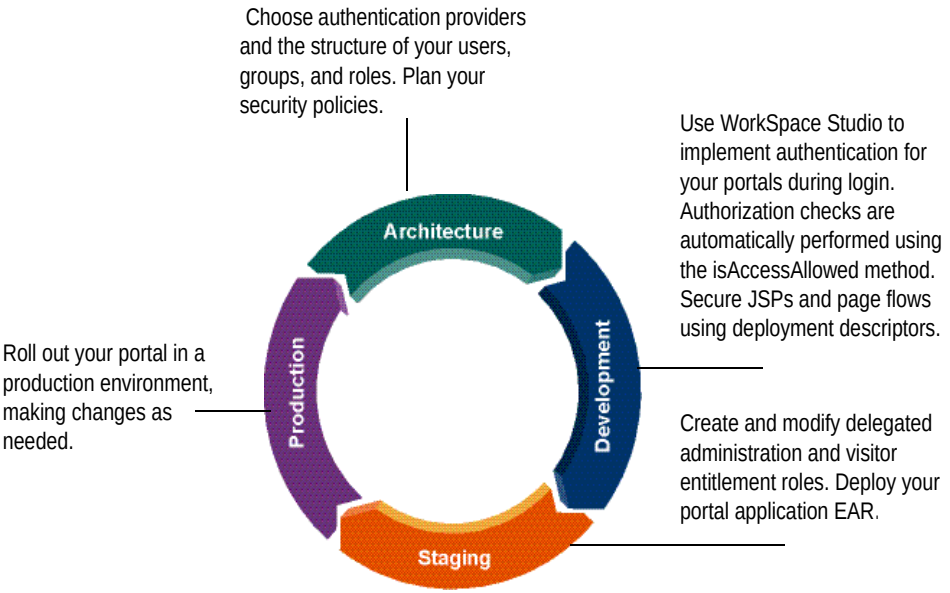
Delegated administration roles are mapped to administrative functions on portal resources using security policies. Given the appropriate privileges, administrators can delegate both the privilege to administer a given resource capability and the privilege for the delegatee to delegate further.

Security Features in the WebLogic Portal Life Cycle

For more information about the portal life cycle, see the WebLogic Portal Overview.

Figure 1-2 shows how security features fit into the portal life cycle.

Figure 1-2 How Security Features Fit into the Four Phases of the WebLogic Portal Life Cycle



The life cycle contains the following four phases:

- Architecture
- Development
- Staging
- Production

Architecture

During the architecture phase, you plan how to organize security policies and roles, and how that fits into your overall security strategy. You determine which WebLogic or custom authentication providers to use. You also determine how to secure your application using visitor entitlements and your application resources using delegated administration and deployment descriptors.

Development

In the development phase, you can add log in and log out functionality to your portal applications.

You can also use the isUserInRole method (or tag) to verify role membership for a specific user within your application logic. This type of programmatic security, coded into portlets, books, and pages, enables you to customize a user's path through a portal and perform fine-grained entitlement-checking.

You can use the isAccessAllowed method (or tag) to allow or deny access to a specific application resource within your application logic. This is the same method that is automatically called by the runtime framework to determine access to portal resources.

Staging

During the staging phase you can test the functionality you created in the development phase. You also set up your roles and roles and security policies. Development and staging phases often occur simultaneously. You might move iteratively between these two phases, developing and then testing what you created. If you return to the development phase and make changes, you must redeploy your portal application to see the changes in the staging phase. If you plan to use more than one development team to create your portal, determine how you will merge and assemble the code.

Production

After you test your portal application in the staging phase, use the production phase to perfect your production environment. For example, in the production phase you might use the WebLogic Portal Administration Console to make changes to your portal application, such as changing authentication providers or making small changes in your roles or policies.

Credential Vault

The Credential Vault is a safe storage mechanism that saves user credentials for use in any remote application. A user would enter his or her user name and password only once in a lifetime in portal. From then on whenever the user accesses a portal page that needs this information, the portlet will automatically obtain their credentials and pass them to the remote application without requiring the user to logon and enter their information again.

For more information, see "Securing Third-Party Applications" on page 5-1.

Getting Started

If you are new to portal development, see the WebLogic Portal Overview for more information about the portal life cycle.

For more information about users, groups, and user profiles, see the User Management Guide.

For more information about WebLogic Server security, see Security for BEA WebLogic Server 10.0.

Part I Architecture

This section contains guidelines to help you plan your security infrastructure.

Part I includes the following chapter:

- Chapter 2, "Planning a Security Strategy"

During the planning stage, you determine how many and which authentication providers to use. You also determine how to set up your roles and how that relates to your user, group, and user profile strategy. You also start creating visitor entitlement roles and a hierarchy of delegated administration roles. Developing a security strategy can save you time during the other phases of the portal life cycle.

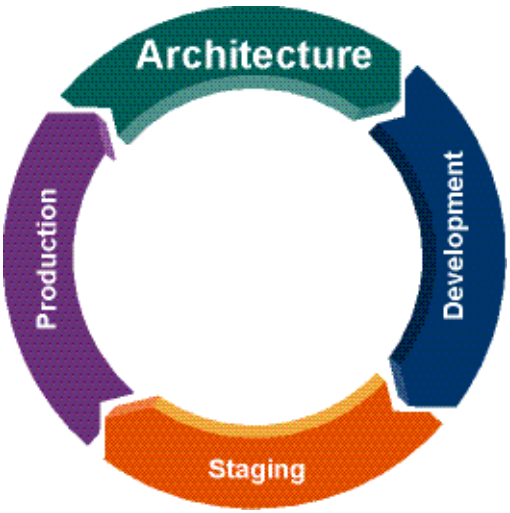
You can access existing user stores using the authentication providers WebLogic Server provides, or by developing and configuring one of your own. WebLogic Server provides RDBMS authentication providers including the SQL Authenticator, Read-only SQL Authenticator, and Custom RDBMS Authenticator. WebLogic Server also provides authenticators for its embedded LDAP server and external LDAP servers including Open LDAP, Sun iPlanet, Microsoft Active Directory, and Novell NDS LDAP servers.

The default authentication provider for WebLogic Portal is the SQL Authenticator, but you can use only a custom authentication provider, or a combination of WebLogic and custom providers.

You can create roles based on existing groups and create additional roles depending on how you want to administer your portal applications and what you want visitors to your portal applications to be able to view and modify.

The authentication and authorization strategies that you implement in a test or staging environment might be different from those on a production system.

For a description of the architecture phase of the portal life cycle, see the WebLogic Portal Overview. The portal life cycle is shown in the following graphic:



CHAPTER 2

Planning a Security Strategy

Developing a plan to implement security features can save you time during later phases in the portal life cycle.

This chapter includes the following sections:

- Developing Your Security Strategy
- Choosing WebLogic and Custom Authentication Providers
- Setting Up Role-Based Authorization
- Designing Security for Optimal Performance

Developing Your Security Strategy

Use the following guidelines to plan your security strategy:

1. Determine which authentication providers to use.

   You can use multiple WebLogic and custom authentication providers. If a security realm has multiple authentication providers configured, JAAS control flags determine the order of execution of the authentication providers, as described in Setting the JAAS Control Flag Option. For information on choosing authentication providers, see "Choosing WebLogic and Custom Authentication Providers" on page 2-3. For information on configuring authentication providers in the WebLogic Portal Administration Console, see Chapter 6, "Managing Security Providers".

2. Determine the structure of your users and groups.

If the user store you are accessing already has users organized into groups, you can build a hierarchical tree in the WebLogic Portal Administration Console that matches the existing group structure. If you are using the default WebLogic authentication provider, you do not need to build a group hierarchy tree.

If you are creating new groups, you can structure your users into groups and subgroups that look like your organization, which makes it easier to manage users. A group can be a related collection of users, such as a department, team, or regional office. A user can belong to more than one group, and groups can belong to other groups. For more information on users and groups, see the User Management Guide.

After you determine your group structure, you can create, modify, and delete groups using WorkSpace Studio or WebLogic Portal Administration Console.

You can later associate groups with delegated administration roles, to control administrator access to portal resources, and visitor entitlement roles, to control the visitor experience in your portal applications.

3. Determine if you want to collect and store information about users (user profile properties) and if you want to be able to edit those properties.

   You can use user profile information to target users with personalized content, e-mails, pre-populated forms, and discounts based on the rules you set up. For more information, see the User Management Guide.

4. Create a unified user profile (UUP) to retrieve additional user profile information from other user stores to define rules for personalization, delegated administration, and visitor entitlement.

   For more information, see the User Management Guide.

5. Create delegated administration and visitor entitlement roles using the WebLogic Portal Administration Console.

   When you create a role policy for delegated administration and visitor entitlement roles, you can associate users and groups with the role. You can also create a regular expression that enables you to specify date, time, user, and HTTP session and request characteristics for the role. For high level tasks associated with creating delegated administration and visitor entitlement roles, see “Setting Up Role-Based Authorization” on page 2-6.

6. Create security policies to protect specific portal resources or types of portal resources. You do this by associating a role policy with the resource or resource type.

   BEA recommends basing security policies on roles rather than users or groups. Basing security policies on roles enables you to manage access based on a role that a user or group is granted, which is a more flexible method of management. Your user and group structure

is usually static (especially in organizations where control of the user store is centralized), while the roles that users and groups belong to can be more dynamic.

For information on creating security policies for portal administrators, see Chapter 7, "Configuring Delegated Administration". For information on creating security policies for portal visitors, see Chapter 8, "Configuring Visitor Entitlements". For information on available authentication techniques you can use to perform access checks at runtime in your portal applications (for example to determine if a user is assigned a specific role) see Chapter 10, "Implementing Authorization Programmatically".

Choosing WebLogic and Custom Authentication Providers

The default authentication provider for WebLogic Portal is the WebLogic SQL Authenticator, which has a PointBase RDBMS as its demonstration user store. You can use another RDBMS, another WebLogic provider (such as the WebLogic LDAP provider), or you can use a custom authentication provider not supplied by WebLogic Server. Once you connect a custom authentication provider to WebLogic Server, the users in the custom provider user store can log into your portal.

Note: Do not use the WebLogic LDAP provider if you have a large numbers of users and groups. The WebLogic LDAP user store is only sufficient for storing a small number of users and groups.

Setting Up a WebLogic Authentication Provider

The default authentication provider for WebLogic Portal is the WebLogic SQL Authenticator. During installation, demonstration users accounts and data are created in the default PointBase RDBMS that ships with the WebLogic SQL Authenticator.

In most production environments, you must configure the SQL Authenticator to use your own RDBMS user store rather than the PointBase demonstration RDBMS. Scripts are available in *WEBLOGIC_HOME*/common/p13n/db/*rdbms_name* to configure each supported RDBMS as a user store.

Previous releases of WebLogic Portal included a WebLogic Portal-specific RDBMS Authenticator. This has been deprecated. If you are upgrading from a previous release, you can choose whether to change your default authentication provider using the WebLogic Upgrade Wizard during the domain upgrade. For information on how to upgrade your user store manually or using the WebLogic Upgrade Wizard, see the Upgrade Guide.

You can also use the WebLogic LDAP Authenticator, which is the default authentication provider for WebLogic Server, as a user store. Due to performance limitations, only use the LDAP Authenticator if you have a small number of users and groups, or if you are in a staging, rather than production, environment. For information on configuring the WebLogic LDAP provider, see Managing the Embedded LDAP Server.

Tip: The default role mapping provider is the WebLogic XACML role mapping provider, which stores role policies separately in the embedded WebLogic LDAP server. The WebLogic XACML role mapping provider is required for WebLogic Portal, and suitable for most needs. It is unlikely that you will need to configure a custom role mapping provider.

Setting Up a Custom Authentication Provider

If you are using a custom authentication provider, you can connect that provider to WebLogic Server (assuming it is a supported type), so that the users in that custom provider can log in to your portal applications. This user store can contain users, passwords, and user properties; it may or may not contain group information.

Develop custom authentication providers according to the guidelines in How to Develop a Custom Authentication Provider. You can add users to a custom authentication provider using WorkSpace Studio, the WebLogic Portal Administration Console, or by adding a user directly through the authentication provider. For more information, see the User Management Guide.

The users in that custom provider can then log in to your portals.

Deciding When to Use Multiple Authentication Providers

In addition to the default WebLogic SQL Authenticator, WebLogic Portal supports the use of multiple authentication providers, once they have been configured to work with WebLogic Server. If you have users and groups stored in more than one custom authentication provider, you may want to access the user stores for each of these providers.

Using multiple authentication providers, you can:

- Allow users in each authentication provider to log into your portal applications.
- Provide personalized applications to users in each authentication provider. You can define personalization rules based on the roles the users or groups have been assigned.

- Entitle users in each authentication provider to visitor access to portal application resources based on the visitor entitlement roles the users or groups have been assigned.
- Give users in each authentication provider administrative access to portal application resources. You can define delegated administration roles based on the users or groups have been assigned.

Notes: If your external user store contains additional properties for users and groups (for example, e-mail and phone numbers), accessing those properties involves separate development steps to create a UUP. For more information, see the User Management Guide.

It is possible (but not recommended) to store an identical username or group name in more than one authentication provider. For example, user `foo` can reside in the embedded LDAP server and in an RDBMS user store. In that case, WebLogic Portal uses only one user profile for user `foo`.

If you are using an RDBMS user store, be aware of case sensitivity when entering names for users and groups. For example, user `Bob` is different from user `bob`.

Setting Up Multiple Authentication Providers

The following high-level steps guide you through the process of setting up multiple authentication providers for use with WebLogic Portal. Perform these steps for each authentication provider:

1. If you are using any custom authentication providers, develop them according to the guidelines in How to Develop a Custom Authentication Provider.
2. Configure each authentication provider to be used with WebLogic Server. For information on how to modify configuration settings in the WebLogic Server Administration Console, see Configuring Authentication Providers. This section also lists the authentication providers included with WebLogic Server.
3. In the WebLogic Server Administration Console, connect to each authentication provider.

   Each authentication provider you connect to WebLogic Server also becomes visible in drop-down lists in WorkSpace Studio and the WebLogic Portal Administration Console, as shown in Figure 2-1.

Figure 2-1 Choose the Authentication Provider to Use

4. You can view and administer external users and groups in the WebLogic Portal Administration Console or by accessing the external user store directly.

Selecting Read-Only or Write Access to User Information

If you want to be able to edit information about users and groups in an external user store, you must develop your custom authentication provider to allow write access. If the provider is writable, you can create and modify users, groups, and user profiles that are stored in that provider. For instructions on setting up custom authentication providers and making them writable, see Configuring Security Providers.

The default configuration for supported custom authentication providers is read-only access to users and groups from the WebLogic Portal Administration Console and the WebLogic Server Administration Console. If the authentication provider is read-only, you can view the users, groups, and user profiles that are stored in that provider, but you cannot change them.

The WebLogic SQL Authenticator and the WebLogic LDAP Authenticator provide write access by default.

Setting Up Role-Based Authorization

A security role is a privilege granted to users or groups based on specific conditions. Roles are the basis for access control in WebLogic Portal. Administrators can create roles, assign users and groups to roles, and define access control to protected resources based on roles.

Roles can sometimes be mapped directly to groups. The difference between groups and roles is that group membership is statically assigned by a server administrator, while role membership is dynamically determined based on information including the username, group membership, user profile property values, session and request attributes, and date and time functions. Roles can also be scoped to specific WebLogic resources within a single application in a WebLogic Server domain, while groups are always scoped to an entire WebLogic Server domain.

A role policy consists of a role name and role definition. Once you create a role policy, you define an association between the role and a WebLogic resource. This association, called a security policy, specifies who has what access to the WebLogic resource. The default role mapping provider is the WebLogic XACML provider, which uses the embedded LDAP server to store role policies.

Note: The WebLogic XACML role mapping provider is required for WebLogic Portal, and suitable for most needs. It is unlikely that you will need to configure a custom role mapping provider.

Understanding Global and Scoped Roles

Roles can be scoped to any resource in an application. A role that applies to all resources deployed within a security realm (the entire WebLogic Server domain) is called a *global role*. A role that applies to a specific instance of a resource deployed in a security realm (such as a portal-specific resource) is called a *scoped role*. You can combine role types when creating security policies for WebLogic resources.

Global Roles

Global roles are roles that apply to all WebLogic resources in a security realm. You administer global roles using the WebLogic Server Administration Console.

WebLogic Server automatically creates certain default global roles, with default privileges and group assignments for each role. When a user is added to one of these groups, that user is automatically granted the global role. All the global roles defined in the security realm are shown in the WebLogic Server Administration Console **Global Roles** page. For more information, see Default Global Roles.

Scoped Roles

Scoped roles apply to a specific WebLogic resource. Scoped roles can be used for visitor entitlements and delegated administration.

Delegated administration roles are enterprise-application scoped. This is because it is desirable to have one administration role for all the web applications (each of which contains one or more portals). You must carefully consider your security requirements when setting up delegated administration roles.

Visitor entitlement roles on portal resources can be web-application scoped or enterprise-application scoped. If each web application has different requirements for constraints on visitor access, you should typically use web-application scoped roles. However, if you want to use the same roles in multiple web applications within an enterprise application, you can use enterprise-application scoped roles.

Using the WebLogic Portal Administration Console, you can configure a scoped role for portal resources in the resource library or in a specific desktop.

To protect *all instances* of a specific book, page, or portlet, or all books, pages, or portlets, set the security policies for the resource or resource type in the portal resource library. The library contains the master versions of all portal resources, and the security policies set in the library apply to a resource wherever it appears in the desktop (Portals node).

You can also set security policies for a desktop resource, such as a book, page, or portlet, in the desktop. If you set security policies for a resource in a desktop (and not in the resource library), they are only enforced for that instance in that desktop. You can also set security policies on an entire desktop.

Note: If you set a security policy on a specific resource in a desktop (for example, a portlet) it applies only to that instance of the resource. Therefore, if you do not secure a resource within the resource library, you must secure each instance of the resource, wherever it appears in the hierarchy of books and pages in the desktop.

Restricting Portal Visitor Access Using Entitlements

Use visitor entitlement roles to enable customization and personalization in your portal applications, as well as to control visitor access to your portal. Visitor entitlements control visitor access to portal resources, groups, and content management resources.

Visitor entitlement roles dynamically determine what access privileges a portal visitor has based on username, group membership, profile, session and request attributes, and date and time functions. Security policies determine what capabilities for a given resource are available to a given role. Entitlement capabilities differ by resource, as described in Chapter 8, “Configuring Visitor Entitlements”.

Tip: Resources without entitlements placed on them are available to everyone.

Visitor access is based on the visitor entitlement roles you create in the WebLogic Portal Administration Console.

Protecting Portal Resources Using Visitor Entitlements

You can create entitlements to control visitor access to the following types of portal resources:

- Library
- Desktops
- Books
- Pages
- Communities
- Templates

- Look and feels
- Portlet categories
- Portlets

Depending on the resource, you can set visitor entitlements in the resource library or the desktop. Within a given desktop, you can entitle specific resources in the desktop such as pages, books, and portlets. Within the library, you can entitle specific books, pages, and portlets, or all resources in one of these categories.

Visitor entitlements in the resource library apply to all instances of the resource in portal applications. However, they do not bar you from setting more local policies in the desktop.

When a request is made by a portal visitor for access to a resource, access checks are performed in the following order:

1. Security policies for that resource in the desktop are enforced.
2. Security policies for that resource in the resource library are enforced.
3. Security policies for that resource type in the library are enforced.

Note: If you only set protection for resources in the desktop, they are only enforced in the desktop. To protect the resource globally, you must set entitlements on the resource in the library.

To configure visitor entitlements on portal resources:

1. Make sure users and groups are configured in your user store. For information about creating users and managing user profiles and groups, see the User Management Guide.
2. Create a visitor entitlement role if one with the appropriate settings does not exist, as described in "Creating Visitor Entitlement Roles" on page 8-2.
3. Assign users, groups, or conditions to the visitor entitlement role, as described in "Adding Users, Groups, and Conditions in Visitor Entitlement Roles" on page 8-3.
4. Create security policies to determine what capabilities your role has for a given portal resource, as described in "Setting Visitor Entitlements on Portal Resources in the Library" on page 8-14 and "Setting Visitor Entitlements on Portal Resources in the Desktop" on page 8-17.

For information on creating, viewing, and modifying visitor entitlement roles, see Chapter 8, "Configuring Visitor Entitlements".

Protecting Content Management Resources Using Visitor Entitlements

You can create entitlements to control access to the following types of content management resources:

- Repositories
- Content
- Content types
- Workflows

Each has visitor capabilities that are based on the type of resource.

Protecting Groups Using Visitor Entitlements

GroupSpace and other community creators and owners can invite others to join the Community. Visitor entitlements determine whether a creator or owner can view potential members using the Browse options when selecting who to invite. For more information on GroupSpace and how to use invitations in GroupSpace, see the GroupSpace Guide.

The only visitor capability for groups is View access to the group, which determines whether the community owner or creator can see the group and the users in the group.

Setting Up a Delegated Administration Role Hierarchy

Delegated administration roles allow you to determine what actions various administrators can take using administrative tools on different resources. Delegated administration roles are mapped to administrative functions using security policies. You can create and maintain separate administration capabilities depending on your organizational needs. For example, you can create separate administrators for your company's Human Resources and Accounts Payable departments. The portal resources associated with each department can only be managed by the administrators you specify.

WebLogic Portal includes a default system administrator, `PortalSystemDelegator`, which is the role where the hierarchical delegation starts. By default, all members of the `Administrators` group are assigned the `PortalSystemDelegator` role. Anyone assigned the `PortalSystemDelegator` role has unlimited access to administrative tasks anywhere in the enterprise portal application. You can create as many different administrators as you need by creating administrator roles and then assigning specific users, groups, or conditions to those roles.

You can use delegated administration to propagate access privileges within a hierarchy of roles that define the structure for delegated administration. You have flexibility in the way you set up your administration hierarchy and assign privileges to your administrators. You can create different levels of administrators, each with varying degrees of access, as described in "Example Role Hierarchy" on page 2-11. You can also create administrators that can, in turn, delegate administration tasks to other administrators.

Notes: If you are using more than one authentication provider, you can have a group in one provider with a name that is the same as a group in another provider. When you set delegated administration on a group, an administrator in that delegated administration role can manage that group in all providers that contain that group (if the administrator has delegated administration privileges for all the providers).

Example Role Hierarchy

A role hierarchy enables you to keep a tight control on how delegation happens and who can delegate to whom. In the WebLogic Portal Administration Console, one administrator (mapping to a role) might want to create sub-roles for other administrators with limited administration privileges.

A child role has a subordinate relationship to its parent role. A role can delegate only to its sub-roles, thus restricting delegated administration. The following is an example hierarchy.

RoleA can delegate to:

- Role1
- Role2
- Role3 can delegate to:
  - Role 3.1
  - Role 3.2

RoleB can delegate to:

- Role4
- Role5
- Role6

The user assigned RoleA cannot delegate to the sub-roles of RoleB as a peer role. RoleA can delegate to any of its descendants.

Child roles do not inherit the privileges of the parent role. If you delete a child role, you are removing it from the system.

Note: Child roles within a delegated administration role must be uniquely named. For example, you cannot have a delegated administration role called RoleA with a child role called RoleB if you already have a child role called RoleB elsewhere in the hierarchy.

Setting Up Administrative Roles

You can create delegated administration roles at any time after you have determined your role strategy:

- Before the portals are assembled
- While assembling and designing a portal in an interactively
- After the portal has been assembled

The following steps ensure that your administrators are set up correctly:

1. Model your delegated administration hierarchy to fit the needs of your organization.
2. Create a role for each administrator type.
3. Define role policies in any of the following ways:
   - Add one or more users to a role
   - Add one or more groups to a role
   - Add users using expressions (user profile properties, session and request attributes, and date and time functions)
4. Define security policies by assigning delegated administration privileges to various resources, including:
   - Groups
   - Portal resources
   - Interaction management objects
   - Content management objects
   - Authentication providers
   - Visitor entitlements

For more information, see Chapter 7, “Configuring Delegated Administration.”

Designing Security for Optimal Performance

Plan and create your groups according to the roles you will create. If you do not consider roles in your group planning, you might have to modify your groups to accommodate roles. For more information on groups, see the User Management Guide.

For more efficient management, grant roles to groups rather than to individual users.

Note: Do not use the WebLogic LDAP provider to store large numbers of users and groups. The WebLogic LDAP user store is only sufficient for storing a small number of users and policies for roles and entitlements.

For information on designing visitor entitlements for performance, see "Designing Visitor Entitlements for Performance" on page 8-27.

Part II Development

In the development phase, you can add log in and log out functionality to your portal applications using WorkSpace Studio.

Part II includes the following chapters:

- Chapter 3, “Securing Your Portal Deployment”
- Chapter 10, “Implementing Authorization Programmatically”
- Chapter 4, “Preventing Direct Access to Portal Application Resources”

You can also use the isUserInRole method (or tag) to verify role membership for a specific user within your application logic. This type of programmatic security, coded into portlets, books, and pages, enables you to customize a user’s path through a portal and perform fine-grained entitlement-checking.

You can use the isAccessAllowed method (or tag) to allow or deny access to a specific application resource within your application logic. This is the same method that is automatically called by the runtime framework to determine access to portal resources.

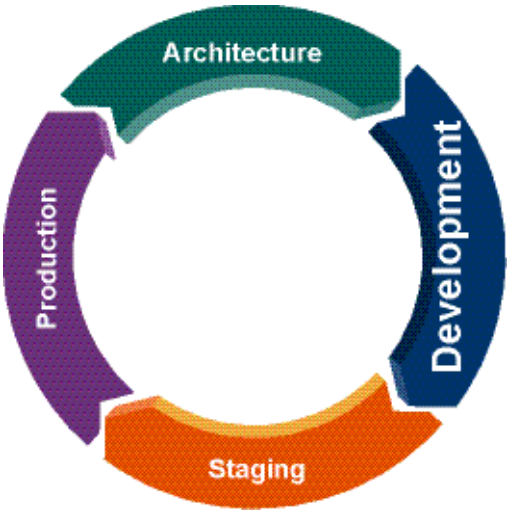
The decisions you made during the architecture phase shape what you do in the development phase.

Consider setting up a common development environment for the development phase and the staging phase. You can use the staging phase as a staging environment to add users and groups to test the functionality you created in the development phase. You might move iteratively between these two phases, developing and then testing what you created. The users and groups you add in

the staging environment might be different than the users and groups you use in the production system.

If you have moved on to the production phase and then go back to make changes that affect the development phase, you must redeploy your portal application in order to view your changes.

For a detailed description of the development phase of the portal life cycle, see the WebLogic Portal Overview. The portal life cycle is shown in the following graphic:



CHAPTER 3

Securing Your Portal Deployment

This chapter discusses best practices, tips, and techniques for securing your portal deployment. While other chapters in this guide specifically deal with securing access to portal resources by adding entitlements, managing security providers, and setting up delegated administration, this chapter discusses additional steps you can take to secure your portal in a production environment, such as using firewalls, securing database connections, securing WSRP producers, and blocking access through non-HTTP protocols, such as RMI, EJB, and JNDI.

Tip: The information in this chapter is specific to WebLogic Portal. Before continuing, we recommend you read the WebLogic Server document, *Securing a Production Environment* for detailed information on securing your WebLogic Server installation.

This chapter discusses the following security topics:

- Encrypting Sensitive Information
- Using Firewalls
- Securing the WebLogic Portal Administration Console
- Securing Database Communications
- Reviewing Policies and Visitor Entitlements
- Securing WSRP Applications
- Blocking Non-HTTP Protocols

- Securing the Content Management System
- Securing UUP Data
- Application-Scoping Resources
- Securing GroupSpace Applications
- Securing WebDAV Web Application
- Implementing Authentication Programmatically

Encrypting Sensitive Information

A simple security measure is to ensure that sensitive data, such as passwords or credit card numbers, are never stored in clear text.

Using Firewalls

Use a firewall to secure your portal software. A number of firewall options are available, from IP protocol filtering to content filtering. Each of these can provide a good starting point for securing your portal software.

You can obtain the best protection by using a two firewall solution. Place a firewall in front of WebLogic Server (or a certified WebLogic-supported server) and use a WebLogic-supported server plug-in. The plug-in communicates the portal software through another firewall. These firewalls can be configured to filter out all traffic except HTTP and HTTPS. In addition, the plug-in and WebLogic-supported server can be configured to perform two-way SSL between the plug-in and WebLogic-supported server.

Depending on the firewall it may be possible in some situations (non-encrypted) to filter the content of the packets that pass between the plug-in and the web server.

The drawback to this method is that both firewalls are filtering the same protocol; therefore, if an attacker can pass though the first firewall then they could pass though the second firewall as well. To mitigate this possible security hole, a certified secure version of a supported WebLogic web server can be used.

For more information on firewalls, see “Supported Web Servers, Browsers, and Firewalls.”

Securing the WebLogic Portal Administration Console

One of the simplest measures you can take is to secure the WebLogic Portal Administration console.

Table 3-1 Securing the Administration Console

Security Action	Description
Change the default user name and password.	It is recommended that you use a different user name and password for portal administration than the ones used for administration of WebLogic Server.
Do not deploy in production.	If you do not have a reason to publicly deploy the Administration Console in your production environment, undeploy it or protect it behind your firewall.
Use a WebLogic virtual host and network channel to restrict authorized access only from within a specific network.	Configure the network channel to only listen on HTTPS and only on an internal network IP address. Configure the virtual host to use the network channel. Target the Administration Console only to the virtual host so that it is only be visible to users with access to the internal network IP address. See the WebLogic Server documents, "Virtual Hosts" and "Configure Custom Network Channels."

Securing Database Communications

In some extreme cases it may be beneficial to secure database communications. While not all drivers support such a mechanism a number of providers do provide encrypted communication to the database. This extra encryption does come with a heavy price in performance but is an option for the most security sensitive customers.

Reviewing Policies and Visitor Entitlements

It is important to review the delegated administration and visitor entitlement policies before deploying a portal to a production environment. Delegated administration provides a mechanism for propagating WebLogic Portal Administration Console privileges within a hierarchy of roles. Visitor entitlements allow you to define who can access the resources in a portal application and what they can do with those resources.

Table 3-2 Reviewing Policies and Visitor Entitlements

Security Action	Description
Review the delegated administration policies.	Be sure the delegated administration policies are configured the way you want them. It is recommended that you change the default group. For detailed information, see Chapter 7, "Configuring Delegated Administration."
Review visitor entitlements.	Be sure to define entitlements for each portlet in the Library. Entitlements are a complicated subject and it is important to read the documentation and understand their configuration before deploying your portal. For detailed information, see Chapter 8, "Configuring Visitor Entitlements."

Securing WSRP Applications

This section lists best practices for securing WebLogic Portal applications that use WSRP. For detailed information on WSRP security, see the *Federated Portals Guide*.

Table 3-3 Securing WSRP Applications

Security Action	Description
For producer applications, protect the WSDL path, and all paths corresponding to the WSDL ports.	The producer's WSDL should declare ports for HTTPS (SSL) service entry points only. For more information on WSDL, see the chapter "Federated Portal Architecture" in the *Federated Portals Guide.*
Configure the producer and consumer to always require a security token.	By default, a token is sent only when a user is authenticated. See the chapter "Establishing WSRP Security with SAML" in the *Federated Portals Guide* for detailed information.
When logging out on a consumer, always do the following: • Terminate the session • Enable destroySessions support	These steps ensure all producer sessions are also terminated.

Table 3-3 Securing WSRP Applications

Security Action	Description
Selectively offer portlets, books, and pages as remote.	By default, all portlets deployed in a WebLogic Portal producer application are available to consumers as remote portlets. You can, however, specify which portlets are actually available to consumers by setting the **Offer As Remote** property in the WorkSpace Studio **Properties** view for the portlet. Apply this same action to remoteable books and pages. For more information, see the chapter "Offering Books, Pages, and Portlets to Consumers" in the *Federated Portals Guide.*
Use consumer entitlements.	Consumer entitlement allows producers to decide which portlets to offer to consumers based on registration properties. For detailed information, see the chapter "Consumer Entitlement" in the *Federated Portals Guide.*
When publishing portlets, books, or pages to a UDDI registry, use credential aliases instead of username/passwords.	For detailed information on publishing to UDDI registries, see the chapter "Publishing to UDDI Registries" in the *Federated Portals Guide.*
Avoid using User Name Token (UNT). If you *must*, use UNT with password digest enabled.	Username Token, or UNT, is an alternative to SAML and provides the same basic single sign-on capability as SAML provides. Username Token lets you map the local user on the consumer to a user on the producer. For detailed information, see the chapter "Configuring Username Token Security" in the *Federated Portals Guide.*
Require signatures when using SAML.	When configuring the Asserting Party properties on the producer, enable the Signatures Required property. This action requires all assertions to be signed. For more information, see the chapter "Establishing WSRP Security with SAML" in the *Federated Portals Guide.*
Manage the keystore carefully.	Generate your own keys in the keystore, and delete the default keys from the keystore. For detailed information, see the chapter "Establishing WSRP Security with SAML" in the *Federated Portals Guide.*

Table 3-3 Securing WSRP Applications

Security Action	Description
Disable the WSRP SOAP monitor.	You can monitor activity between producers and consumers by using the message monitor servlet installed with WorkSpace Studio. For information on the monitor, see the chapter "Other Topics and Best Practices" in the *Federated Portals Guide.*
Use a custom resource connection filter that retrieves only valid and known resources.	By default, an unprotected resource can be retrieved from any known producer server. For more information, see the WebLogic Server section "Using Connection Filters."

Blocking Non-HTTP Protocols

Firewall the server so that all non-HTTP protocols are blocked. These include such protocols as JNDI, EJB, RMI, and T3.

Securing the Content Management System

This section includes tips on securing the content management system.

Table 3-4 Securing the Content Management System

Security Action	Description
Place entitlements on all content nodes that could contain sensitive data.	For detailed information on entitling content nodes, see "Setting Visitor Entitlements on Content Management Resources" on page 8-22.
Review delegated administration policies for the content management resources.	Be sure the delegated administration policies are configured the way you want them for the content management resources. For more information, see "Setting Delegated Administration on Content Management Resources" on page 7-27.
Secure the WebDAV API.	See "Securing WebDAV Web Application" on page 3-8.

Securing UUP Data

Unified User Profile (UUP) allows you to store user-specific information. For detailed information on UUP and WebLogic Portal, see the chapter "Configuring a UUP," in the *User Interaction Management Guide*.

Table 3-5 Securing UUP Data

Security Action	Description
Encrypt sensitive information stored as UUP data.	If you intend to store sensitive information, such as credit card numbers, as UUP data, you must take measures to encrypt that data. I would repeat the general statement here: **Tip:** Never store passwords or sensitive data (like credit cards) in clear text.
Avoid storing sensitive user profile information in the default profile store. Use UUP instead.	Using UUP allows the data to be stored further back in the network stack. Sensitive customer data preexisting in protected repositories can be accessed only as needed through a custom UUP implementation.
Protect UUP access methods.	UUP access methods can be role protected through EJB method protection declaration in `ejb-jar.xml`.

Application-Scoping Resources

Configure application-scoped resources to further protect your portal system. To do this, you need to create JDBC Pool definitions in the portal application and reference them in the `weblogic-application.xml` configuration file. In addition to JDBC Pools, any JMS resources can also be application-scoped in the same manner.

For more information, see "Configuring JDBC Application Modules for Deployment" and "Configuring JMS Application Modules for Deployment."

Securing GroupSpace Applications

This section explains how to prevent users from creating new GroupSpaces from within GroupSpace. The ability of members to create new GroupSpaces from within GroupSpace is

controlled by the community role (the admin designation of your community capability). For more information on community roles and capabilities, see the *WebLogic Portal Communities Guide*.

Specified in the `communities-config.xml` file, the following kind of member can access the administration tools to create a new GroupSpace from within another GroupSpace:

```
<capability>
     <name>manager</name>
     <display-name>Manager</display-name>
     <admin>true</admin>
 </capability>
```

The following kind of member cannot:

```
<capability>
     <name>memberuser</name>
     <display-name>Member User</display-name>
 </capability>
```

You can check the admin capability of any member with the code in Listing 3-1:

Listing 3-1 Checking the Admin Capability of a GroupSpace Member

```
CommunityContext cc = CommunityContext.getCommunityContext(request);
CommunityUserContext userContext = communityContext.getCommunityUserContext();
CommunityMember thisMember = userContext.getMember();
CommunityMembership membership = userContext.getMembership(thisMember.getId(),
  cc.getCommunity().getCommunityDefinitionId());
if(membership.hasAdminCapability())
{
    ...
}
```

Securing WebDAV Web Application

This section explains how to secure the WebDAV web application. WebDAV is automatically deployed with your WebLogic Portal application.

Listing 3-2 shows the defatul security constraint stanza found in the `web.xml` for the WebDAV web application WAR file, `WEBLOGIC_HOME\cm\lib\modules\webdav-web-lib.war`. To

change the default security setting for WebDAV, you can change the default `<role-name>` attribute of the `<auth-constraint>` element to only allow members of a specific administrative role to have access to the WebDAV web application.

Tip: It is recommended that you use a deployment plan to modify the default security constraint in `web.xml` settings. See the *Production Operations Guide* for information on using deployment plans.

Listing 3-2 WebDAV Configuration in web.xml

```
<security-constraint id="securityconstraint">
     <web-resource-collection>
          <web-resource-name>webdav</web-resource-name>
          <description>Security constraint for webdav</description>
          <url-pattern>/*</url-pattern>
     </web-resource-collection>
     <auth-constraint>
          <role-name>AllAuthenticatedUsers</role-name>
     </auth-constraint>
</security-constraint>
```

Implementing Authentication Programmatically

This section explains best practices and techniques for implementing login/logout functionality programatically. Sample code to handle login and logout in a JSP portlet is included in this section.

Always Redirect After Login or Logout

Redirecting after a login or logout prevents certain serious security problems, such as:

- Receiving a previous user's session in a WSRP portlet.
- Seeing previous form values show up in a new page.
- Seeing the wrong portlet instance, such as a user instance after logout.

The recommended redirect techniques for commonly used portlet types are:

- For JSP portlets, the best practice is to redirect and reload the portal after a log in or a log out operation using the portal framework's JspContentContext.sendRedirect() method. See "Sample JSP Login/Logout Code" on page 3-10.
- For Struts portlets, use the JspContentBackingContext.sendRedirect() method to perform the redirect.

  Note: Avoid calling HttpResponse.sendRedirect() to perform the redirect in JSP and Struts portlets. The best practice is to always use the appropriate WLP mechanisms. See also "Avoid Using JSP Tags for Login and Logout" on page 3-10.
- For page flow portlets, the best practice is to perform the redirect by marking the forward from the login action with the redirect annotation. Another technique is to place the login/logout logic in the page flow Controller.
- For JSF portlets, you can call HttpResponse.sendRedirect() to perform the redirect. Be sure to obtain the response object from the ExternalContext, which functions as a request wrapper that ensures the portal takes the correct action.

Avoid Using JSP Tags for Login and Logout

The `<auth:login>` tag performs weak authentication (of the username and password combination) against the current security realm, and sets the authenticated user as the current WebLogic user. The `<auth:logout>` tag ends the current user's WebLogic Server session.

WARNING: The use of these tags can lead to certain security problems because they are called after rendering has started. At the render phase of the portlet life cycle, it is impossible to apply the best practice of redirecting the portal, as explained in "Always Redirect After Login or Logout" on page 3-9. The redirect must be called during the handlePostbackData life cycle phase.

For more information on these tags refer to the JSP Tag Javadoc.

Sample JSP Login/Logout Code

The sample code listed in this section demonstrates how to perform a log in and log out in a JSP portlet. Listing 3-3 lists a backing file that uses the com.bea.p13n.security.Authentication helper class to perform log in and log out operations. This code also demonstrates the best practice of calling JspContentContext.sendRedirect() after a log in or a log out. See also "Always Redirect After Login or Logout" on page 3-9.

Listing 3-4 shows a sample JSP that submits the user's name and password to the server. To use this JSP, create a portlet from it and add the backing file in Listing 3-3 to the portlet.

Note: The portal framework can only perform a redirect during the handlePostbackData phase of its life cycle; therefore, the authentication code is placed in the handlePostbackData() method of the backing file, as demonstrated in Listing 3-3.

Listing 3-3 Backing File That Performs Authentication

```
package examples.login;

import com.bea.netuix.servlets.controls.content.JspContentContext;
import com.bea.netuix.servlets.controls.content.backing.AbstractJspBacking;
import com.bea.p13n.security.Authentication;
import com.bea.portlet.GenericURL;
import com.bea.portlet.PostbackURL;

import javax.security.auth.login.LoginException;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class LoginBacking extends AbstractJspBacking {
  private static final long serialVersionUID = 1L;
  public static final String REDIRECT_ACTION = "redirect";

  public boolean handlePostbackData(HttpServletRequest request,
    HttpServletResponse response) {
        if (isRequestTargeted(request)) {
          if (request.getParameter(GenericURL.STATE_PARAM) == null) {
              String username = request.getParameter("username");
              String password = request.getParameter("password");

              PostbackURL url = PostbackURL.createPostbackURL(request, response);

              if (username != null && password != null) {
                try {
                  Authentication.login(username, password, request, response);
                }
                catch (LoginException le) {
                 request.setAttribute("loginErrorMessage3", new String("true"));
```

```
                return false;
              }
            }
            else if (request.getParameter("logout") != null) {
               Authentication.logout(request);
            }

            url.addParameter(GenericURL.LOADSTATE_PARAM, "false");
            url.addParameter(GenericURL.PAGE_LABEL_PARAM, "login");

            try {
                JspContentContext jspContext =
                 JspContentContext.getJspContentContext(request);
              jspContext.sendRedirect(url.toString());
            }
            catch (Exception ie) {
               ie.printStackTrace();
            }
          }
       }
       return true;
   }
}
```

Listing 3-4 Sample Login JSP

```
<%@ page import="com.bea.portlet.WindowURL" %>

<h3 align="center">WebLogic Portal Login/Logout</h3>

<%
   WindowURL url = WindowURL.createWindowURL(request, response);

   if (request.getUserPrincipal() == null)
   {
      String errorMessage = (String) request.getAttribute("loginErrorMessage3");
%>
<form method="post" id="backingFileLoginForm" action="<%=url.toString()%>"
type="POST">
```

```
   <table border="0" width="100%">
      <tr>
           <td align="center" colspan="2">
              <% if (errorMessage != null) { %>
              <font color="red">Login failed. Please try again.</font><br>
              <% } %>
              Please enter your username and password below.<br>
           </td>
      </tr>
      <tr>
           <td align="right">
              Username:
           </td>
           <td align="left">
              <input type="text" size=15 name="username" >
           </td>
      </tr>
      <tr>
           <td align="right">
              Password:
           </td>
           <td align="left">
              <input type="password" size=15 name="password" >
           </td>
      </tr>
      <tr>
           <td colspan="2" align="center">
              <input type="submit" value="Login">
           </td>
      </tr>
   </table>
</form>
<%
   }
   else
   {
%>
<form method="post" id="backingFileLoginForm" action="<%=url.toString()%>"
type="POST">
   <table border="0" width="100%">
      <tr>
```

```
         <td align="center">
            <b><%=request.getUserPrincipal().getName()%></b>, Welcome to
              WebLogic Portal!
         </td>
      </tr>
      <tr>
         <td align="center">
            <input type="hidden" name="logout" value="1">
            <input type="submit" value="Logout">
         </td>
      </tr>
   </table>
</form>
<%
}
%>
```

CHAPTER 4

Preventing Direct Access to Portal Application Resources

You can control visitor access to portal resources using visitor entitlements in the WebLogic Portal Administration Console. However, you must also use deployment descriptors to secure the JSPs and page flows contained in a portlet; otherwise a malicious user can access those resources directly if they know the correct URL.

This chapter contains the following section:

Securing Resources Using Deployment Descriptors

You must use J2EE security to prevent direct access to JSPs and page flows; otherwise, a user can access those resources directly by entering the correct URL.

Note: Descriptor security is only intended to prevent direct access to the JSP or page flow using a URL; it is not used when a portal renders a portlet.

Scoped roles are defined in their respective deployment descriptors.

- Enterprise-application-scoped roles are defined in `application.xml` and `weblogic-application.xml`
- Web-application-scoped roles which apply to resources within a project are defined in `web.xml` and `weblogic.xml`
- EJB-scoped roles which apply only to resources within an EJB are defined in `ejb-jar.xml` and `weblogic-ejb-jar.xml`

An example URL to a JSP is:

```
http://emp_app/employmentPortal/portlets/hr/vpSalaries.jsp
```

To prevent direct access to portlets, add a security entry in your portal web project's `/WEB-INF/web.xml` file. Listing 4-1 shows an example `web.xml` file.

Listing 4-1 Using Declarative Security to Block Direct Access to Portlets

```
!-- Use declarative security to block direct address to portlets -->
<security-constraint>
   <display-name>Default Portlet Security Constraints</display-name>
      <web-resource-collection>
      <web-resource-name>Portlet Directory</web-resource-name>
      <url-pattern>/portlets/*</url-pattern>
      <http-method>GET</http-method>
      <http-method>POST</http-method>
   </web-resource-collection>
<auth-constraint>
   <role-name>Admin</role-name>
</auth-constraint>
<user-data-constraint>
   <transport-guarantee>NONE</transport-guarantee>
</user-data-constraint>
</security-constraint>
```

This security entry in the `web.xml` file protects all files in the portal web project's `/portlet` directory and its subdirectories from being directly accessed using a request URL.

WARNING: A `<url-pattern>` of `/portlets/*.jsp` is not legal syntax and does not protect subdirectories.

These protected resources are still displayed in entitled portlets, but only for users entitled to access those portlets.

Resources such as images, which do not require security restrictions, must be stored in unsecured directories outside the `/portlets` directory.

Note: Certain URL or EJB resources can be secured using the WebLogic Server Administration Console. Before using this technique, you must copy security configurations from existing deployment descriptors during the initial deployment of URL or EJB resources, or reinitialize the security configuration for URL or EJB resources to their original state. For more information see Import Security Data from Deployment Descriptors.

CHAPTER 5

Securing Third-Party Applications

Some portlets need to connect to back-end systems or remote applications with a username and password. If a remote application uses the same credentials as those used by the portal, the portlets can re-use the credentials. However, in most cases, the credentials required by remote systems are not same as user credentials used for logging into the portal. For optimal user experience, a single sign-on is needed. The Credential Vault provides a safe storage mechanism to securely set and retrieve credentials for portlets accessing remote applications.

Understanding the Credential Vault

In previous releases of WebLogic Portal, you could implement functionality similar to the Credential Vault by storing credentials in user profiles. However, user profile properties are not encrypted and not a safe place for storing credentials. Although it is possible to encrypt properties using WebLogic Server encryption methods, they are complex, limited, and required custom programming. Additionally, JCA cannot be used because JCA adapters do not fit into the portlet scope.

The Credential Vault provides APIs that allow portlets to store and access user credentials and use those credentials to log into remote applications on behalf of the user. With these APIs, a developer can build secure repositories that store usernames and passwords, plus optional metadata required by the resource, such as a URL. The username and password are encrypted, while the metadata (name value pairs of `String` type) are stored in plain text. The Credential Vault does not provide the mechanism to pass the credentials to the remote system.

In addition to the APIs, the Credential Vault provides a GUI in the WebLogic Portal Administration Console, where a portal administrator can create a system credential vault. The three types of credential vaults are as follows:

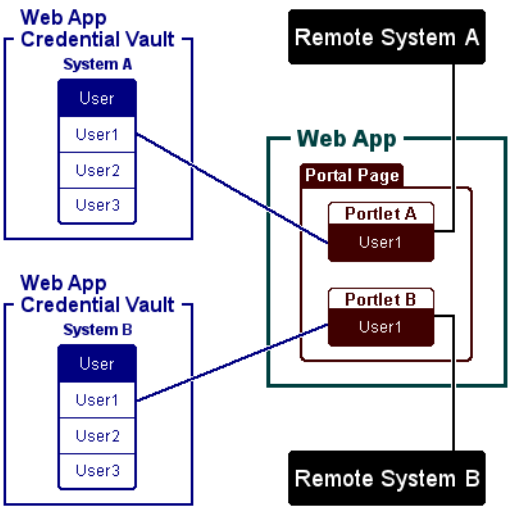
- User Credential Vault
- User + Resource Credential Vault
- System Credential Vault

User Credential Vault

A user credential vault securely stores a user's credentials for a remote system for individual portlets based only on the username. In other words, a credential entry is accessible to all the resource instances of a logged-in user. In the WebLogic Portal Javadoc, this entry is called USER_TYPE.

Using the Credential Vault APIs in a typical implementation, a portlet is created that allows a user to provide their login information to a remote system. After that, whenever the user logs into the portal, the portlet code automatically reads the login information and passes it to the remote application without the user having to re-enter their credentials. The following figure shows an example of this type of credential vault, where two portlets on the same portal page connect to two different remote systems.

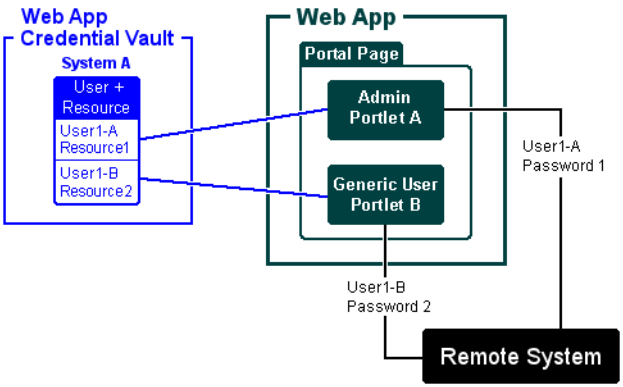
Figure 5-1 User Credential Vaults



User + Resource Credential Vault

Use this type of credential vault when a portal contains two or more portlets that connect to the same remote system for users who require different credentials. For example, one portlet might connect the user to the remote system as a generic user and the other portlet might connect the user as an administrator. In a User + Resource Credential Vault, credential entry is only accessible to a particular resource instance of the logged-in user. In the WebLogic Portal Javadoc, this entry is called `RESOURCE_TYPE`.

Figure 5-2 User + Resource Credential Vault



System Credential Vault

An administrator can create a system credential vault that stores a global username and password so that credential entry is accessible to all resource instances of all logged-in users. In the WebLogic Portal Javadoc, this entry is called `SYSTEM_TYPE`.

This type of credential vault allows multiple users to share the same password (and username) to access a remote resource. It is best suited for use as a demonstration account, where a user has read-only access to a resource before getting an account. You can use either the Credential Vault APIs or the WebLogic Portal Administration Console to create a system credential vault. For more information, see "Using the Credential Vault APIs" on page 5-6 and "Creating or Viewing System Credentials in the Administration Console" on page 5-17.

Visibility

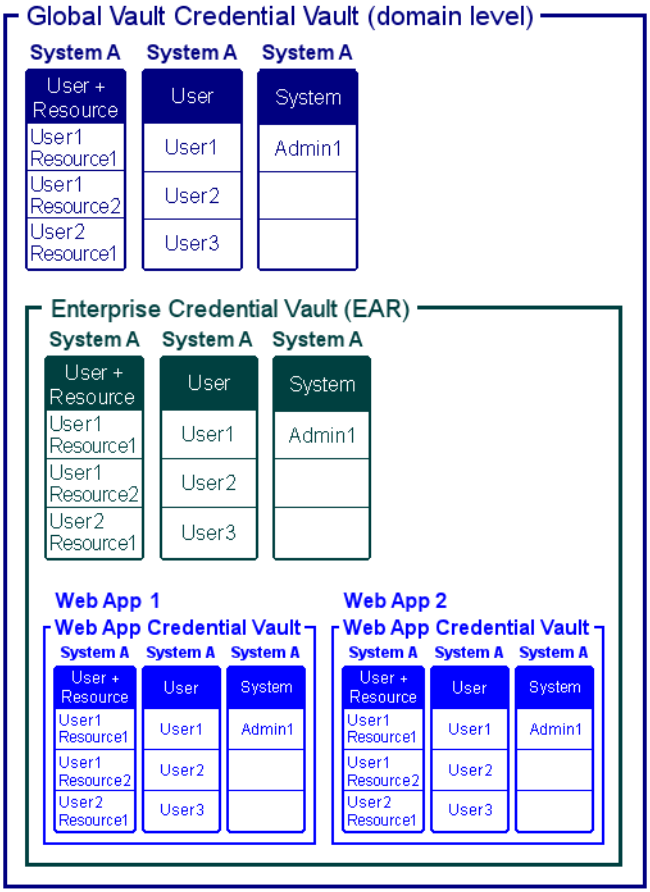
You can specify the scope of the three types of credential vaults entry across web applications, enterprise applications, and within a domain:

- Web Application—web application-scoped credential entries are not visible to different web applications within the same enterprise application.
- Enterprise application—enterprise application-scoped credential entries are not visible to a different enterprise application in the same domain.
- Domain—domain-scoped credential entries are global entries, which are visible to all applications within the domain.

Scoping allows you to shadow the names of the credential vaults within the Java EE application. Within the same scope, the names must be unique. In different scopes you can shadow the name. From an outer scope, all the names are visible to any of its containing scope. For example, if both an EAR scope and WAR scope have a credential vault named *foo*, the credential vault in WAR scope is used because *foo* in WAR scope shadows the *foo* within the EAR scope.

The following figure shows an example.

Figure 5-3 Credential Vaults



When accessing the credential vault, the code associated with a portlet traverses the credential vaults until it finds an entry for the user. If no credentials are found, you can use the APIs to present a login to the user. The following example provides more details:

Suppose an employee logs into a Human Resources portal page with their WebLogic authenticated username and password. This page contains portlets to services such as payroll, stock options, and career development. The employee wants to change their payroll deduction and the payroll system exists on a remote SAP application. When the employee clicks the Payroll portlet, the system looks for the login credentials in the payroll credential vault (Web App). If it finds the employee's credentials, the portlet logs in the user to the SAP payroll application.

If the employee's credentials do not exist in that credential vault, the portlet looks for them in the enterprise credential vault (EAR). If the enterprise credential vault contains the employee's credentials, the portlet logs the employee into the SAP payroll application. If not, the system looks for the credentials in the global credential vault (domain). If the portlet finds the employee's credentials, the employee is logged on to the remote SAP application. If not, the portlet presents a login to the SAP application.

Once the user successfully logs onto the SAP application, the credentials are stored in one of the credential vaults; which credential vault is determined by the system design.

Using the Credential Vault APIs

The package `com.bea.p13n.security.management.credentials`, documented in Javadoc, lets you create entries and access the three types of credential vaults. The Javadoc provides the detailed information that you need for implementing credential vaults. The following sections provide the framework for using creating and accessing credential vaults.

Note: Some of the following sections use `RESOURCE_TYPE` (User + Resource) as an example of the credential vault type. To change type, simply change `RESOURCE_TYPE` to either `USER_TYPE` or `SYSTEM_TYPE`.

Initialize the Credential Vault

The entry point to the credential vault API is through the Credential Vault Service interface:

```
CredentailVaultService cvs =
com.bea.wlp.services.Services.getService(com.bea.p13n.security.management.
credentials.CredentailVaultService.class)
```

Construct the Resource Key

The resource key contains the scope and Resource ID of the credential service. You must specify the scope because the portlet instance is a web-scoped resource.

```
ResourceKey rc = new ResourceKey(Scope.getWebApplicationScope(request),
portletInstanceId);
```

Tip: Although the resource ID is not required for User or System credential vaults, it is a best practice to use the actual Resource ID. This will allow you to easily convert the credential vault to a different type in the future.

Creating a Credential Entry

When you create an entry you set the resource key and entry type for the credential vault. The entry type is one of the following: `USER_TYPE` (User), `RESOURCE_TYPE` (User + Resource), or `SYSTEM_TYPE` (System).

```
return cvs.createCredentialEntry(entryName, EntryType.RESOURCE_TYPE, null,
rc);
```

You can also provide a description for the entry. If you do not want a description, use `null`.

Accessing a Credential Entry

When you access an entry, you supply the resource key and entry type for the credential vault. The entry type is one of the following: `USER_TYPE` (User), `RESOURCE_TYPE` (User + Resource), or `SYSTEM_TYPE` (System).

```
return cvs.fetchCredentialEntry(entryName, EntryType.RESOURCE_TYPE, rc);
```

Note: If more than one entry exists with same name but a different scope, the one with closest scope to requesting resource is retrieved.

Updating a Credential Entry

To add or change the username and password, use `UserPasswordCredential`.

```
entry.setCredential(new UserPasswordCredential(username,
password.getBytes("UTF-8")));
```

Deleting a Credential Entry

To remove a credential entry, use `removeCredentialEntry`.

```
cvs.removeCredentialEntry(entryName, EntryType.RESOURCE_TYPE, rc);
```

Credential Vault Examples

Sample code showing CRUD operations for each type of credential vault may be available on Dev2dev. On Dev2Dev, search for Credential Vault.

The following examples show the CRUD operations for a user (`USER_TYPE`) credential vault. For User + Resource and System Credential Vaults, you only need to change the type to `RESOURCE_TYPE` or `SYSTEM_TYPE` respectively.

- `CredentialVaultHelper`—demonstrates the code you need for using the Credential Vault API.
- `EmailAccountBacking` file—shows the portlet logic needed get and update an entry in a User Credential Vault.
- `index.jsp`—displays the email account credential entry.
- `config.jsp`—updates the email account credential entry.

The `CredentialVaultHelper` class demonstrates all of the code you need for using the Credential Vault API.

Listing 5-1 CredentialHelper Class

```
package examples.credential;

import javax.servlet.http.HttpServletRequest;

import
com.bea.p13n.security.management.credentials.AlreadyExistsException;
import
com.bea.p13n.security.management.credentials.CredentialVaultService;
import com.bea.p13n.security.management.credentials.ResourceKey;
import com.bea.p13n.security.management.credentials.Scope;
import com.bea.p13n.security.management.credentials.CredentialEntry;
import
com.bea.p13n.security.management.credentials.CredentialEntry.EntryType;

public class CredentialHelper
{
   private static CredentialVaultService cvs =
com.bea.wlp.services.Services.getService(CredentialVaultService.class);

   public static CredentialEntry
createCredentialEntryForPortletInstance(String entryName,
HttpServletRequest request)
      throws AlreadyExistsException
```

```
   {
      String portletInstanceId =
com.bea.netuix.util.PortalFrameworkUtils.getUniquePortletDesktopString(req
uest);
      // need webapp scope since portlet instance is web scoped resource.
      ResourceKey rc = new
ResourceKey(Scope.getWebApplicationScope(request), portletInstanceId);
      return cvs.createCredentialEntry(entryName, EntryType.RESOURCE_TYPE,
null, rc);
   }
   public static CredentialEntry
getCredentialEntryForPortletInstance(String entryName, HttpServletRequest
request)
   {
      String portletInstanceId =
com.bea.netuix.util.PortalFrameworkUtils.getUniquePortletDesktopString(req
uest);
      // need webapp scope since portlet instance is web scoped resource.
      ResourceKey rc = new
ResourceKey(Scope.getWebApplicationScope(request), portletInstanceId);
      return cvs.fetchCredentialEntry(entryName, EntryType.RESOURCE_TYPE,
rc);
   }

   public static void removeCredentialEntryForPortletInstance(String
entryName, HttpServletRequest request)
   {
      String portletInstanceId =
com.bea.netuix.util.PortalFrameworkUtils.getUniquePortletDesktopString(req
uest);
      // need webapp scope since portlet instance is web scoped resource.
      ResourceKey rc = new
ResourceKey(Scope.getWebApplicationScope(request), portletInstanceId);
      cvs.removeCredentialEntry(entryName, EntryType.RESOURCE_TYPE, rc);
   }

   public static CredentialEntry createUserCredentialEntry(String
```

```
entryName, HttpServletRequest request)
      throws AlreadyExistsException
   {
      // resourceId is not important in user type credential, you can use
anything you like
      String dummyResourceId =
com.bea.netuix.util.PortalFrameworkUtils.getUniquePortletDesktopString(req
uest);
      // this case use enterprise scope, so all portal web can see this
entry, but not another enterpise app target to external users.
      ResourceKey rc = new ResourceKey(Scope.getApplicationScope(),
dummyResourceId);
      return cvs.createCredentialEntry(entryName, EntryType.USER_TYPE,
null, rc);
   }

   public static CredentialEntry getUserCredentialEntry(String entryName,
HttpServletRequest request)
   {
      // resourceId is not important in user type credential, you can use
anything you like
      String dummyResourceId =
com.bea.netuix.util.PortalFrameworkUtils.getUniquePortletDesktopString(req
uest);
      // this case use enterprise scope, so all portal web can see this
entry, but not another enterpise app target to external users.
      ResourceKey rc = new ResourceKey(Scope.getApplicationScope(),
dummyResourceId);
      return cvs.fetchCredentialEntry(entryName, EntryType.USER_TYPE, rc);
   }

   public static void removeUserCredentialEntry(String entryName,
HttpServletRequest request)
   {
      // resourceId is not important in user type credential, you can use
anything you like
      String dummyResourceId =
com.bea.netuix.util.PortalFrameworkUtils.getUniquePortletDesktopString(req
uest);
```

```
      // this case use enterprise scope, so all portal web can see this
entry, but not another enterpise app target to external users.
      ResourceKey rc = new ResourceKey(Scope.getApplicationScope(),
dummyResourceId);
      cvs.removeCredentialEntry(entryName, EntryType.USER_TYPE, rc);
   }

   // only portal administrator have the privilige to create system type
credential
   public static CredentialEntry createCommonCredentialEntry(String
entryName, HttpServletRequest request)
      throws AlreadyExistsException
   {
      // resourceId is not important in system type credential, you can use
anything you like
      String dummyResourceId =
com.bea.netuix.util.PortalFrameworkUtils.getUniquePortletDesktopString(req
uest);
      // this case use enterprise scope, so all portal web can see this
entry, but not another enterpise app target to external users.
      ResourceKey rc = new ResourceKey(Scope.getApplicationScope(),
dummyResourceId);
      return cvs.createCredentialEntry(entryName, EntryType.SYSTEM_TYPE,
null, rc);
   }

   public static CredentialEntry getCommonCredentialEntry(String entryName,
HttpServletRequest request)
   {
      // resourceId is not important in system type credential, you can use
anything you like
      String dummyResourceId =
com.bea.netuix.util.PortalFrameworkUtils.getUniquePortletDesktopString(req
uest);
      // this case use enterprise scope, so all portal web can see this
entry, but not another enterpise app target to external users.
      ResourceKey rc = new ResourceKey(Scope.getApplicationScope(),
dummyResourceId);
      return cvs.fetchCredentialEntry(entryName, EntryType.SYSTEM_TYPE,
```

```
rc);
   }
   // only portal administrator have the privilige to create system type
credential
   public static void removeCommonCredentialEntry(String entryName,
HttpServletRequest request)
   {
      // resourceId is not important in system type credential, you can use
anything you like
      String dummyResourceId =
com.bea.netuix.util.PortalFrameworkUtils.getUniquePortletDesktopString(req
uest);
      // this case use enterprise scope, so all portal web can see this
entry, but not another enterpise app target to external users.
      ResourceKey rc = new ResourceKey(Scope.getApplicationScope(),
dummyResourceId);
      cvs.removeCredentialEntry(entryName, EntryType.SYSTEM_TYPE, rc);
   }
}
```

The `EmailAccountBacking` file shows an example of the portlet logic needed get and update an entry in a User (`USER_TYPE`) Credential Vault. For more information about backing files, see Backing Files in the *Portlet Development Guide*.

Listing 5-2 EmailAccountBacking File

```
package examples.credential;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import com.bea.netuix.servlets.controls.content.backing.AbstractJspBacking;
import com.bea.netuix.servlets.controls.portlet.backing.PortletBackingContext;
import com.bea.netuix.servlets.controls.window.WindowCapabilities;

import com.bea.p13n.security.management.credentials.AlreadyExistsException;
import com.bea.p13n.security.management.credentials.CredentialEntry;
import com.bea.p13n.security.management.credentials.UserPasswordCredential;
```

```
public class EmailAccountBacking extends AbstractJspBacking {

   private static final long serialVersionUID = -4933113295375846721L;

   private CredentialEntry entry;


   @Override
   public boolean handlePostbackData(HttpServletRequest request,
HttpServletResponse response)
   {
      // retrieve credential entry from vault
      entry =
CredentialHelper.getCredentialEntryForPortletInstance("emailAccount",
request);

      if ( isRequestTargeted(request) )
      {
         // get submitted parameters if transit from config page
         String server = request.getParameter("server");
         String protocol = request.getParameter("protocol");
         String username = request.getParameter("username");
         String password = request.getParameter("password");

         if ( username != null && password != null)
         {
            if ( entry == null )
            {
               // create a new credential entry if not already exists
               try
               {
                  entry =
CredentialHelper.createCredentialEntryForPortletInstance("emailAccount",
request);
               }
               catch (AlreadyExistsException e)
               {
                  e.printStackTrace();
               }
            }

            // reset or update credential and/or properties
            try
            {
               entry.setCredential(new UserPasswordCredential(username,
password.getBytes("UTF-8")));
               entry.setAttribute("server", server);
               entry.setAttribute("protocol", protocol);
            }
            catch (Exception e)
```

```
            {
               e.printStackTrace();
            }
         }
         else if (request.getParameter("delete") != null)
         {
            // delete the credential and move back to config page
            CredentialHelper.removeCredentialEntryForPortletInstance("emailAcc
ount", request);
            PortletBackingContext.getPortletBackingContext(request).setupModeC
hangeEvent(WindowCapabilities.EDIT.getName());
            return true;
         }
         return false;
      }

      if ( entry == null )
      {
         // no credential has been set, switch to config page
         PortletBackingContext.getPortletBackingContext(request).setupModeChan
geEvent(WindowCapabilities.EDIT.getName());
         return true;
      }
      else if ( !
WindowCapabilities.VIEW.equals(PortletBackingContext.getPortletBackingContext(
request).getWindowMode()) )
      {
         PortletBackingContext.getPortletBackingContext(request).setupModeChan
geEvent(WindowCapabilities.VIEW.getName());
         return true;
      }

      return false;
   }

   @Override
   public boolean preRender(HttpServletRequest request, HttpServletResponse
response)
   {
      // show this portlet for only login user
      if ( request.getUserPrincipal() == null )
         return false;

      return true;
   }
}
```

The `index.jsp` displays the email account credential entry.

Listing 5-3 Credential Vault index.jsp

```
<%@ page language="java" contentType="text/html;charset=UTF-8"%>
<%@ taglib uri="http://www.bea.com/servers/portal/tags/netuix/render"
prefix="render" %>

<%@page
import="com.bea.p13n.security.management.credentials.UserPasswordCredential"
%>
<%@page import="com.bea.p13n.security.management.credentials.CredentialEntry"
%>
<%@page import="examples.credential.CredentialHelper" %>

<%
   CredentialEntry entry =
CredentialHelper.getCredentialEntryForPortletInstance("emailAccount",
request);
   if (entry == null)
   {
      out.println("Credential not exists. Click edit button on right of portlet
title bar to config credential");
   }
   else
   {
      UserPasswordCredential credential =
(UserPasswordCredential)entry.getCredential();
%>

Email account for this portlet instance is:
<form method="post" action="<render:windowUrl/>">
<table>
   <tr>
      <td>mail server: </td>
      <td><%=entry.getAttribute("server")%>
   </tr>
   <tr>
      <td>protocol: </td>
      <td><%=entry.getAttribute("protocol")%>
   </tr>
   <tr>
      <td>username: </td>
      <td><%=credential.getPrincipalName()%>
   </tr>
   <tr>
      <td>password: </td>
```

```
      <td><%=new String(credential.getPrincipalPassword(), "UTF-8")%>
   </tr>
   <tr>
      <td colspan="2" align="center">
         <input type="hidden" name="delete" value="1">
         <input type="submit" value="delete"/>
      </td>
   </tr>
</table>
</form>
<%
   }
%>
```

The `config.jsp` updates the email account credential entry.

Listing 5-4 Credential Vault config.jsp

```
<%@ page language="java" contentType="text/html;charset=UTF-8"%>
<%@ taglib uri="http://www.bea.com/servers/portal/tags/netuix/render"
prefix="render" %>
<%@ page import="com.bea.netuix.servlets.controls.window.WindowCapabilities" %>

<form method="post" action="<render:windowUrl
windowMode='<%=WindowCapabilities.VIEW.getName()%>'/>">
Note: do not leave field empty, this simple demo does not validate input.<br/>
Otherwise, a NPE exception may appear.
<table>
   <tr>
      <td>mail server: </td>
      <td><input type="text" name="server"/>
   </tr>
   <tr>
      <td>protocol: </td>
      <td><input type="text" name="protocol" />
   </tr>
   <tr>
      <td>username: </td>
      <td><input type="text" name="username" />
   </tr>
   <tr>
      <td>password: </td>
      <td><input type="password" name="password" />
   </tr>
```

```
   <tr>
      <td colspan="2" align="center"><input type="submit" value="save"/></td>
   </tr>
</table>
</form>
```

Creating or Viewing System Credentials in the Administration Console

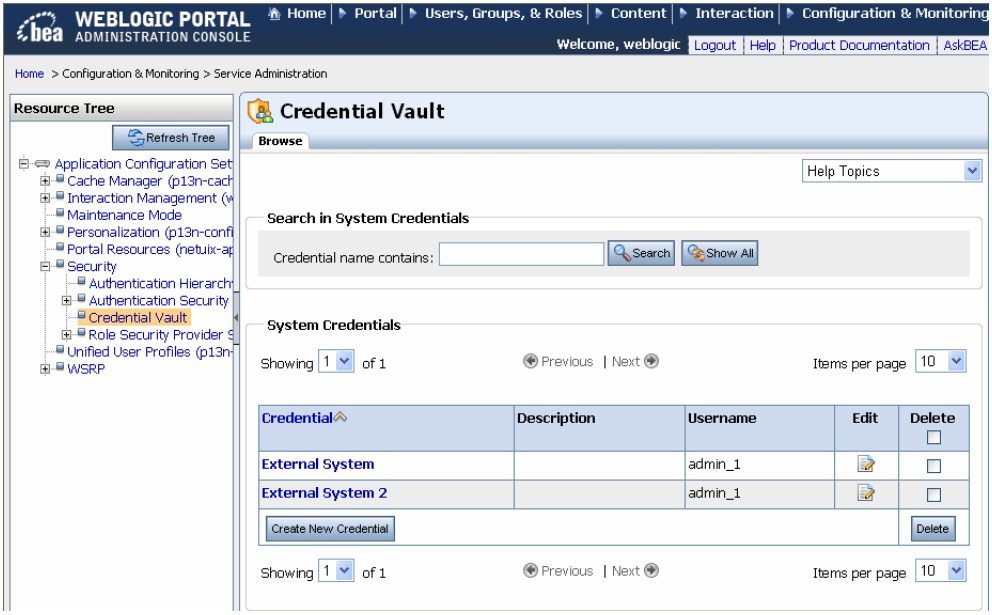
The WebLogic Portal Administration Console provides an administrator with the ability to create a system credential vault. A system credential vault is a vault for a global username and password, as described in "System Credential Vault" on page 5-3.

Note: When creating a system credential vault using the Portal Administration console, you are creating an enterprise application-scoped credential vault whose entries are not visible to a different enterprise application in the same domain.

To view system credentials:

1. Start the Administration Console.
2. Choose **Configuration & Monitoring > Service Administration**.
3. In the Resource Tree, select **Security > Credential Vault**.

Figure 5-4 Credential Vault



4. To create a new credential, click **Create New Credential**.
5. Specify the values specified in Table 5-1.

Table 5-1 System Credentials

Credential Name	The name of the system credential
Description	An optional description of the credential
Username	The name of the portal administrator
Password	The password to access the resource.
Password (again)	Confirm the resource password.
Metadata Name and Value Pair.	The name of the metadata and any data needed by the external system, such as URL or directory path.

6. To enter additional metadata, click **Add Another**.

7. Click **Save**.

Part III Staging

This section contains instructions for managing authentication providers and configuring delegated administration and visitor entitlement roles using the WebLogic Portal Administration Console.

Part III includes the following chapters:

- Chapter 6, “Managing Security Providers”
- Chapter 7, “Configuring Delegated Administration”
- Chapter 8, “Configuring Visitor Entitlements”
- Chapter 9, “Deploying Security Components”

During staging, you test the functionality you created in the development phase. This staging environment simulates a production environment.

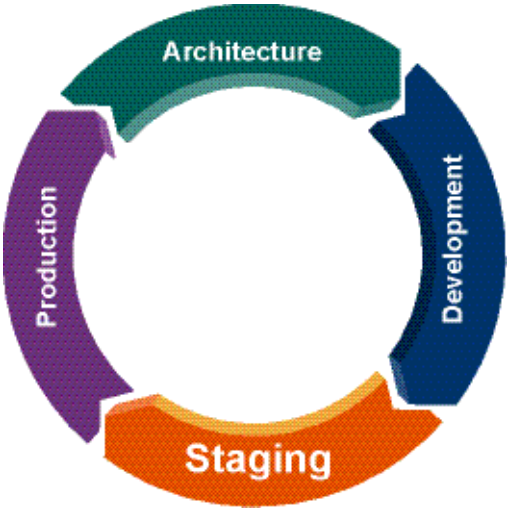
Consider setting up a common development environment for the development phase and the staging phase. You might move iteratively between these two phases, developing and then testing what you created.

As part of the staging phase, verify that delegated administration and visitor entitlement roles that you set up are working. You can also review personalization logic (see the Interaction Management Guide) and content management (see the Content Management Guide) in this phase.

When you move to a production environment in the production phase, your users and groups from the staging phase are not automatically moved there. BEA propagation tools do not move user

and group data. Roles are propagated to the destination. For more information on deploying the security components of your portal, see Chapter 9, “Deploying Security Components”.

For a description of the staging phase of the portal life cycle, see the WebLogic Portal Overview. The portal life cycle is shown in the following graphic:



CHAPTER 6

Managing Security Providers

This chapter describes how to view and configure authentication and role mapping providers and security provider services.

In the **Users, Groups, & Roles > Security Providers** menu, you can view detailed information about how providers have been configured to interact with the WebLogic Portal Administration Console. This menu shows the access privileges for each provider you have configured to supply authentication and role-based authorization capabilities, including whether or not you can view, remove, or modify users, groups, and roles.

In the **Configuration Settings > Service Administration** menu, you can determine whether or not text entry of users and groups is allowed for security providers that do not allow read access, and you can prevent specific users or groups from being created or deleted. You can also configure user management and group management roles that determine runtime operations that can be performed by these roles using the UserProvider and GroupProvider APIs.

This chapter includes the following sections:

Viewing Configured Security Providers

Use the WebLogic Portal Administration Console to view the access privileges for each provider you have configured to supply authentication and role-based authorization capabilities.

The authentication providers and role mappers you connect to WebLogic Server are configured in specific ways. For example, the WebLogic SQL Authenticator is typically configured to allow you to add and remove users and groups using the WebLogic Portal Administration Console, while a custom authenticator may be configured to provide only read access to users and groups.

Perform the following steps to view the configured security providers:

1. Choose **Users, Groups, & Roles > Security Providers**.
2. Select **Security Providers** in the Security Providers tree.

The Browse Security Providers tab shows the title and description for each category of provider, including authentication providers and role mappers, as shown in Figure 6-1.

Figure 6-1 Security Providers

From this tab, you can choose a type of security provider, either Authentication Providers or Role Mappers, to view additional information.

Viewing Configured Authentication Providers

Authentication providers store users, passwords, and groups, which can be viewed and managed directly in those providers. The providers are also configured with rules for how tools such as the WebLogic Portal Administration Console interact with them.

The WebLogic SQL Authenticator (the default authentication provider) and WebLogic LDAP Authenticator provide read and write access from the WebLogic Portal Administration Console (and the WebLogic Server Administration Console) by default.

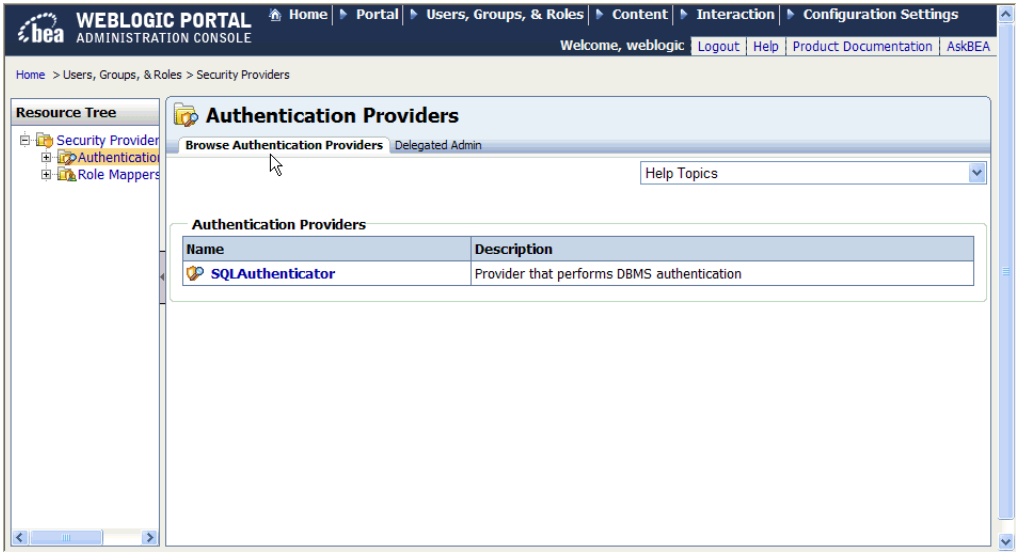
The typical configuration for users and groups in supported external authentication providers is read-only access from the WebLogic Portal Administration Console (and the WebLogic Server Administration Console). To provide write access to external users and groups from the WebLogic Portal Administration Console, you must develop your custom authentication provider to allow write access. If you are using any custom authentication providers, develop them according to the guidelines in How to Develop a Custom Authentication Provider.

Perform the following steps to view the configured authentication providers:

1. Choose **Users, Groups, & Roles > Security Providers**.
2. Select **Authentication Providers** in the Security Providers tree.

The Browse Authentication Providers tab shows the title and description for each authentication provider, as shown in Figure 6-2. At least one authentication provider, `SQLAuthenticator`, is present by default.

Figure 6-2 Authentication Providers



Tip: You can also build group hierarchy trees for authentication providers in the WebLogic Portal Administration Console. A tree view of groups provides a convenient visual mode for changing profile values, finding users within groups, and adding users and groups to delegated administration and visitor entitlement roles. For more information, see the User Management Guide.

Viewing Authentication Provider Details

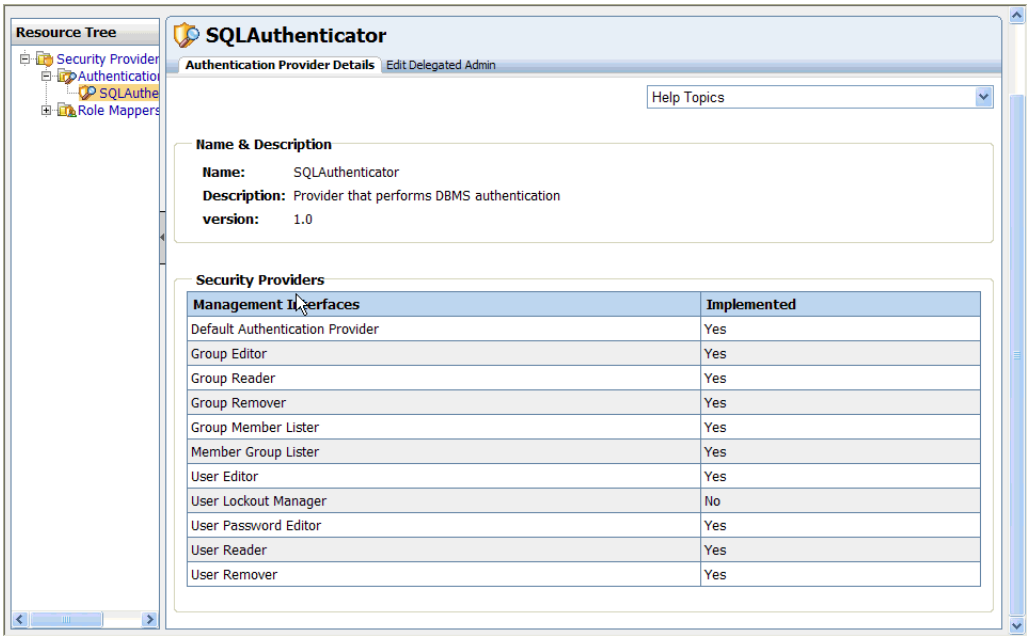
Perform the following steps to view the details for a configured authentication provider:

1. Choose **Users, Groups, & Roles > Security Providers**.

2. Select **Authentication Providers** in the Security Providers tree.
3. Select the authentication provider for which you would like to see details.

The Authentication Provider Details tab shows the name, description, and version of the authentication provider. It also shows which management interfaces are implemented for the provider.

Figure 6-3 Authentication Provider Details



Descriptions of the available management interfaces are listed in Table 6-1.

Table 6-1 Authentication Provider Management Interfaces

Management Interface	Description
Default Authentication Provider	Indicates whether or not this was the first authentication provider configured in WebLogic Server. The default does not change, regardless of which authentication provider is currently being used.
Group Editor	Indicates whether or not you can manage groups with the WebLogic Portal Administration Console; for example, whether you can add groups, move groups, and add users to groups.
Group Reader	Indicates whether or not you can view groups with theWebLogic Portal Administration Console.
Group Remover	Indicates whether or not you can remove groups with the WebLogic Portal Administration Console.
Group Member Lister	Indicates whether or not you can use the WebLogic Portal Administration Console to search within a group for users or subgroups that match a given name pattern.
Member Group Lister	Indicates whether or not you can view groups in the WebLogic Portal Administration Console that directly contain a user or a group.
User Editor	Indicates whether or not you can modify group membership for users with the WebLogic Portal Administration Console.
User Lockout Manager	User lockout settings include how many unsuccessful login attempts a user can make before being prevented from future login attempts. For information about how to modify user lockout settings in the WebLogic Server Administration Console, see the Administration Console Online Help.
User Password Editor	Indicates whether or not you can modify user passwords in the WebLogic Portal Administration Console.
User Reader	Indicates whether or not you can view users in the WebLogic Portal Administration Console.
User Remover	Indicates whether or not you can delete users in the WebLogic Portal Administration Console.

To provide *write access* to external users and groups from the WebLogic Portal Administration Console, the authentication provider must be configured to allow it. This is a development task. For more information, see Configuring WebLogic Security Providers.

If an authentication provider does not provide *read access* to users and groups with the WebLogic Portal Administration Console, you can still use text entry fields to type in the names of existing users and groups for selection. For example, if you want to change the user profile property values for a user stored in a provider that does not support read access, you can type the name of the user in the Users Management tree to select the user for property modifications. For information about allowing text entry, see "Enabling Text Entry for Authentication Providers" on page 6-12.

For information on determining if you need to develop a custom authentication provider, and how to develop one, see How to Develop a Custom Authentication Provider. If you want to add an authentication provider, see "Choosing WebLogic and Custom Authentication Providers" on page 2-3.

Removing Authentication Providers

If you remove an authentication provider using the WebLogic Server Administration Console, be sure to also remove the provider from the WebLogic Portal Administration Console from the **Service Administration > Authentication Hierarchy Service** tree. For more information, see the User Management Guide.

Viewing Configured Role Mappers

A role mapping provider determines which security roles apply to operations performed on a resource.The default role mapping provider is the WebLogic XACML provider, `XACMLRoleMapper`, which uses the embedded LDAP server to store role policies.

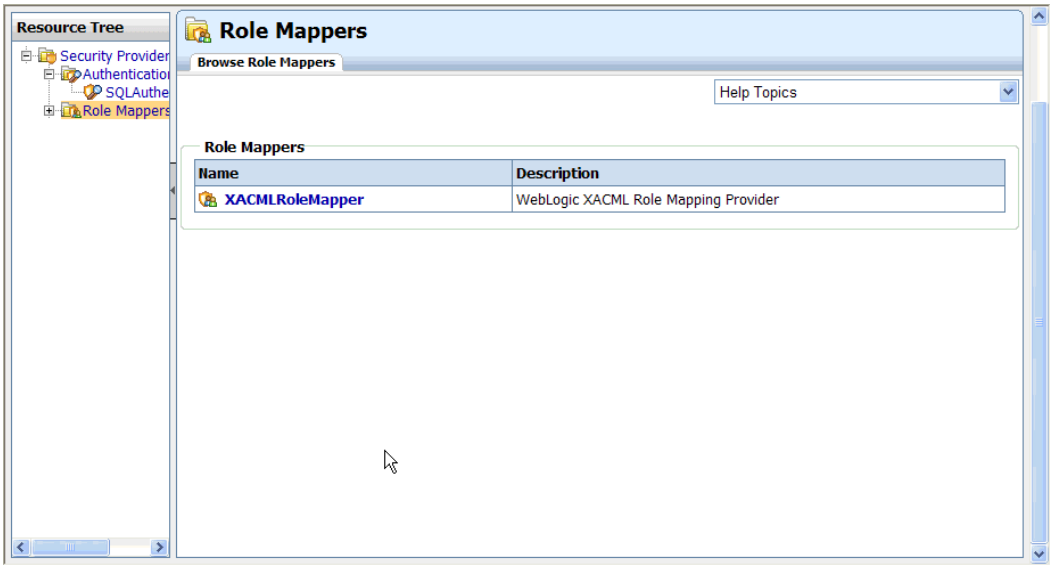
Note: The WebLogic XACML role mapping provider is required for WebLogic Portal, and suitable for most needs. It is unlikely that you will need to configure a custom role mapping provider.

Perform the following steps to view the configured role mappers:

1. Choose **Users, Groups, & Roles > Security Providers**.
2. Select **Role Mappers** in the Security Providers tree.

The Browse Role Mappers tab shows the title and description for each role mapper. The default role mapper, `XACMLRoleMapper`, is present by default, as shown in Figure 6-4.

Figure 6-4 Role Mappers



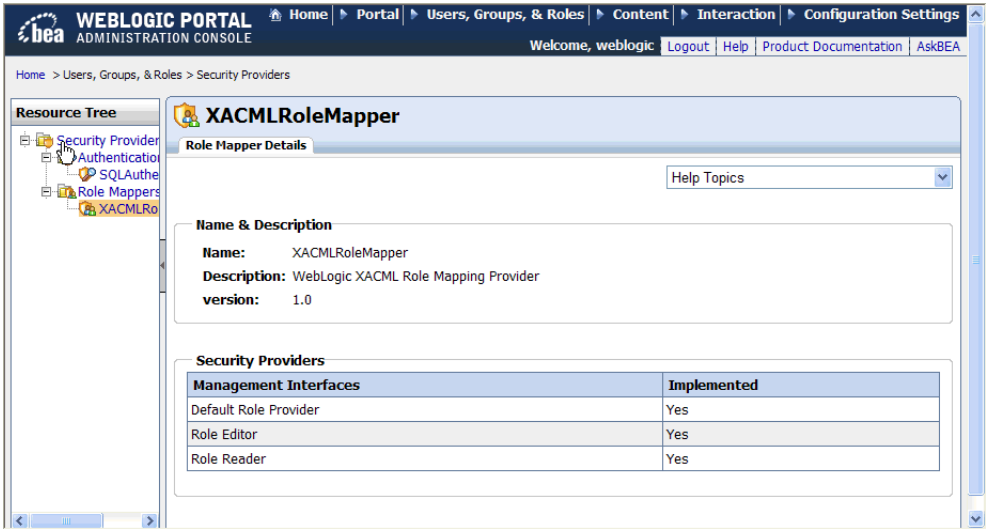
Viewing Role Mapper Details

Perform the following steps to view the details for a configured role mapper:

1. Choose **Users, Groups, & Roles > Security Providers**.
2. Select **Role Mappers** in the Security Providers tree.
3. Select the role mapper for which you would like to see details from the tab or from the tree.

The Role Mapper Details tab shows the name, description, and version of the role mapper. It also shows which management interfaces are implemented for the role mapper.

Figure 6-5 Role Mapper Details



Descriptions of the available management interfaces are listed in Table 6-2.

Table 6-2 Role Mapper Management Interfaces

Management Interface	Description
Default Role Provider	Indicates whether or not this was the first role mapper configured in WebLogic Server. The default does not change, regardless of which role mapper is currently being used.
Role Editor	Indicates whether or not you can modify role definitions in the WebLogic Portal Administration Console.
Role Reader	Indicates whether or not you can read roles in the WebLogic Portal Administration Console.

Viewing Authentication Provider Services

Perform the following steps to view the configured authentication provider services:

1. Choose **Configuration Settings > Service Administration**.

2. In the tree, select **Application Configuration Settings > Security > Authentication Security Provider Service**.

The Authentication Security Provider Service window shows the name and description for each service that has been configured. The `AllAtnProviders` service configuration settings apply to all authentication provider services, unless the settings are overridden for an individual authentication provider service.

Viewing Authentication Provider Service Details

Perform the following steps to view detailed information about a configured authentication provider service:

1. Choose **Configuration Settings > Service Administration**.
2. In the tree, select **Application Configuration Settings > Security > Authentication Security Provider Service**.
3. Select the authentication provider service for which you want to see detailed information. The `AllAtnProviders` service configuration settings apply to all authentication provider services; these settings can be overridden for an individual authentication provider service.

Detailed information about the selected authentication provider service is displayed.

The **Predicate Text Entry Enabled?** check box determines whether delegated administrators can add user, group, and role names to role and security policies by entering their names in a text box. For more information, see "Enabling Text Entry for Authentication Providers" on page 6-12.

You can also see which roles have the capability to create, read, update, or delete groups and users using the GroupProvider and UserProvider APIs.

The Anonymous role includes any unauthenticated user. The Self role is the logged in authenticated user, and indicates whether that user can perform operations for themselves, such as adding themselves to a group or changing their password.

Table 6-3 describes the group management and user management capabilities.

Table 6-3 Descriptions of Group Management and User Management Capabilities

Can Create	Determines whether the role can create groups or users, using the GroupProvider API or UserProvider API, respectively.
Can Read	Determines whether the role can see groups or users, using the GroupProvider API or UserProvider API, respectively.

Table 6-3 Descriptions of Group Management and User Management Capabilities

Can Update	Determines whether the role can update groups or users, using the GroupProvider API or UserProvider API, respectively.
Can Delete	Determines whether the role can delete groups or users, using the GroupProvider API or UserProvider API, respectively.

You can also restrict groups and users with specified names from being created or deleted.

Table 6-4 describes group and user naming restrictions you can set.

Table 6-4 Descriptions of Naming Restrictions

Protected	Determines whether a group or user with the specified name can be deleted.
Reserved	Determines whether a group or user with the specified name can be created.

From this window, you can add an authentication provider service to configure, as described next, or edit configuration settings, as described in "Configuring Authentication Provider Services" on page 6-12.

Adding Authentication Security Provider Services

You can add an existing authentication provider service so that you can view and edit its configuration settings in the **Service Administration** menu.

Perform the following steps to add an authentication security provider service:

1. Choose **Configuration Settings > Service Administration**.
2. Select **Security > Authentication Security Provider Service in the Application Configuration Settings** tree.
3. Click **Add Security Provider Service**.
4. In the **Add Authentication Security Provider Service to Security Service** dialog, select the name of the authentication provider from the drop-down list.
5. Optionally, add a description of the service.
6. Optionally, check the box to enable predicate text entry for the service. For more information, see "Enabling Text Entry for Authentication Providers" on page 6-12.

7. Click **Update**.

The service you have added appears in the list of services.

Updates to any of these settings require either enterprise application redeployment or server restart.

Configuring Authentication Provider Services

You can modify the following configuration settings for authentication provider services:

- Enable and disable text entry fields in the WebLogic Portal Administration Console for entering known user and group names in authentication providers that do not allow read access
- Specify which roles can be used in runtime operations using the GroupProvider and UserProvider APIs
- Prevent specific users or groups from being created or deleted

Enabling Text Entry for Authentication Providers

Some authentication providers may not allow read access to users and groups by external tools such as the WebLogic Portal Administration Console. If providers do not allow read access to users and groups, you can enable a text entry field that allows you to type in user and group names in the **User Management**, **Groups Management**, **Delegated Administration**, and **Visitor Entitlements** menus for those providers. By enabling text entry, you override the requirement that SSPI providers implement reader interfaces.

The text box, which appears in the tree section when text entry is enabled, allows you enter the names of known users and groups. You can assign profiles for those users or groups, and define delegated administration and visitor entitlements policies using those users and groups. When a user from a non-readable authentication provider logs in, the profile created for that user enables authorization checks to be performed for the user.

To enable text entry for an authentication security provider service:

1. Choose **Configuration Settings > Service Administration**.
2. In the tree, select **Application Configuration Settings > Security > Authentication Security Provider Service**.
3. Select the authentication provider service for which you want to see detailed information.

4. Click the **Edit** icon next to **Configuration Settings for: *ServiceName***.
5. Select the **Predicate Text Entry Enabled?** check box.

This change requires either enterprise application redeployment or server restart.

Adding Group Management Roles

When you add a group management role to an authentication provider service, you enable capabilities for manual runtime checks performed by API calls to group providers. This provides a low-level alternative to using visitor entitlements on groups. For each group management capability (create, read, update, and delete), you can specify which roles are allowed to perform the task.

Note: Use existing global or enterprise-application scoped roles.

Perform the following steps to add role capabilities for the GroupProvider API:

1. Choose **Configuration Settings > Service Administration**.
2. In the tree, select **Application Configuration Settings > Security > Authentication Security Provider Service**.
3. Select the authentication provider service for which you want to add a group management role.
4. In the Group Management Delegated Administration section, click **Add Group Management Rol**e.
5. Enter a role name. Use existing global or enterprise-application scoped role names.
6. Select the capabilities for the role, as described in Table 6-3, “Descriptions of Group Management and User Management Capabilities,” on page 6-10.
7. Click **Update**.

The new role appears in the list of Group Management Roles. This change requires either enterprise application redeployment or server restart.

Editing Group Management Roles

Group Management role capabilities are used for manual runtime checks performed by API calls to group providers. This provides a low-level alternative to using visitor entitlements on groups.

Perform the following steps to edit group management role capabilities for the GroupProvider API:

1. Choose **Configuration Settings > Service Administration**.
2. In the tree, select **Application Configuration Settings > Security > Authentication Security Provider Service**.
3. Select the authentication provider service for which you want to edit a group management role.
4. Click the role name or the **Edit** icon for the role you want to edit.
5. Select the capabilities for the role, as described in Table 6-3, "Descriptions of Group Management and User Management Capabilities," on page 6-10.
6. Click **Update**.

The updated role appears in the list of Group Management Roles. This change requires either enterprise application redeployment or server restart.

Adding User Management Roles

When you add a user management role to an authentication provider service, you enable capabilities for manual runtime checks performed by API calls to user providers. For each user management capability (create, read, update, and delete), you can specify which roles are allowed to perform the task.

Note: Use existing global or enterprise-application scoped roles.

Perform the following steps to add user management role capabilities for the UserProvider API:

1. Choose **Configuration Settings > Service Administration**.
2. In the tree, select **Application Configuration Settings > Security > Authentication Security Provider Service**.
3. Select the authentication provider service for which you want to add a user management role.
4. In the User Management Delegated Administration section, click **Add User Management Rol**e.
5. Enter a role name. Use existing global or enterprise-application scoped role names.
6. Select the capabilities for the role, as described in Table 6-3, "Descriptions of Group Management and User Management Capabilities," on page 6-10.

7. Click **Update**.

The new role appears in the list of User Management Roles. This change requires either enterprise application redeployment or server restart.

Editing User Management Roles

User Management role capabilities are used for manual runtime checks performed by API calls to group user.

Perform the following steps to edit user management role capabilities for the UserProvider API:

1. Choose **Configuration Settings > Service Administration**.
2. In the tree, select **Application Configuration Settings > Security > Authentication Security Provider Service**.
3. Select the authentication provider service for which you want to edit a user management role.
4. Click the role name or the **Edit** icon for the role you want to edit.
5. Select the capabilities for the role, as described in Table 6-3, “Descriptions of Group Management and User Management Capabilities,” on page 6-10.
6. Click **Update**.

The updated role appears in the list of User Management Roles. This change requires either enterprise application redeployment or server restart.

Adding Protected and Reserved Group Names

For each authentication provider, you can specify group names that cannot be created or deleted.

Perform the following steps to set restrictions on group names:

1. Choose **Configuration Settings > Service Administration**.
2. In the tree, select **Application Configuration Settings > Security > Authentication Security Provider Service**.
3. Select the authentication provider service for which you want to restrict group names.
4. In the Protected/Reserved Groups section, click **Add Protected/Reserved Group**.
5. Enter a group name.

6. Select the **Protected** check box if you want to prevent a group with this name from being deleted.
7. Select the **Reserved** check box if you want to prevent a group with this name from being created.
8. Click **Update**.

The group name role appears in the list of Protected/Reserved Groups. This change requires either enterprise application redeployment or server restart.

Editing Protected and Reserved Group Names

Perform the following steps to edit the restrictions for group names that are in the list of Protected/Reserved Groups:

1. Choose **Configuration Settings > Service Administration**.
2. In the tree, select **Application Configuration Settings > Security > Authentication Security Provider Service**.
3. Select the authentication provider service for which you want to change restrictions on group names.
4. In the Protected/Reserved Groups section, click a group name or the **Edit** icon for that group.
5. Select the **Protected** check box if you want to prevent a group with this name from being deleted.
6. Select the **Reserved** check box if you want to prevent a group with this name from being created.
7. Click **Update**.

The new restrictions for this group name role appears in the list of Protected/Reserved Groups. This change requires either enterprise application redeployment or server restart.

Adding Protected and Reserved User Names

For each authentication provider, you can specify user names that cannot be created or deleted.

Perform the following steps to set restrictions on user names:

1. Choose **Configuration Settings > Service Administration**.

2. In the tree, select **Application Configuration Settings > Security > Authentication Security Provider Service**.
3. Select the authentication provider service for which you want to restrict user names.
4. In the Protected/Reserved Users section, click **Add Protected/Reserved User**.
5. Enter a user name.
6. Select the **Protected** check box if you want to prevent a user with this name from being deleted.
7. Select the **Reserved** check box if you want to prevent a user with this name from being created.
8. Click **Update**.

The user name role appears in the list of Protected/Reserved Users. This change requires either enterprise application redeployment or server restart.

Editing Protected and Reserved User Names

Perform the following steps to edit the restrictions for user names that are in the list of Protected/Reserved Users:

1. Choose **Configuration Settings > Service Administration**.
2. In the tree, select **Application Configuration Settings > Security > Authentication Security Provider Service**.
3. Select the authentication provider service for which you want to change restrictions on user names.
4. In the Protected/Reserved Users section, click a user name or the **Edit** icon for that user.
5. Select the **Protected** check box if you want to prevent a user with this name from being deleted.
6. Select the **Reserved** check box if you want to prevent a user with this name from being created.
7. Click **Update**.

The new restriction for this user name appears in the list of Protected/Reserved Users. This change requires either enterprise application redeployment or server restart.

Viewing Role Provider Services

Perform the following steps to view the configured role provider services:

1. Choose **Configuration Settings > Service Administration**.
2. In the tree, select **Application Configuration Settings > Security > Role Security Provider Service**.

The Role Security Provider Service window shows the name and description for each service that has been configured. The `AllRolePrividers` service configuration settings apply to all role mapping provider services, unless the settings are overridden for an individual role provider service.

The default role mapping provider is the WebLogic XACML provider, `XACMLRoleMapper`, which uses the embedded LDAP server to store role policies.

Note: The WebLogic XACML role mapping provider is required for WebLogic Portal, and suitable for most needs. It is unlikely that you will need to configure a custom role mapping provider.

Viewing Role Provider Service Details

Perform the following steps to view detailed information about a configured role provider service:

1. Choose **Configuration Settings > Service Administration**.
2. In the tree, select **Application Configuration Settings > Security > Role Security Provider Service**.
3. Select the role provider service for which you want to see detailed information. The `AllRoleProviders` service configuration settings apply to all role provider services, unless the settings are overridden for an individual role provider service.

The **Predicate Text Entry Enabled?** capability determines whether delegated administrators can add user, group, and role name predicates to role and security policies by entering their names in a text box. For more information, see "Enabling Text Entry for a Role Mapping Providers" on page 6-19.

Adding Role Mapping Provider Services

You can add an existing role provider service so that you can view and edit the configuration settings in the **Service Administration** menu.

Perform the following steps to add a role security provider service:

1. Choose **Configuration Settings > Service Administration**.
2. Select **Security > Role Security Provider Service** in the Application Configuration Settings tree.
3. Click **Add Security Provider Service**.
4. Select the name of the role mapping provider from the drop-down list.
5. Optionally, add a description of the service.
6. Optionally, check the box to enable predicate text entry. For more information, see "Enabling Text Entry for a Role Mapping Providers" on page 6-19.
7. Click **Update**.

The service you have added appears in the list of services.

Updates to any of these settings require either enterprise application redeployment or server restart.

Configuring Role Mapping Provider Services

The default role mapping provider is the WebLogic XACML role mapping provider, which uses the embedded LDAP server to store role policies. The WebLogic XACML role mapping provider is required for WebLogic Portal, and suitable for most needs. It is unlikely that you will need to configure a custom role mapping provider.

Enabling Text Entry for a Role Mapping Providers

Some role providers may not allow read access to role policies by external tools such as the WebLogic Portal Administration Console. If providers do not allow read access to roles, you can enable a text entry field that allows you to type in role names in the **Delegated Administration** and **Visitor Entitlements** menus for those providers. By enabling text entry, you override the requirement that SSPI providers implement reader interfaces.

The text box, which appears in the menu tree when text entry is enabled, allows you enter the names of known roles. You can define delegated administration and visitor entitlements policies using these role names.

To enable text entry for a role security provider service:

1. Choose **Configuration Settings > Service Administration**.
2. In the tree, select **Application Configuration Settings > Security > Role Security Provider Service**.
3. Select the role provider service for which you want to see detailed information.
4. Click the **Edit** icon next to **Configuration Settings for: *ServiceName***.
5. Select the **Predicate Text Entry Enabled?** check box.

This change requires either enterprise application redeployment or server restart.

CHAPTER 7

Configuring Delegated Administration

This chapter provides an overview of how to configure delegated administration using the WebLogic Portal Administration Console. Delegated administration provides a mechanism for propagating WebLogic Portal Administration Console privileges within a hierarchy of roles.

In your organization, you typically want individuals to have different access privileges to various administration tasks and resources. For example, a system administrator might have access to every feature in the WebLogic Portal Administration Console. The system administrator might then create a portal administrator role that can manage instances of portal resources in specific desktop views of your portal, and a library administrator role that can manage your portal resource library.

WebLogic Portal has one predefined delegated administration role, `PortalSystemDelegator`. By default, all members of the `Administrators` group are assigned the `PortalSystemDelegator` role. Anyone assigned the `PortalSystemDelegator` role has unlimited access to administrative tasks anywhere in the enterprise portal application. Other delegated administration roles only have access to resources if that access has been explicitly granted.

You can create as many different administrators as you need by creating administrator roles and then assigning role membership dynamically, based on username, group membership, user profile property values, session and request attributes, and date and time functions.

You can use delegated administration to propagate access privileges within a hierarchy of roles that define the structure for delegated administration. You have flexibility in the way you set up your administration hierarchy and assign privileges to your administrators. Given the appropriate privileges, administrators can delegate both the privilege to administer a given resource

capability and the privilege for the delegatee to delegate further. For additional information on role hierarchies, see "Setting Up a Delegated Administration Role Hierarchy" on page 2-10.

This chapter includes the following sections:

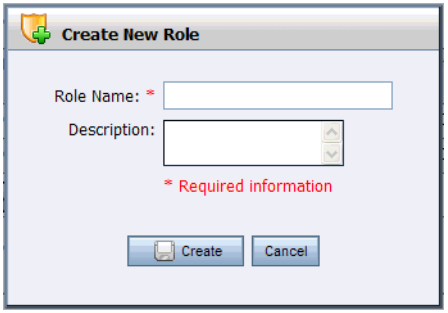
- Creating Delegated Administration Roles
- Adding Users, Groups, and Conditions in Delegated Administration Roles
- Removing Users, Groups, and Conditions from Delegated Administration Roles
- Modifying Conditions in Delegated Administration Roles
- Granting Additional Delegation Properties to Roles
- Viewing Delegated Administration Role Details
- Viewing the Delegated Resources
- Renaming Delegated Administration Roles
- Deleting Delegated Administration Roles
- Setting Delegated Administration on Authentication Providers
- Removing Delegated Administration on Authentication Providers
- Setting Delegated Administration on Groups
- Removing and Editing Delegated Administration on Groups
- Setting Delegated Administration on Portal Resources in the Library
- Setting Delegated Administration on Portal Resources in the Desktop
- Removing and Editing Delegated Administration on Portal Resources
- Setting Delegated Administration on Interaction Management Resources
- Removing Delegated Administration on Interaction Management Resources
- Setting Delegated Administration on Content Management Resources
- Removing and Editing Delegated Administration on Content Management Resources
- Setting Delegated Administration on Visitor Entitlement Roles
- Removing Delegated Administration from Visitor Entitlement Roles

Creating Delegated Administration Roles

Perform the following steps to create a new delegated administration role:

1. Choose **Users, Groups, & Roles > Delegated Administration**.
2. In the Delegated Administration tree, select the parent role for which you want to create a new child role. The `PortalSystemDelegator` role is the top level parent role, and exists before any other delegated administration roles have been created.
3. From the Browse Roles tab, click **Create New Role**.
4. In the dialog box that appears, enter the name of the new role, and optionally, a description, and click **Create**.

Figure 7-1 Create New Role Dialog



The new delegated administration role appears in the resource tree.

You can now define the role by adding users to the role, adding groups to the role, or using expressions. For more information, see "Adding Users, Groups, and Conditions in Delegated Administration Roles" on page 7-4.

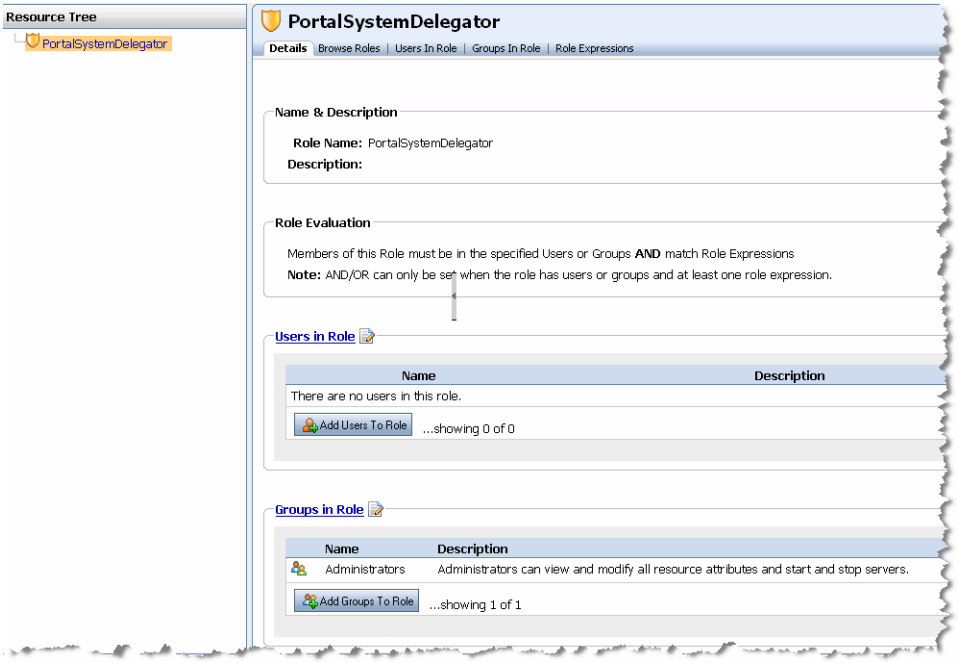
Note: When you are establishing your delegated administration role hierarchy, child role names must be unique. For example, you cannot have a delegated administration role called `RoleA` with a child role of `RoleB` if you already have a child role called `RoleB` elsewhere in the hierarchy.

Adding Users, Groups, and Conditions in Delegated Administration Roles

Once you create delegated administration roles in the WebLogic Portal Administration Console, you can assign users and groups to them. You can also use expressions, based on use profile properties, dates, and times, to determine who is assigned a delegated administration role.

Figure 7-2 shows the Details tab for the `PortalSystemDelegator` role.

Figure 7-2 PortalSystemDelegator Role - Details Tab



Adding Users to Delegated Administration Roles

When you add a user to a role, you grant that user access the administrative privileges attributed to that role. This section describes how to add one or more users to a role.

If you have a large number of users you want to add to a role, for the best performance add users to groups, then create roles with those groups, or use expressions.

Tip: Roles can sometimes be mapped directly to groups. The difference between groups and roles is that group membership is statically assigned by a server administrator, while role membership is dynamically determined based on information including the username, user profile property values, group membership, and dates and times. Roles can also be scoped to specific WebLogic resources within a single application in a WebLogic Server domain, while groups are always scoped to an entire WebLogic Server domain.

Perform the following steps to add one or more users to a delegated administration role:

1. Choose **Users, Groups, & Roles > Delegated Administration**.
2. In the Delegated Administration tree, select the role for which you want to add users.
3. Select the Users in Role tab.
4. Click **Add Users To Role**.
5. If necessary, find the users you want to add to the role using the Search feature. Users appear in the Search Results section.

   Tip: If you are using an SQL authentication provider, be aware that user names are case sensitive. For example, user `Bob` is different than user `bob`.

6. Select the check box next to each user you want to add, and click **Add**. Selected users now appear in the Users to Add section.
7. Click **Save**.

Any users you have added now appear in the Users in Role section of the Details and Users in Role tabs.

Adding Groups to Delegated Administration Roles

When you add a group to a role, you grant the members (users) of that group—and users in any sub-groups of that group—access to the administrative privileges attributed to that role.

Perform the following steps to add a group to a delegated administration role:

1. Choose **Users, Groups, & Roles > Delegated Administration**.
2. In the Delegated Administration tree, select the role for which you want to add groups.

3. Select the Groups in Role tab.
4. Click **Add Groups To Role**.
5. If necessary, find the groups you want to add to the role using the Search feature. Groups appear in the Search Results section.

   Tip: If you are using an SQL authentication provider, be aware that group names are case sensitive. For example, group `Managers` is different than group `managers`.

6. Select the check box next to each group you want to add, and click **Add**. Selected groups now appear in the Groups to Add section.

   Note: If a list of groups is not displayed, make sure you have built a group hierarchy tree for the authentication provider. If you do not see a list of groups after building a group hierarchy tree, the authentication provider might not allow read access. To see if your authentication provider allows read access, view the authentication provider details, as described in "Viewing Authentication Provider Details" on page 6-4.

   You can activate a text field for group name entry for authentication providers that do not allow read access, as described in "Enabling Text Entry for Authentication Providers" on page 6-12.

Any groups you have added now appear in the Groups in Role section in the Details and Groups in Role tabs.

Adding Conditions to Delegated Administration Roles with Expressions

You can use expressions to set conditions, in addition to username and group membership, that dynamically determine membership in a delegated administration role. Conditions specify the values of user profile properties, session and request attributes, dates, and times.

For example, you can define a role with the following type of expression: If a logged-in user has the `administrator` property set to `true` and the time is between 9 a.m. and 5 p.m. PST, the user is a role member.

Perform the following steps to add conditions to a delegated administration role:

1. Choose **Users, Groups, & Roles > Delegated Administration**.
2. In the Delegated Administration tree, select the role to which you want to add conditions.

3. Select the Role Expression tab.
4. In the top left corner of the tab, ANY or ALL is underlined. By selecting, you can toggle between these values.
5. For each expression you want to create, click **Add Condition**. When you select a condition, it expands to let you specify the value. You can create an expression from a drop-down list containing the following options:
   - The date is:

     Specify a date using the calendar.
   - It is after a given date:

     Specify a date using the calendar.
   - It is after a given date and time:

     Specify a date and time using the calendar.
   - It is between two times:

     Specify a time range using the calendars.
   - It is between two dates:

     Specify a date range using the calendars.
   - It is between two date/times:

     Specify a range of dates and times using the calendars.
   - The visitor, visitor's HTTP request, or visitor's HTTP session has characteristics:

     To set characteristics, you must specify a Property Set, a Property from the property set, a Value for the property, and the ANY or ALL comparator. Specify a property value from the pull-down menu. You can click **Add Another Value** to add multiple properties and corresponding values.
   - The consumer's registration has these values:

     Specify WSRP registration properties. For more information, see the Federation Guide.

   Tip: User profile properties, HTTP session and request properties, and WSRP registration properties are created by developers in WorkSpace Studio.
6. Click **Save** to apply the conditions.

Note: If you define roles with expressions whose evaluation changes during the processing of a request, you may need to adjust your portal application cache settings to ensure that the correct role definition is retrieved instead of a cached role.

Removing Users, Groups, and Conditions from Delegated Administration Roles

You can change who is assigned a role by removing users, groups, and conditions from delegated administration roles.

Removing Users from Delegated Administration Roles

If you want to revoke user access to administrative privileges associated with a role, you can remove the user from the role.

Perform the following steps to remove one or more users from a delegated administration role:

1. Choose **Users, Groups, & Roles > Delegated Administration**.
2. In the Delegated Administration tree, select the role from which you want to remove users.
3. Select the Users in Role tab.
4. In the list of users, select the check box in the Remove section next to each user you want to remove. By selecting the check box in the header above the user names, you can remove all users from the role.
5. Click **Remove**.

Users you have removed no longer appear in the Users in Role tab or in the Details tab under Users in Role.

Removing Groups from Delegated Administration Roles

Perform the following steps to remove one or more groups from a role:

1. Choose **Users, Groups, & Roles > Delegated Administration**.
2. In the Delegated Administration tree, select the role from which you want to remove groups.
3. Select the Groups in Role tab.

4. In the list of groups, select the check box in the Remove section next to each group you want to remove. By selecting the check box in the header above the group names, you can remove all groups from the role.
5. Click **Remove**.

Groups you have removed no longer appear in the Groups in Role tab or in the Details tab under Groups in Role.

Removing Conditions in Delegated Administration Roles

Perform the following steps to remove one or more conditions from a role:

1. Choose **Users, Groups, & Roles > Delegated Administration**.
2. In the Delegated Administration tree, select the role from which you want to remove conditions.
3. Select the Expressions in Role tab.
4. In the list of conditions, select the check box in the Delete section next to each one you want to remove. By selecting the check box in the header above the conditions, you can remove all conditions from the role.
5. Click **Delete**.

Conditions you have removed no longer appear in the Role Expressions tab or in the Expressions in Role section of the Details tab.

Modifying Conditions in Delegated Administration Roles

You can modify an existing expression in a delegated administration role, as long as you do not want to change the type of condition. For example, if you created a condition based on a date range, you can change the dates.

You can also add a condition from the Role Expressions tab; see "Adding Conditions to Delegated Administration Roles with Expressions" on page 7-6 for more information. To remove a condition, see "Removing Conditions in Delegated Administration Roles" on page 7-9.

Perform the following steps to modify a role condition:

1. Choose **Users, Groups, & Roles > Delegated Administration**.
2. In the Delegated Administration tree, select the role for which you want to modify a condition.

3. Select the Role Expressions tab.
4. Click **Edit** for the condition you want to modify.
5. Specify the new value or values for the condition.
6. Click **Save**.

The modified condition appears in the list of conditions.

Granting Additional Delegation Properties to Roles

You can allow a selected role to manage sub-roles. For example, you can allow another administrator to create child roles, delete them, move them, and add users to them in the selected role. Each role in the node can be granted the privileges to edit its subordinate roles.

Note: You can modify those roles that are below you in the administrator hierarchy. You cannot modify your own role or any roles above you in the hierarchy.

Perform the following steps to grant delegation authority to an existing role:

1. Choose **Users, Groups, & Roles > Delegated Administration**.
2. In the Delegated Administration tree, select the role for which you want to view or modify delegation properties.
3. From the Details tab, select **Advanced Options**, or click the **Edit** icon next to it.
4. To allow any user or group assigned this role to manage sub-roles, select the check box **Administrators in this Role can manage child Roles**.

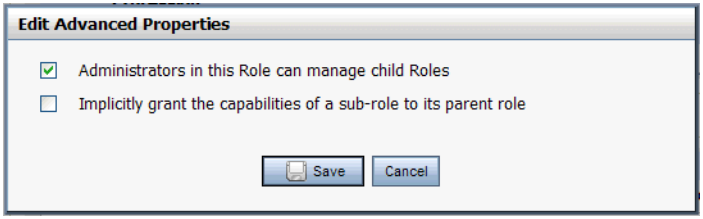
Tip: If you want parent roles to automatically be granted any capabilities granted to a child role, select the check box **Implicitly grant the capabilities of a sub-role to its parent role**. This option is displayed only when the `PortalSystemDelegator` role is selected in the tree, because it is applied globally to all roles.

---

5. Click **Save**.

Figure 7-3 shows the Advanced Properties dialog.

Figure 7-3 Advanced Properties Dialog



Viewing Delegated Administration Role Details

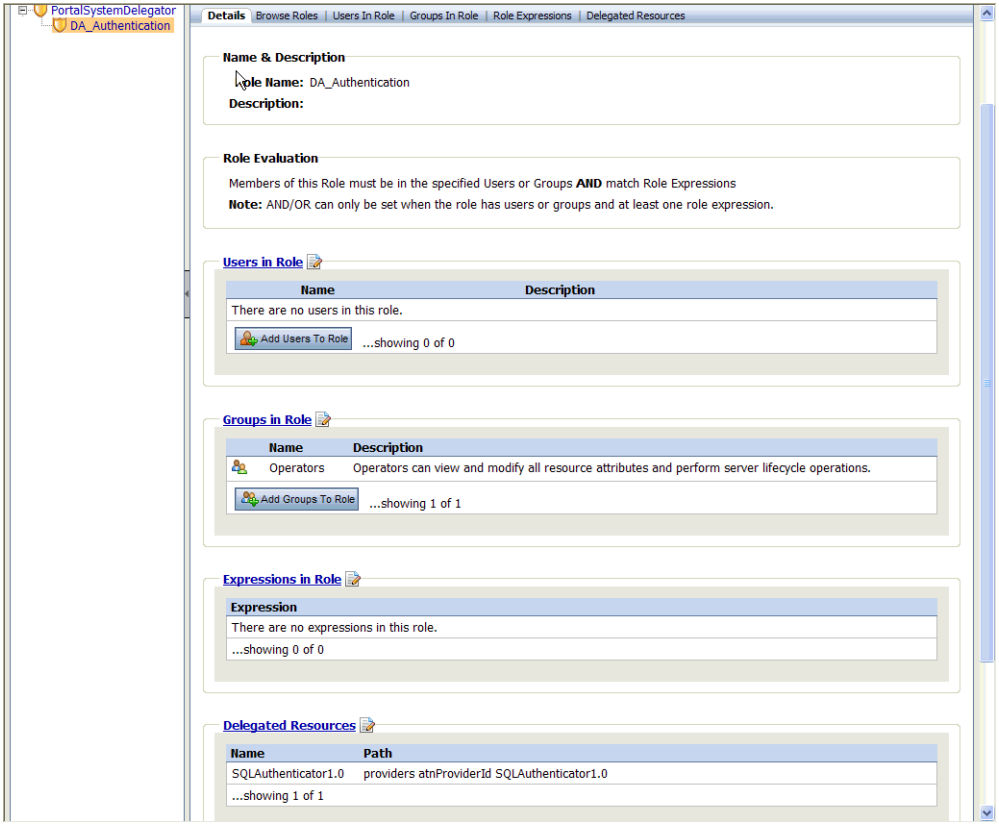
Once you have created a role, you can select it in the Delegated Administration tree.

Perform the following steps to view the details about a delegated administration role:

1. Choose **Users, Groups, & Roles > Delegated Administration**.
2. In the Delegated Administration tree, select the role for which you want to see detailed information.
3. Select the Details tab.

Figure 7-4 shows the Details tab for the `DA_Authentication` role.

Figure 7-4 Delegated Administration Details Tab



Viewing the Delegated Resources

You can view the portal resources that are associated with a role. This is useful because before you can delete a delegated administration role, you have to remove the security policies associated with the role. A security policy is created when you define an association between a WebLogic resource and one or more users, groups, or roles. Hence, a role policy defines a role and a security policy defines an authorization constraint associated with that role.

Tip: You can delete security policies from the policy summary page, or from the Delegated Admin tab for the specific resource.

Perform the following steps to view a delegated administration role's delegated resources.

1. Choose **Users, Groups, & Roles > Delegated Administration**.
2. In the Delegated Administration tree, select a role.
3. Select the Delegated Resources tab.

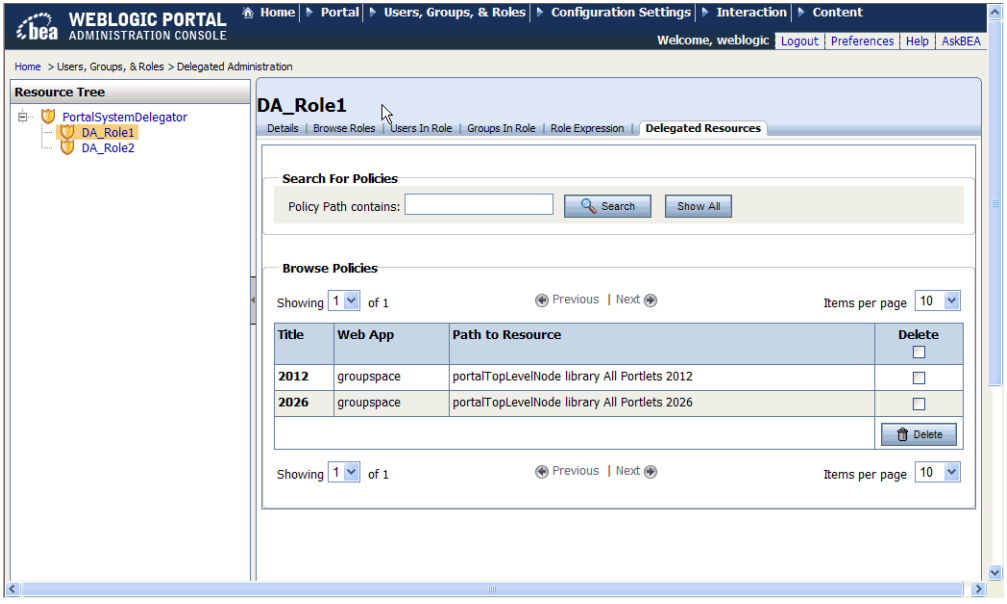
There you can view the information for the role's delegated resources:

- Title of the resource
- Web application name
- Path to the resource

Tip: From this tab, you can delete a security policy by selecting the check box in the Delete column and clicking **Delete**.

Figure 7-5 shows the Delegated Resources tab.

Figure 7-5 Delegated Resources Tab



Renaming Delegated Administration Roles

You can change the name and description of existing delegated administration role if there are no security policies associated with the role. For information about viewing the policies associated with a role, see "Viewing the Delegated Resources" on page 7-12.

Tip: If there are policies associated with a role, it does not appear as editable in the Details tab.

Perform the following steps to rename a delegated administration role:

1. Choose **Users, Groups, & Roles > Delegated Administration**.
2. In the Delegated Administration tree, select the role you want to rename.
3. From the Details tab, select **Name & Description**, or click the **Edit** icon next to it.
4. In the dialog box that appears, type the new name, and optionally, a new description, and click **OK**.

The new role name appears in the Delegated Administration tree and the tabs.

Deleting Delegated Administration Roles

When you delete a delegated administration role, the child roles associated with it are also deleted.

A delegated administration role can only be deleted when no security policies are associated with it. If there are security policies associated with a role you are trying to delete, a warning is displayed. You must delete all such references before you can delete the role.

Perform the following steps to delete a delegated administration role:

1. Choose **Users, Groups, & Roles > Delegated Administration**.
2. In the Delegated Administration tree, select the *parent role* for the role want to delete.
3. In the list of roles, select the check box next to any roles you want to delete.
4. Click **Delete**.

If you receive a message that the role cannot be deleted while resource dependencies exist, select the Delegated Resources tab for that role to view, and optionally delete, the resource dependencies. For more information, see "Viewing the Delegated Resources" on page 7-12.

Setting Delegated Administration on Authentication Providers

You can determine which portal administrators can manage each authentication provider by assigning delegated administration roles to the provider.

The only capability that can be specified for an authentication provider is Can Use. This allows you to manage users and groups from this authentication provider.

Note: If you attempt to assign a delegated administration role to a group as described in "Setting Delegated Administration on Groups" on page 7-16, and you do not have Can Use capability, a dialog box asks if you would like to grant access to the provider as well. If you click **OK**, you provide access to the authentication provider and assign the delegated administration role to the group. If you click **Cancel**, the role is not allowed access to the authentication provider.

Perform the following steps to assign delegated administration to an authentication provider:

1. Choose **Users, Groups, & Roles >Security Providers**.
2. In the Security Providers tree, navigate to the provider for which you want to set delegated administration.
3. Select the Edit Delegated Admin tab.
4. Click **Add Role**.
5. In the list of roles in the Search Results section, select the check box next to any roles you want to add and click **Add**. The selected roles are added to the Roles to Add section.

   You can remove a role from the Roles to Add section by selecting the check box next to the role and clicking **Remove Selected**.
6. Click **Save**.
7. In the Delegate Capabilities to Resource dialog, select the check box for the Can Use capability. By selecting the check box in the header above the role names, you enable the Can Use capability for all roles.
8. Click **Save**.

The roles you have added are listed in the Browse Roles Delegated to this Resource section.

Removing Delegated Administration on Authentication Providers

If you no longer want administrator capabilities to be available for an authentication provider, you can remove administrator capabilities from it.

The only administrator capability for authentication providers is Can Use, so if you edit the role to remove this capability, the delegated administration role is removed from the authentication provider.

Perform the following steps to remove delegated administration on an authentication provider:

1. Choose **Users, Groups, & Roles >Security Providers**.
2. In the Security Providers tree, navigate to the provider for which you want to remove delegated administration.
3. Select the Edit Delegated Admin tab.
4. Under Browse Roles Delegated to this Resource, select the check box in the Remove Role column for each role you want to remove. By selecting the check box in the header above the role names, you remove the Can Use capability from all roles.
5. Click **Remove**.

The changes you make are reflected in the Browse Roles Delegated to this Resource section.

Setting Delegated Administration on Groups

You can determine which portal administrators can manage each group by assigning delegated administration roles to the group. Table 7-1 describes administrator capabilities for groups.

Table 7-1 Descriptions of Administrator Capabilities for Groups

Profile Admin	Determines whether the administrator can edit profile properties in the Group Profile tab and the User Profile tab for users in that group.
Read User/Group	Determines whether the administrator can view information about the group and users in the group.
Create Update Delete User/Group	Determines whether the administrator can manage the group and users in the group (including adding users to groups).

Tip: If you are using more than one authentication provider, it is possible to have a group in one provider with an identical name to a group in another provider. When you set delegated administration on a group, an administrator in that delegated administration role is able to administer that group in all providers that contain that group, if the administrator also has administrator capabilities for the other providers.

Perform the following steps to assign delegated administration to a group:

1. Choose **Users, Groups, & Roles > Group Management**.
2. In the Groups tree, select the group for which you want to set delegated administration.

   Note: If a list of groups is not displayed, make sure you have built a group hierarchy tree for the authentication provider. If you do not see a list of groups after building a group hierarchy tree, the authentication provider might not allow read access. To see if your authentication provider allows read access, view the authentication provider details, as described in "Viewing Authentication Provider Details" on page 6-4.

   You can activate a text field for group name entry for authentication providers that do not allow read access, as described in "Enabling Text Entry for Authentication Providers" on page 6-12.
3. Select the Delegated Admin tab.
4. Click **Add Role**.
5. In the list of roles in the Search Results section, select the check box next to any roles you want to add and click **Add**. The selected roles are added to the Roles to Add section.

   You can remove a role from the Roles to Add section by selecting the check box next to the role and clicking **Remove Selected**.
6. Click **Save**.

   Note: Roles that are allowed to administer groups must also have Can Use capability to access the authentication provider. If the delegated administration role you are assigning to the group does not have access to the authentication provider, a dialog box asks if you would like to grant access to the provider as well. Click **OK** to provide access to the authentication provider and assign the delegated administration role to the group. Click **Cancel** if you do not want the role to have access to the authentication provider. The delegated administration role is not assigned to the group if you click **Cancel**.

7. In the Delegate Capabilities to Resource dialog, select the check boxes for the capabilities listed in Table 7-1, "Descriptions of Administrator Capabilities for Groups," on page 7-16. By selecting the check box in the header above the role names, you enable that capability for all roles.
8. Click **Save**.

The roles you have added are listed in the Browse Roles Delegated to this Resource section.

Removing and Editing Delegated Administration on Groups

If you no longer want administrator capabilities to be available for a group of users, you can remove delegated administration from the group. You can also change the capabilities of a delegated administration role on a group, which is also described in this procedure.

Tip: You can also remove a delegated administration role from a group from the Delegated Resources tab for that role. From this tab, delete a security policy by selecting the check box in the Delete column and clicking **Delete**.

Perform the following steps to remove or edit delegated administration on a group:

1. Choose **Users, Groups, & Roles > Group Management**.
2. In the Groups tree, select the group for which you want to remove or edit delegated administration.

   Note: If a list of groups is not displayed, make sure you have built a group hierarchy tree for the authentication provider. If you do not see a list of groups after building a group hierarchy tree, the authentication provider might not allow read access. To see if your authentication provider allows read access, view the authentication provider details, as described in "Viewing Authentication Provider Details" on page 6-4.

   You can activate a text field for group name entry for authentication providers that do not allow read access, as described in "Enabling Text Entry for Authentication Providers" on page 6-12.
3. Select the Delegated Admin tab.
4. From the Browse Roles Delegated to this Resource section:

- To remove the delegated administration role from the group:

a. Select the check box in the Remove Role column for each role you want to remove. By selecting the check box in the header above the role names, you can remove that capability from all roles in that column.

b. Click **Remove**.

- To edit the capabilities of the delegated administration role on the group:

a. Select the check box in the Edit Capabilities column for each role you want to change capability for.

b. Click **Edit**.

c. In the Delegate Capabilities to Resource dialog, select the check boxes for the capabilities you want each role to have (see Table 7-1, "Descriptions of Administrator Capabilities for Groups," on page 7-16). By selecting the check box in the header above the role names, you enable that capability for all roles.

d. Click **Save**.

The changes you make are reflected in the Browse Roles Delegated to this Resource section.

Setting Delegated Administration on Portal Resources in the Library

Security policies determine what capabilities a delegated administration role has for a given portal resource. You can set delegated administration on portal resources in the resource library or in the desktop (Portals node). Within the library, you can set administrator capabilities on specific books, pages, and portlets, or all resources in each of these categories.

You can control administrator access to the following types of portal resources in the library:

- Library
- Portlets
- Portlet categories
- Books
- Layouts
- Look and feels
- Menus

- Pages
- Shells
- Templates
- Themes

Each has administrator capabilities that are based on the type of resource, as shown in Table 7 -2.

Table 7 -2 Administrator Capabilities According to Portal Resource Type in the Library

	Manage Definition	Create/Remove Instances	Use Definition	View Templates
Library	✓	✓	✓	
Portlet	✓	✓		
Portlet Category	✓	✓		
Book	✓	✓		
Layout	✓		✓	
Look and Feel	✓		✓	
Menu	✓		✓	
Page	✓	✓		
Shell	✓		✓	

Table 7 -2 Administrator Capabilities According to Portal Resource Type in the Library

	Manage Definition	Create/Remove Instances	Use Definition	View Templates
Template (Community and Desktop)				✔
Theme	✔		✔	

Table 7-3 describes each administrator capability.

Table 7-3 Descriptions of Administrator Capabilities for Portal Resources in the Library

Manage Definition	Determines who can edit resource definitions that are propagated globally throughout all instances of this resource. For example, an administrator who has Manage Definition capability on a page can edit the contents of the page (including portlets), the position of the portlets on the page, or the entitlements for the page, and those changes are applied to every instance of that page. The page is then available as a template that other administrators can use to create new pages. An administrator with this capability can also create and remove instances of this resource.
Create/Remove Instances	Determines whether the administrator can make these portal resources available to specific portals. Once you create a resource in the library to use as a template for other pages, you must make that resource available for portal administrators to select from the list of resources available for their portlet. For example, an administrator can create a page template in the library, then make that page available for specific portals to use as resources.
Use Definition	Determines whether the administrator can use the resource. If this is the only capability granted, the administrator cannot modify the resource, and the resource is not displayed in the Library tree. Resources that can be used with desktops, books and pages can be given Use Definition capabilities. For example, if an administrator has Use Definition capability for a layout, they can only use that layout within the context of the page that they are allowed to manage.
View Templates	Only applicable to templates; determines if the administrator can create a new desktop or community based on a template.

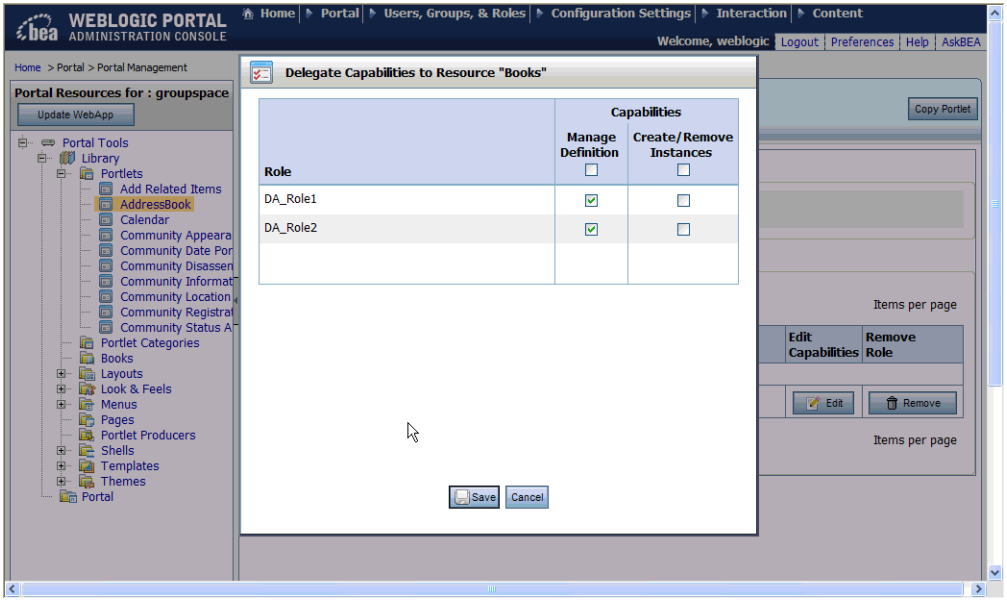
Perform the following steps to set delegated administration on a portal resource in the library:

1. Choose **Portal Management** > **Portal**.
2. From the Library node in the Portal Resources tree, navigate to and then select the portal resource (or resource type) for which you want to set delegated administration.
3. Select the Delegated Admin tab.
4. Click **Add Role**.
5. Optionally, search for the role you want to add by role name.
6. In the list of roles in the Search Results section, select the check box next to any roles you want to add and click **Add**. The selected roles are added to the Roles to Add section.

   You can remove a role from the Roles to Add section by selecting the check box next to the role and clicking **Remove Selected**.
7. Click **Save**.
8. In the Delegate Capabilities to Resource dialog, select the check boxes for the capabilities you want each role to have (see Table 7 -2, “Administrator Capabilities According to Portal Resource Type in the Library,” on page 7-20). By selecting the check box in the header above the role names, you enable that capability for all roles.
9. Click **Save**.

The roles you have added are listed in the Browse Roles Delegated to this Resource section. Figure 7-6 shows an example Delegate Capabilities to Resource Dialog.

Figure 7-6 Choosing Administrator Capabilities



Setting Delegated Administration on Portal Resources in the Desktop

Security policies determine what capabilities a delegated administration role has for a given portal resource. You can set delegated administration on portal resources in the library or in the desktop (Portals node). Within a given desktop you can set administrator capabilities on specific instances resources, such as a page, book, or portlet in that desktop. You can also set administrator capabilities on an entire desktop or community.

You can control administrator access to the following types of portal resources in the desktop:

- Portals
- Templates
- Desktops
- Communities
- Books

- Pages
- Portlets

The only capability that can be specified for a desktop instance of a portal resource is the Manage Instance capability. This allows administrators to manage that specific instance of the resource.

Perform the following steps to set delegated administration on a portal resource in the desktop:

1. Choose **Portal Management** > **Portal**.
2. From the Portals node in the Portal Resources tree, navigate to and then select the resource instance for which you want to set delegated administration.
3. Select the Delegated Admin tab.
4. Click **Add Role**.
5. Optionally, search for the role you want to add by role name.
6. In the list of roles in the Search Results section, select the check box next to any roles you want to add and click **Add**. The selected roles are added to the Roles to Add section.

   You can remove a role from the Roles to Add section by selecting the check box next to the role and clicking **Remove Selected**.
7. Click **Save**.
8. In the Delegate Capabilities to Resource dialog, select the check box for the Manage Instance capability. By selecting the check box in the header above the role names, you enable the Manage Instance capability for all roles.
9. Click **Save**.

The roles you have added are listed in the Browse Roles Delegated to this Resource section.

Removing and Editing Delegated Administration on Portal Resources

If you no longer want administrator capabilities to be available for a portal resource, you can remove administrator capabilities from it. You can also change the capabilities of a delegated administration role on a portal resource, which is also described in this procedure.

In the desktop, the only administrator capability for a resource instance is Manage Instance, so if you edit the role to remove this capability, the delegated administration role is removed from the resource instance.

Tip: You can also remove a delegated administration role from a portal resource from the Delegated Resources tab for that role. From this tab, you can delete a security policy by selecting the check box in the Delete column and clicking **Delete**.

Perform the following steps to remove or edit delegated administration from a portal resource or type of portal resource:

1. Choose **Portal > Portal Management**.
2. From the Library or Portals node in the Portal Resources tree, navigate to the resource or resource type from which you want to remove delegated administration.
3. Select the Delegated Admin tab.
4. From the Browse Roles Delegated to this Resource section:

- To remove the delegated administration role from the portal resource:
  a. Select the check box in the Remove Role column for each role you want to remove. By selecting the check box in the header above the role names, you can remove administration capabilities from all roles.
  b. Click **Remove**.
- To edit the capabilities of the delegated administration role on the portal resource:
  a. Select the check box in the Edit Capabilities column for each role you want to change capability for.
  b. Click **Edit**.
  c. In the Delegate Capabilities to Resource dialog, select the check boxes for the capabilities you want each role to have (see Table 7 -2, “Administrator Capabilities According to Portal Resource Type in the Library,” on page 7-20). By selecting the check box in the header above the role names, you enable that capability for all roles.
  d. Click **Save**.

The changes you make are reflected in the Browse Roles Delegated to this Resource section.

Setting Delegated Administration on Interaction Management Resources

You can determine the level of access portal administrators have in administering interaction management resources (campaigns, placeholders, segments, and content selectors) by setting delegated administration on them.

The only administrator capability for interaction management resources is Can Manage, which determines whether the administrator can manage that interaction management resource.

Perform the following steps to set delegated administration on an interaction management resource:

1. Depending on the resource for which you want to remove delegated administration, choose **Interaction > Campaigns**, **Interaction > Placeholders**, **Interaction > Segments**, or **Interaction > Content Selectors.**
2. In the tree, select the interaction management resource to which you want to assign delegated administration.
3. Select the Delegated Admin tab.
4. Click **Add Role**.
5. Optionally, search for the role you want to add by role name.
6. In the list of roles in the Search Results section, select the check box next to any roles you want to add and click **Add**. The selected roles are added to the Roles to Add section.

   You can remove a role from the Roles to Add section by selecting the check box next to the role and clicking **Remove Selected**.
7. Click **Save**.
8. In the Delegate Capabilities to Resource dialog, select the check box for the Can Manage capability. By selecting the check box in the header above the role names, you enable that capability for all roles.
9. Click **Save**.

The changes you make are reflected in the Browse Roles Delegated to this Resource section.

Removing Delegated Administration on Interaction Management Resources

If you no longer want administrator capabilities to be available for an interaction management resource, you can remove administrator capabilities from it.

The only administrator capability for interaction management resources is Can Manage, so if you edit the role to remove this capability, the delegated administration role is removed from the interaction management resource.

Perform the following steps to remove delegated administration from an interaction management resource:

1. Depending on the resource for which you want to remove delegated administration, choose **Interaction > Campaigns**, **Interaction > Placeholders**, **Interaction > Segments**, or **Interaction > Content Selectors.**
2. In the tree, select the interaction management resource from which you want to remove delegated administration.
3. Select the Delegated Admin tab.
4. Under Browse Roles Delegated to this Resource, select the check box in the Remove Role column for each role you want to remove. By selecting the check box in the header above the role names, you remove the Can Manage capability from all roles.
5. Click **Remove**.

The changes you make are reflected in the Browse Roles Delegated to this Resource section.

Setting Delegated Administration on Content Management Resources

You can determine the level of access portal administrators have in administering for content management resources.

You can create delegated administration roles to control administration access to the following types of content management resources:

- Repositories
- Content

- Content types
- Workflows

Each has administration capabilities that are based on the type of resource, as shown in Table 7 -4.

Table 7 -4 Administration Capabilities According to Content Management Resource Type

	Create	View	Update	Delete	Publish	Instan-tiate	Assign Workflow	Manage
Content	✓	✓	✓	✓	✓		✓	
Content Type	✓	✓	✓	✓		✓	✓	
Workflow	✓	✓	✓	✓			✓	
Repository								✓

Tip: The capabilities you assign to a delegated administration role determine how the administrator participates in the content workflow. For example, a role that is not granted Publish capabilities cannot transition content to the Published or Retired status.

The capabilities that can be specified for content are described in Table 7-5.

Table 7-5 Descriptions of Administrator Capabilities for Content

Create	Determines whether administrators can create content.
View	Determines whether administrators can view the content and any properties associated with it.
Update	Determines whether administrators can update the properties and change the content workflow status of the content.
Delete	Determines whether administrators can delete the content.

Table 7-5 Descriptions of Administrator Capabilities for Content

Assign Workflow	Determines whether administrators can assign a workflow with the content.
Publish	Determines whether administrators can approve the content by checking it in with a status other than draft or ready.

The capabilities that can be specified for content types are described in Table 7-6.

Table 7-6 Descriptions of Administrator Capabilities for Content Types

Create	Determines whether administrators can create a content type.
View	Determines whether administrators can view the content type and its properties.
Update	Determines whether administrators can modify a content type.
Delete	Determines whether administrators can delete a content type.
Instantiate	Determines whether administrators can create content based on this content type.
Assign Workflow	Determines whether administrators can assign a workflow to the content type.

The capabilities that can be specified for content workflows are described in Table 7-7.

Table 7-7 Descriptions of Administrator Capabilities for Content Workflows

Create	Determines whether administrators can create a content workflow.
View	Determines whether administrators can view the properties of a content workflow.
Update	Determines whether administrators can modify a content workflow.
Delete	Determines whether administrators can delete a content workflow from the repository.
Assign Workflow	Determines whether the workflow is available for selection when an administrator assigns a workflow to a content type or content.

The only capability that can be specified for a repository is the Manage capability. This determines whether administrators can modify the properties of the repository.

Perform the following steps to set delegated administration on content:

1. Choose **Content > Content Management**.
2. In the Content tree, navigate to the resource on which you want to set administrator capabilities:
   - To set delegated administration on workflows, select Repositories, and navigate to the workflow.
   - To set delegated administration on a content type. select Types, and navigate to the content type.
   - To set delegated administration on content, select Content, and navigate to the content.
   - To set delegated administration on a repository, select Repository and select the repository.
3. Select the Delegated Admin tab.
4. Click **Add Role**.
5. In the list of roles in the Search Results section, select the check box next to any roles you want to add and click **Add**. The selected roles are added to the Roles to Add section.

   You can remove a role from the Roles to Add section by selecting the check box next to the role and clicking **Remove Selected**.
6. Click **Save**.
7. In the Delegate Capabilities to Resource dialog, select the check boxes for the capabilities you want each role to have (see Table 7-5, Table 7-6, and Table 7-7 for capabilities on content, content types, and workflows, respectively). By selecting the check box in the header above the role names, you enable that capability for all roles.
8. Click **Save**.

The roles you have added are listed under Browse Roles Delegated to this Resource.

Removing and Editing Delegated Administration on Content Management Resources

If you no longer want administrator capabilities to be available for content, a content type, or a workflow, you can remove administrator capabilities from it. You can also change the

capabilities of a delegated administration role on the content management resource, which is also described in this procedure.

Tip: You can also remove a delegated administration role from a content management resource from the Delegated Resources tab for that role. From this tab, you can delete a security policy by selecting the check box in the Delete column and clicking **Delete**.

Perform the following steps to remove or edit delegated administration on a content management resource:

1. Choose **Content > Content Management**.
2. In the Content tree, navigate to the resource on which you want to remove delegated administration.
3. Select the Delegated Admin tab.
4. From the Browse Roles Delegated to this Resource section:

- To remove the delegated administration role from the content management resource:
  a. Select the check box in the Remove Role column for each role you want to remove. By selecting the check box in the header above the role names, you can remove administration capabilities from all roles.
  b. Click **Remove**.
- To edit the capabilities of the delegated administration role on the content management resource:
  a. Select the check box in the Edit Capabilities column for each role you want to change capability for.
  b. Click **Edit**.
  c. In the Delegate Capabilities to Resource dialog, select the check boxes for the capabilities you want each role to have (see Table 7-5, Table 7-6, and Table 7-7 for capabilities on content, content types, and workflows, respectively). By selecting the check box in the header above the role names, you enable that capability for all roles.
  d. Click **Save**.

The changes you make are reflected in the Browse Roles Delegated to this Resource section.

Setting Delegated Administration on Visitor Entitlement Roles

You can determine the level of access portal administrators have in administering visitor entitlement roles by setting delegated administration on them.

The only administrator capability for visitor entitlements is Manage Role, which determines whether the administrator can manage that visitor entitlement role.

Perform the following steps to set delegated administration on a visitor entitlement role:

1. Choose **Users, Groups, & Roles > Visitor Entitlements**.
1. In the Visitor Roles tree, select the visitor role to which you want to assign delegated administration.
2. Select the Delegated Admin tab.
3. Click **Add Role**.
4. Optionally, search for the role you want to add by role name.
5. In the list of roles in the Search Results section, select the check box next to any roles you want to add and click **Add**. The selected roles are added to the Roles to Add section.

   You can remove a role from the Roles to Add section by selecting the check box next to the role and clicking **Remove Selected**.
6. Click **Save**.
7. In the Delegate Capabilities to Resource dialog, select the check box for the Manage Roles capability. By selecting the check box in the header above the role names, you enable that capability for all roles.
8. Click **Save**.

The changes you make are reflected in the Browse Roles Delegated to this Resource section.

Removing Delegated Administration from Visitor Entitlement Roles

If you no longer want administrator capabilities to be available for a visitor entitlement role, you can remove administrator capabilities from it.

The only administrator capability for visitor entitlements is Manage Role, so if you edit the role to remove this capability, the delegated administration role is removed from the visitor entitlement role.

Perform the following steps to remove delegated administration from a visitor entitlement role:

1. Choose **Users, Groups, & Roles > Visitor Entitlements**.
2. In the Visitor Roles tree, select the role from which you want to remove delegated administration.
3. Select the Delegated Admin tab.
4. Under Browse Roles Delegated to this Resource, select the check box in the Remove Role column for each role you want to remove. By selecting the check box in the header above the role names, you can remove the Manage Role capability from all roles.
5. Click **Remove**.

The changes you make are reflected in the Browse Roles Delegated to this Resource section.

CHAPTER 8

Configuring Visitor Entitlements

This chapter provides an overview of visitor entitlements. Visitor entitlements allow you to define who can access the resources in a portal application and what they can do with those resources. This access is based on the role assigned to a portal visitor, allowing for flexible management of the resources. Use the WebLogic Portal Administration Console to configure visitor entitlements.

Visitor entitlement roles dynamically determine what access privileges a portal visitor has based on username, group membership, user profile properties, session and request attributes, and date and time functions. For example, the `Gold Member` role could be assigned to certain visitors because they are part of the frequent flyer program and have flown more than 50,000 miles in the previous year. This role is dynamically assigned to visitors when they log in to the site.

As another example, if you have an Employee Review portlet, you can create a visitor entitlement role called `Managers` and assign only managers to this role. Only logged in portal visitors who are assigned that role can view the Employee Review portlet.

Note: If no visitor entitlement roles exist, the default behavior is to *allow* access to the portal and portal resources to all visitors. Content management entitlements are an exception to this policy. If there are no entitlements set on content management components, then those components are not accessible to visitors.

This chapter includes the following sections:

Creating Visitor Entitlement Roles

Visitor entitlement roles dynamically determine what access privileges a portal visitor has based on username, group membership, user profile properties, session and request attributes, and date and time functions.

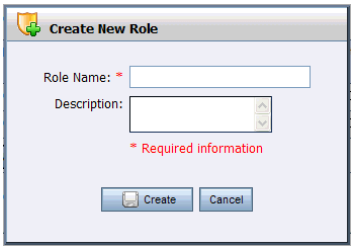
Perform the following steps to create a new visitor entitlement role:

1. Choose **Users, Groups, & Roles > Visitor Entitlements**.
2. In the Visitor Roles tree, select Visitor Roles.

   Note: You can also change the scope of the role, or set the scope to enterprise level, as described in "Using Web-Application or Enterprise-Application Scoped Roles for Entitlements on Portal Resources" on page 8-13.

3. From the Browse Roles tab, click **Create New Role**.
4. In the dialog box that appears, enter the name of the new visitor role, and optionally, a description, and click **Create**.

Figure 8-1 Create New Role Dialog



The new visitor entitlement role appears in the resource tree.

You can now define the role by adding users to the role, adding groups to the role, or using expressions. For more information, see "Adding Users, Groups, and Conditions in Visitor Entitlement Roles" on page 8-3.

After you define the visitor entitlement role, you can set entitlements on portal resources, content management resources, and groups.

Adding Users, Groups, and Conditions in Visitor Entitlement Roles

Once you create visitor roles in the WebLogic Portal Administration Console, you can add users and groups to them. You can also create conditions, based on user profile properties, session and request attributes, dates, and times, that determine who is assigned a visitor entitlement role.

Adding Users to Visitor Entitlement Roles

When you add a user to a visitor role, you grant that visitor access to the resources in a portal application and determine what they can do with those resources. This section describes how to add one or more users to a visitor role.

For optimal performance, if you have a large number of users you want to add to a role, either:

- Add the users to groups and then create roles with those groups, as described in "Adding Groups to Visitor Roles" on page 8-4

- Create roles with expressions, as described in "Adding Conditions to Visitor Roles with Expressions" on page 8-5

Perform the following steps to add one or more users to a visitor entitlement role:

1. Choose **Users, Groups, & Roles > Visitor Entitlements**.
2. In the Visitor Roles tree, select the role for which you want to add users.
3. Select the Users in Role tab.
4. Click **Add Users To Role**.
5. If necessary, find the users you want to add to the role using the Search feature. Users appear in the Search Results section.

Tip: If you are using an SQL authentication provider, be aware that user names are case sensitive. For example, user `Bob` is different than user `bob`.

---

6. Select the check box next to each user you want to add, and click **Add**. Selected users now appear in the Users to Add section.
7. Click **Save**.

Any users you have added now appear in Users in Role section in the Details and Users in Role tabs.

Adding Groups to Visitor Roles

When you add a group to a role, you grant the members (users) in that group—and users in any sub-groups of that group—access to all of the visitor entitlements attributed to that role.

Perform the following steps to add a group to a visitor role:

1. Choose **Users, Groups, & Roles > Visitor Entitlements**.
2. In the Visitor Roles tree, select the role for which you want to add groups.
3. Select the Groups in Role tab.
4. Click **Add Groups To Role**.
5. If necessary, find the groups you want to add to the role using the Search feature. Groups appear in the Search Results section.

Tip: If you are using an SQL authentication provider, be aware that group names are case sensitive. For example, group `Managers` is different than group `managers`.

6. Select the check box next to each group you want to add, and click **Add**. Selected groups now appear in the Groups to Add section.

Tip: Roles can sometimes be mapped directly to groups. The difference between groups and roles is that group membership is statically assigned by a server administrator, while role membership is dynamically determined based on information including the username, group membership, user profile properties, session and request attributes, and date and time functions. Roles can also be scoped to specific WebLogic resources within a single application in a WebLogic Server domain, while groups are always scoped to an entire WebLogic Server domain.

Note: If a list of groups is not displayed, make sure you have built a group hierarchy tree for the authentication provider. If you do not see a list of groups after building a group hierarchy tree, the authentication provider might not allow read access. To see if your authentication provider allows read access, view the authentication provider details, as described in "Viewing Authentication Provider Details" on page 6-4.

You can activate a text field for group name entry for authentication providers that do not allow read access.

7. Click **Save**.

Any groups you have added now appear in the Groups in Role section in the Details and Groups in Role tabs.

Adding Conditions to Visitor Roles with Expressions

You can use expressions to set conditions, in addition to username and group membership, that dynamically determine membership in a visitor entitlement role. Conditions specify the values of user profile properties, session and request attributes, dates, and times.

For example, you can define a role with the following expression: If a logged-in user has the `administrator` property set to `true` and the time is between 9 a.m. and 5 p.m. PST, the user is a role member.

Perform the following steps to add conditions to a visitor role:

1. Choose **Users, Groups, & Roles** > **Visitor Entitlements**.

2. In the Visitor Roles tree, select the role to which you want to add conditions.
3. Select the Role Expression tab.
4. In the top left corner of the window, ANY or ALL is underlined. By selecting, you can toggle between these values.
5. For each expression you want to create, click **Add Condition**. When you select a condition, it expands to let you specify the value. You can create an expression from a drop-down list containing the following options:
   - The date is:

     Specify a date using the calendar.
   - It is after a given date:

     Specify a date using the calendar.
   - It is after a given date and time:

     Specify a date and time using the calendar.
   - It is between two times:

     Specify a time range using the calendars.
   - It is between two dates:

     Specify a date range using the calendars.
   - It is between two date/times:

     Specify a range of dates and times using the calendars.
   - The visitor, visitor's HTTP request, or visitor's HTTP session has characteristics:

     To set characteristics, you must specify a Property Set, a Property from the property set, a Value for the property, and the ANY or ALL comparator. Specify a property value from the pull-down menu. You can click **Add Another Value** to add multiple properties and corresponding values.
   - The consumer's registration has these values:

     Specify WSRP registration properties. For more information, see the Federation Guide.

Tip: User profile properties, HTTP session and request properties, and WSRP registration properties are created by developers in WorkSpace Studio.

---

6. Click **Save** to apply the conditions.

Note: If you define roles with expressions whose evaluation changes during the processing of a request, you may need to adjust your portal application cache settings to ensure that the correct role definition is retrieved instead of a cached role.

Removing Users, Groups, and Conditions from Visitor Entitlement Roles

You can change who is assigned a role by removing users, groups, and conditions from visitor entitlement roles.

Removing Users from Visitor Entitlement Roles

If you want to revoke visitor access to the resources in a portal application associated with a role, you can remove a user from the role.

Perform the following steps to remove one or more users from a visitor entitlement role:

1. Choose **Users, Groups, & Roles > Visitor Entitlements**.
2. In the Visitor Roles tree, select the role from which you want to remove users.
3. Select the Users in Role tab.
4. In the Users in Role section, select the check box in the Remove column next to each user you want to remove. By selecting the check box in the header above the user names, you can remove all users from the role.
5. Click **Remove**.

Users you have removed no longer appear in the Users in Role tab or the Users in Role section of the Details tab.

Removing Groups from Visitor Entitlement Roles

If you want to revoke visitor access to the resources in a portal application associated with a role, you can remove a group from the role.

Perform the following steps to remove one or more groups from a visitor entitlement role:

1. Choose **Users, Groups, & Roles > Visitor Entitlements**.
2. In the Visitor Roles tree, select the role from which you want to remove groups.

3. Select the Groups in Role tab.
4. In the Groups in Role section, select the check box in the Remove column next to each group you want to remove. By selecting the check box in the header above the group names, you can remove all groups from the role.
5. Click **Remove**.

Groups you have removed no longer appear in the Groups in Role tab or the Groups in Role section of the Details tab.

Removing Conditions in Visitor Entitlement Roles

Perform the following steps to remove one or more conditions from a role:

1. Choose **Users, Groups, & Roles > Visitor Entitlements**.
2. In the Visitor Roles tree, select the role from which you want to remove conditions.
3. Select the Expressions in Role tab.
4. In the list of conditions, select the check box in the Delete column next to each one you want to remove. By selecting the check box in the header above the conditions, you can remove all conditions from the role.
5. Click **Delete**.

Conditions you have removed no longer appear in the Role Expressions tab or in the Expressions in Role section of the Details tab.

Modifying Conditions in Visitor Entitlement Roles

You can modify an existing expression in a visitor entitlement role, as long as you do not want to change the type of condition. For example, if you created a condition based on a date range, you can change the dates.

You can also add a condition from this tab; see "Adding Conditions to Visitor Roles with Expressions" on page 8-5 for more information. To remove a condition, see "Removing Conditions in Visitor Entitlement Roles" on page 8-8.

Perform the following steps to modify a role condition:

1. Choose **Users, Groups, & Roles > Visitor Entitlements**.
2. In the Visitor Roles tree, select the role for which you want to modify a condition.

3. Select the Role Expressions tab.
4. Click **Edit** for the condition you want to modify.
5. Specify the new value or values for the condition.
6. Click **Save**.

The modified condition appears in the list of conditions.

Viewing Visitor Entitlement Role Details

Once you have created a role, you can select it in the Visitor Roles tree to see a detailed description of the role.

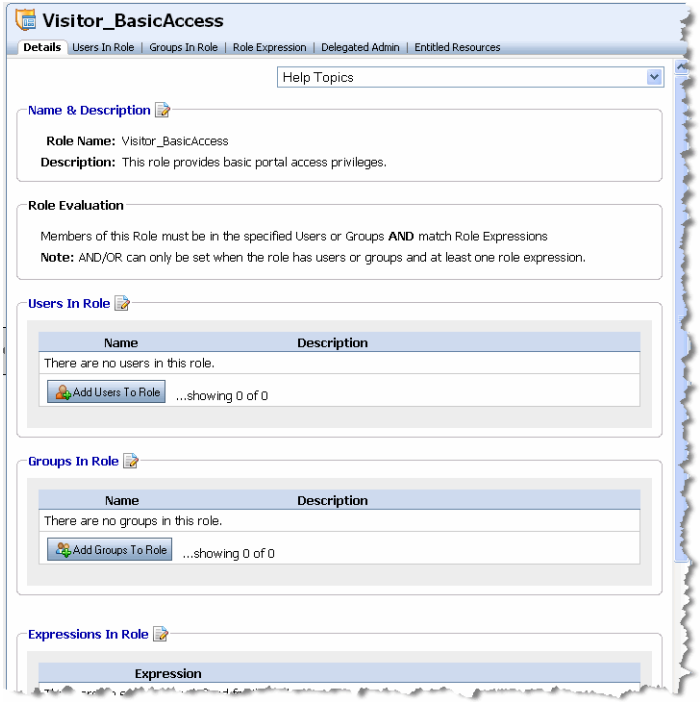
Perform the following steps to view the details of a visitor entitlement role:

1. Choose **Users, Groups, & Roles > Visitor Entitlements**.
2. In the Visitor Roles tree, select the role for which you want to see detailed information.

   Note: To see roles scoped to the enterprise level, or roles in a different web application, set the scope as described in "Creating Visitor Entitlement Roles" on page 8-2.

Figure 8-2 shows the Details tab for the `Visitor_BasicAccess` role.

Figure 8-2 Visitor Entitlements Details Tab



Viewing the Entitled Resources

You can view summary information about a visitor entitlement role to learn what security policies have been created for that role. This is useful because you cannot delete a visitor entitlement role until you remove its access to all resources.

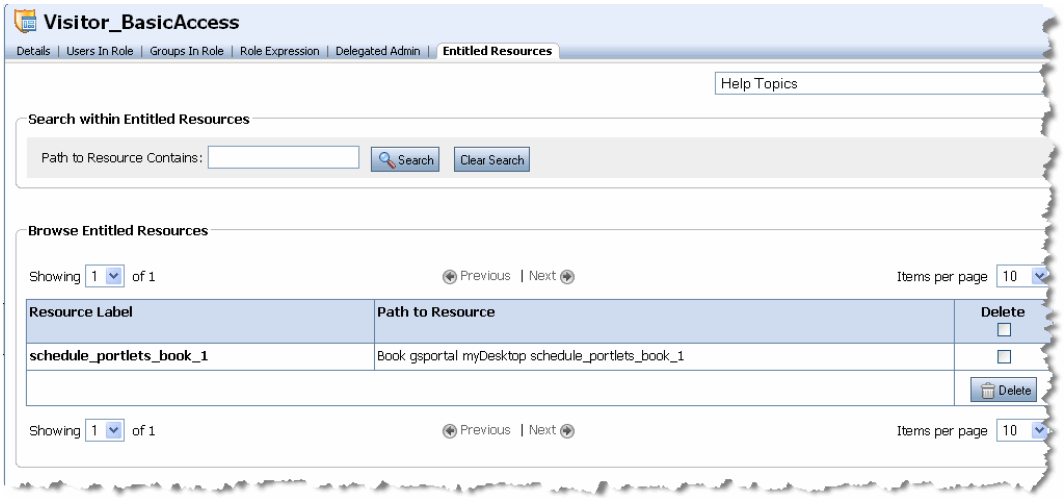
Perform the following steps to view a visitor entitlement role's policy summary information:

1. Choose **Users, Groups, & Roles > Visitor Entitlements**.
2. In the Visitor Entitlement Resource Tree, select a role.
3. Select the Entitled Resources tab. There you can view the information for the role policies:
   - Title of the resource
   - Path to the resource

Tip: From this tab, you can delete one or more role policies by selecting the check box in the Delete column and clicking **Delete**.

Figure 8-3 shows the Entitled Resources tab.

Figure 8-3 Entitled Resources Tab



Renaming Visitor Entitlement Roles

You can change the name and description of existing visitor entitlement role if there are no policies associated with the role. For information about viewing the policies associated with a role, see "Viewing the Entitled Resources" on page 8-10.

Tip: If there are policies associated with a role, it does not appear as editable in the Details tab.

Perform the following steps to rename a visitor entitlement role:

1. Choose **Users, Groups, & Roles > Visitor Entitlements**.
2. In the Visitor Roles tree, select the role you want to rename.
3. From the Details tab, select **Name & Description**, or click the **Edit** icon next to it.

4. In the dialog box that appears, type the new name, and optionally, a new description, and click **OK**.

The new role name appears in the Visitor Roles tree and the tabs.

Deleting Visitor Entitlement Roles

Perform the following steps to delete a visitor entitlement role:

1. Choose **Users, Groups, & Roles > Visitor Entitlements**.
2. In the Visitor Roles tree, select Visitor Roles.
3. In the Roles section, select the check box next to any roles you want to delete.
4. Click **Delete**.

If you receive a message that the role cannot be deleted while there are entitled resources associated with it, select the Entitled Resources tab for that role to view, and optionally delete, the resource dependencies. For more information, see "Viewing the Entitled Resources" on page 8-10.

Choosing Whether to Set Visitor Entitlements on Portal Resources in the Library or the Desktop

You can set visitor entitlements in the resource library or the desktop. Within the library, you can entitle specific books, pages, and portlets, or all resources in each of these categories. Within a given desktop you can entitle specific resources, such as a page, book, or portlet in that desktop. You can also entitle an entire desktop.

Visitor entitlements in the portal resource library apply to all instances of the resource in portal applications. However, they do not bar you from setting more local policies in the desktop. If you set a security policy for a resource in a desktop *but not in the resource library*, it applies only to that instance of the resource. Therefore, if you do not secure a resource within the resource library, you must secure each instance of the resource, wherever it appears in the hierarchy of books and pages in the desktop.

To protect *all instances* of a specific book, page, or portlet, or all books, pages, or portlets, set the security policies for the resource or resource type in the portal resource library. The library contains the master versions of all portal resources, and the security policies set in the library apply to a resource wherever it appears in the desktop (Portals node).

Using Web-Application or Enterprise-Application Scoped Roles for Entitlements on Portal Resources

You can use web-application scoped roles or enterprise-application scoped when setting entitlements on portal resources. If each web application has different requirements for constraints on visitor access, you should typically use web-application scoped roles. However, if you want to use the same roles in multiple web applications within an enterprise application, you can use enterprise-application scoped roles.

Perform the following steps to change the scope of a role:

1. Choose **Users, Groups, & Roles > Visitor Entitlements**.
2. In the section just above the Visitor Roles tree, following the text **Browse Roles from**, click **Update**.
3. Select one of the following radio buttons:
   - Enterprise Application Scope
   - Search for Web Application — All web applications are displayed in the Search Results list. You can find a specific web application using the Search feature.

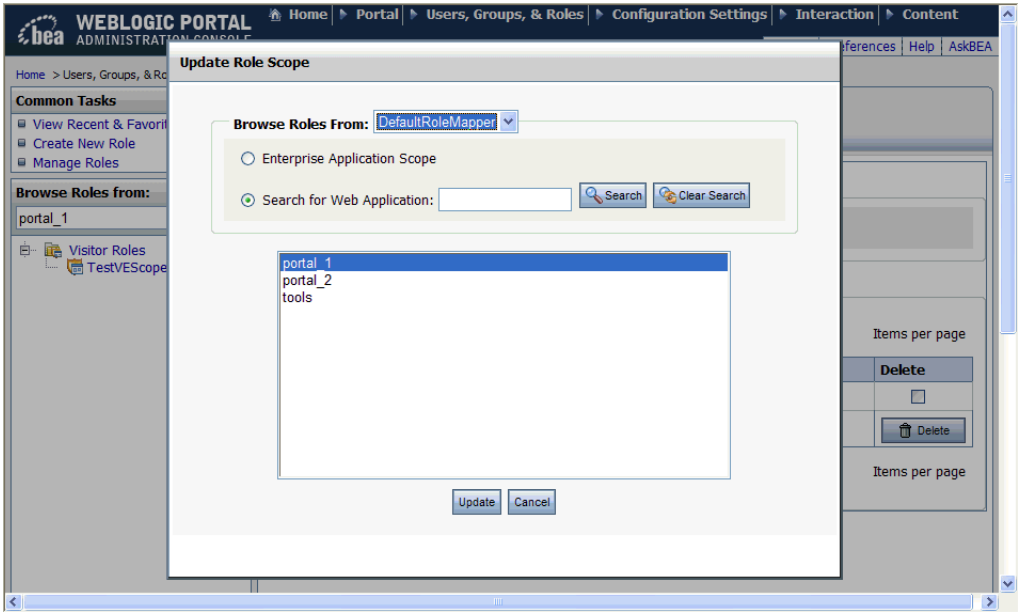
   Figure 8-4 shows the **Update Role Scope** dialog.

   Tip: When you assign a visitor role to a portal resource, you can choose from global WebLogic Server roles as well as enterprise-application and web-application scoped roles.

4. Click **Update**.

The text following **Browse Roles from** in the section above the Visitor Roles tree is updated.

Figure 8-4 Changing the Scope of a Role



Setting Visitor Entitlements on Portal Resources in the Library

Security policies determine what capabilities a visitor entitlement role has for a given portal resource. You can set visitor entitlements in the resource library or in the desktop (Portals node). Within the library, you can entitle specific books, pages, and portlets, or all resources in each of these categories.

Note: To protect *all instances* of a specific book, page, or portlet, or all books, pages, or portlets, set the security policies for the resource or resource type in the portal resource library. The library contains the master versions of all portal resources, and the security policies set in the library apply to a resource wherever it appears in the desktop.

You can create entitlements to control visitor access to the following types of portal resources in the library:

- Library
- Portlets

- Portlet categories
- Books
- Look and feels
- Pages

Each has visitor capabilities that are based on the type of resource, as shown in Table 8 -1.

Table 8 -1 Visitor Capabilities According to Portal Resource Type in the Library

	View	Minimize	Maximize	Edit	Remove	Offered
Library	✓					
Portlet	✓	✓	✓	✓	✓	✓
Portlet Category	✓					
Book	✓	✓	✓	✓	✓	
Look and Feel	✓					
Page	✓			✓	✓	

Table 8-2 describes each visitor capability.

Table 8-2 Descriptions of Visitor Capabilities for Portal Resources in the Library

View	Determines whether the portal visitor can see the resources in the portal desktop or within the Visitor Tools.
Minimize/Maximize	Determines whether the user is able to minimize or maximize the portlet or book. This applies to books within a page, not to the primary book.
Edit	Determines whether the user can rename the resource or modify its properties by either clicking the **Edit** icon within the portal desktop or the **Change Theme** or **Rename** icons within the Visitor Tools.

Table 8-2 Descriptions of Visitor Capabilities for Portal Resources in the Library

Remove	Determines whether the user can delete the resource by clicking the **Remove** icon within the portal desktop or Visitor Tools.
Offered	Determines whether the portlet will be offered (shown to a consumer) from the Web application's WSRP producer for that role. This feature allows producers to control which portlets are offered to specific consumers. For more information on consumer entitlement, see Consumer Entitlement in the *Federated Portals Guide*.

Note: If you create visitor entitlements on a portal resource, these can prevent a portal visitor from seeing a resource they would normally see according to personalization rules.

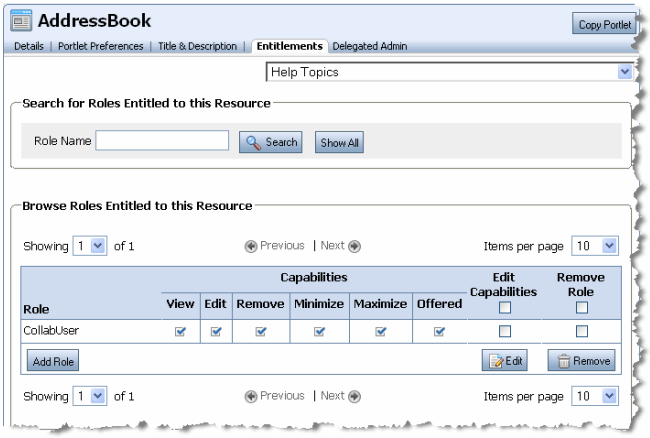
Perform the following steps to set visitor entitlements on a portal resource (or resource category) in the library:

1. Choose **Portal Management** > **Portal**.
2. From the Library node in the Portal Resources tree, navigate to and then select the portal resource (or resource category) for which you want to set visitor entitlements.
3. Select the Entitlements tab.
4. Click **Add Role**.
5. Optionally, search for the role you want to add by role name, or select the radio button to switch between enterprise-application scoped roles and web-application scoped roles.
6. In the list of roles in the Search Results section, select the check box next to any roles you want to add and click **Add**. The selected roles are added to the Roles to Add section.

   You can remove a role from the Roles to Add section by selecting the check box next to the role and clicking **Remove Selected**.
7. Click **Save**.
8. In the Entitle Capabilities to Resource dialog, select the check boxes for the capabilities you want each role to have (see Table 8 -1, "Visitor Capabilities According to Portal Resource Type in the Library," on page 8-15). By selecting the check box in the header above the role names, you enable that capability for all roles.
9. Click **Save**.

The roles you have added are listed in the Browse Roles Entitled to this Resource section, as shown in Figure 8-5.

Figure 8-5 Browse Roles Entitled to this Resource



Setting Visitor Entitlements on Portal Resources in the Desktop

Security policies determine what capabilities a visitor entitlement role has for a given portal resource. You can set visitor entitlements on portal resources in the library or the desktop (Portals node). Within a given desktop you can entitle specific resources, such as a page, book, or portlet in that desktop. You can also entitle an entire desktop or community.

Note: To protect *all instances* of a specific book, page, or portlet, or all books, pages, or portlets, set the security policies for the resource or resource type in the portal resource library. The library contains the master versions of all portal resources, and the security policies set in the library apply to a resource wherever it appears in the desktop.

You can create entitlements to control visitor access to the following types of portal resources in the desktop:

- Portals
- Templates
- Desktops
- Communities

- Books
- Pages
- Portlets

Each has visitor capabilities that are based on the type of resource, as shown in Table 8 -3.

Table 8 -3 Visitor Capabilities According to Portal Resource Type in the Desktop

	View	Minimize	Maximize	Edit	Remove	Create Community	Create, Read, Update, Delete Desktop
Portal						✓	✓
Template (Community and Desktop)	✓						
Desktop	✓						
Community	✓						
Book	✓	✓	✓	✓	✓		
Page	✓			✓	✓		
Portlet	✓	✓	✓	✓	✓		

Table 8-4 describes each visitor capability.

Table 8-4 Descriptions of Visitor Capabilities for Portal Resources in the Desktop

View	Determines whether the portal visitor can see the resources in the portal desktop or within the Visitor Tools.
Minimize/Maximize	Determines whether the user is able to minimize or maximize the portlet or book. This applies to books within a page, not to the primary book.

Table 8-4 Descriptions of Visitor Capabilities for Portal Resources in the Desktop

Edit	Determines whether the user can rename the resource or modify its properties by either clicking the **Edit** icon within the portal desktop or the **Change Theme** or **Rename** icons within the Visitor Tools.
Remove	Determines whether the user can delete the resource by clicking the **Remove** icon within the portal desktop or Visitor Tools.
Create Community	Determines whether the visitor can create a community within that portal.
Create, Read, Update, Delete Desktop	Determines whether the visitor can create, read, update, or delete desktops. These settings are designed to allow administrators to control desktop creation through the REST APIs. See the *Client-Side Development Guide* for information on these APIs.

Note: If you create visitor entitlements on a portal resource, these can prevent a portal visitor from seeing a resource they would normally see according to personalization rules.

Perform the following steps to set visitor entitlements on a portal resource in the desktop:

1. Choose **Portal Management > Portal**.
2. From the Portals node in the Portal Resources tree, navigate to and then select the resource instance for which you want to set visitor entitlements.
3. Select the Entitlements tab.
4. Click **Add Role**.
5. Optionally, search for the role you want to add by role name, or select the radio button to switch between enterprise-application scoped roles and web-application scoped roles.
6. In the list of roles in the Search Results section, select the check box next to any roles you want to add and click **Add**. The selected roles are added to the Roles to Add section.

   You can remove a role from the Roles to Add section by selecting the check box next to the role and clicking **Remove Selected**.
7. Click **Save**.
8. In the Entitle Capabilities to Resource dialog, select the check boxes for the capabilities you want each role to have (see Table 8 -3, “Visitor Capabilities According to Portal Resource Type in the Desktop,” on page 8-18). By selecting the check box in the header above the role names, you enable that capability for all roles.

9. Click **Save**.

The roles you have added are listed in the Browse Roles Entitled to this Resource section.

Removing and Editing Visitor Entitlements on Portal Resources

If you no longer want a visitor role to be assigned to a particular portal resource, you can remove the resource from the visitor entitlement role. You can also change the capabilities of a visitor entitlement role on a portal resource, which is also described in this procedure.

Tip: You can also remove a visitor role from a resource from the Entitled Resources tab for that role. From this tab, you can delete a security policy by selecting the check box in the Delete column and clicking **Delete**.

Perform the following steps to remove a visitor role from a portal resource or category of portal resource:

1. Choose **Portal Management > Portal**.
2. From the Library or Portals node in the Portal Resources tree, navigate to the resource, resource instance, or resource category from which you want to remove the visitor entitlements role.
3. Select the Entitlements tab.
4. From the Browse Roles Entitled to this Resource section:

- To remove the visitor entitlement role from the portal resource:
  a. Select the check box in the Remove Role column for each role you want to remove. By selecting the check box in the header above the role names, you can remove administration capabilities from all roles.
  b. Click **Remove**.
- To edit the capabilities of the visitor entitlement role on the portal resource:
  a. Select the check box in the Edit Capabilities column for each role you want to change capability for.
  b. Click **Edit**.

c. In the Entitle Capabilities to Resource dialog, select the check boxes for the capabilities you want each role to have (see Table 8 -1, “Visitor Capabilities According to Portal Resource Type in the Library,” on page 8-15 and Table 8 -3, “Visitor Capabilities According to Portal Resource Type in the Desktop,” on page 8-18). By selecting the check box in the header above the role names, you enable that capability for all roles.

d. Click **Save**.

The changes you make are reflected in the Browse Roles Entitled to this Resource section.

Setting Visitor Entitlements on Groups

GroupSpace and other community creators and owners can invite others to join the Community. Visitor entitlements determine whether a creator or owner can view potential members using the Browse options when selecting who to invite. For more information on GroupSpace and how to use invitations in GroupSpace, see the GroupSpace Guide.

The only visitor capability for groups is View access to the group, which determines whether the community owner or creator can see the group and the users in the group.

Perform the following steps to set visitor entitlements on a group:

1. Choose **Users, Groups, & Roles > Group Management**.
2. In the Groups tree, select the group for which you want to set visitor entitlements.
3. Select the Entitlements tab.
4. Click **Add Role**.

   You can select from enterprise-application scoped roles (not web-application scoped roles).
5. In the list of roles in the Search Results section, select the check box next to any roles you want to add and click **Add**. The selected roles are added to the Roles to Add section.

   You can remove a role from the Roles to Add section by selecting the check box next to the role and clicking **Remove Selected**.
6. Click **Save**.
7. In the Entitle Capabilities to Resource dialog, select the check box for the View capability. By selecting the check box in the header above the role names, you enable View capability for all roles.
8. Click **Save**.

The roles you have added are listed in the Browse Roles Entitled to this Resource section.

Removing Visitor Entitlements on Groups

If you no longer want visitors assigned to a role to be able to view a particular group, you can remove the visitor entitlement role from the group.

Tip: You can also remove a visitor role from a group from the Visitor Entitlements tree. In the Browse Policies section of the Entitled Resources tab for that role, select the check box in the Delete column for that policy and click **Delete**.

Perform the following steps to remove a visitor role from a group:

1. Choose **Users, Groups, & Roles > Group Management**.
2. In the Groups tree, select the group from which you want to remove the role.
3. Select the Entitlements tab.
4. In the Browse Roles Entitled to this Resource section, select the check box in the Remove Role column for each role you want to remove. By selecting the check box in the header above the role names, you can remove the all visitor roles from that group.
5. Click **Remove**.

The changes you make are reflected in the Browse Roles Entitled to this Resource section.

Setting Visitor Entitlements on Content Management Resources

Create security policies to determine what capabilities a visitor entitlement role has for a given content management resource.

Note: If no visitor entitlement roles exist, the default behavior is to *allow* access to the portal and portal resources to all visitors. Content management entitlements are an exception to this policy. If there are no entitlements set on content management components, then those components are not accessible to visitors.

Tip: Visitor entitlements on content management resources are used in the GroupSpace Document Library Portlet. For more information, see the GroupSpace Guide.

You can create entitlements to control access to the following types of content management resources:

- Repositories
- Content
- Content types
- Workflows

Each has visitor capabilities that are based on the type of resource, as shown in Table 8 -5.

Table 8 -5 Visitor Capabilities According to Content Management Resource Type

	Create	View	Update	Delete	Publish	Instan-tiate	Assign Workflow	Manage
Content	✓	✓	✓	✓	✓		✓	
Content Type	✓	✓	✓	✓		✓	✓	
Workflow	✓	✓	✓	✓			✓	
Repository								✓

Tip: The capabilities you assign to a visitor entitlement role determine how the visitor participates in the content workflow. For example, a role that is not granted Publish capabilities cannot transition content to the Published or Retired status.

The capabilities that can be specified for content are described in Table 8-6.

Table 8-6 Descriptions of Visitor Capabilities for Content

Create	Determines whether visitors can create content.
View	Determines whether visitors can view the content and any properties associated with it.

Table 8-6 Descriptions of Visitor Capabilities for Content

Update	Determines whether visitors can update the properties and change the content workflow status of the content.
Delete	Determines whether visitors can delete the content.
Assign Workflow	Determines whether visitors can assign a workflow with the content.
Publish	Determines whether visitors can approve the content by checking it in with a status other than draft or ready.

The capabilities that can be specified for content types are described in Table 8-7.

Table 8-7 Descriptions of Visitor Capabilities for Content Types

Create	Determines whether visitors can create a content type.
View	Determines whether visitors can view the content type and its properties.
Update	Determines whether visitors can modify a content type.
Delete	Determines whether visitors can delete a content type.
Instantiate	Determines whether visitors can create content based on this content type.
Assign Workflow	Determines whether visitors can assign a workflow to the content type.

The capabilities that can be specified for content workflows are described in Table 8-8.

Table 8-8 Descriptions of Visitor Capabilities for Content Workflows

Create	Determines whether visitors can create a content workflow.
View	Determines whether visitors can view the properties of a content workflow.
Update	Determines whether visitors can modify a content workflow.

Table 8-8 Descriptions of Visitor Capabilities for Content Workflows

Delete	Determines whether visitors can delete a content workflow from the repository.
Assign Workflow	Determines whether the workflow is available for selection when a user assigns a workflow to a content type or content.

The only capability that can be specified for a repository is the Manage capability. This allows you to modify the properties of the repository.

Note: If you create visitor entitlements on a content management resource, these can prevent a portal visitor from seeing content they would normally see according to personalization rules.

Perform the following steps to set visitor entitlements on content:

1. Choose **Content > Content Management**.
2. In the Content tree, navigate to the resource on which you want to set entitlements:

   - To set entitlements on workflows, select Repositories, and navigate to the workflow.

   - To set entitlements on a content type. select Types, and navigate to the content type.

   - To set entitlements on content, select Content, and navigate to the content.

   - To set entitlements on a repository, select Repository and select the repository.
3. Select the Entitlements tab.
4. Click **Add Role**.

   You can select from enterprise-application scoped roles (not web-application scoped roles).
5. In the list of roles in the Search Results section, select the check box next to any roles you want to add and click **Add**. The selected roles are added to the Roles to Add section.

   You can remove a role from the Roles to Add section by selecting the check box next to the role and clicking **Remove Selected**.
6. Click **Save**.
7. In the Entitle Capabilities to Resource dialog, select the check boxes for the capabilities you want each role to have (see Table 8-6, Table 8-7, and Table 8-8 for capabilities on content, content types, and workflows, respectively). By selecting the check box in the header above the role names, you enable that capability for all roles.

8. Click **Save**.

The roles you have added are listed in the Browse Roles Entitled to this Resource section.

Removing and Editing Visitor Entitlements on Content Management Resources

If you no longer want visitor capabilities to be available for content, a content type, or a workflow, you can remove visitor entitlements from it. You can also change the capabilities of the visitor entitlement role on the content management resource, which is also described in this procedure.

Tip: You can also remove a visitor entitlement role from a content management resource from the Entitled Resources tab for that role. From this tab, you can delete a security policy by selecting the check box in the Delete column and clicking **Delete**.

Perform the following steps to remove or edit visitor entitlements on a content management resource:

1. Choose **Content > Content Management**.
2. In the Content tree, navigate to the resource on which you want to remove or edit visitor entitlements.
3. Select the Entitlements tab.
4. From the Browse Roles Entitled to this Resource section:

- To remove the visitor entitlement role from the content management resource:
  a. Select the check box in the Remove Role column for each role you want to remove. By selecting the check box in the header above the role names, you can remove visitor capabilities from all roles.
  b. Click **Remove**.
- To edit the capabilities of the visitor entitlement role on the content management resource:
  a. Select the check box in the Edit Capabilities column for each role you want to change capability for.
  b. Click **Edit**.

c. In the Entitle Capabilities to Resource dialog, select the check boxes for the capabilities you want each role to have (see Table 8-6, Table 8-7, and Table 8-8 for capabilities on content, content types, and workflows, respectively). By selecting the check box in the header above the role names, you enable that capability for all roles.

d. Click **Save**.

The changes you make are reflected in the Browse Roles Entitled to this Resource section.

Designing Visitor Entitlements for Performance

The entitlement engine is called for rules checking during the render phase of an operation, which represents additional system overhead. The entitlements engine is also responsible for managing administrative tasks, which increases that overhead.

The following are recommendations for limiting the performance impact of visitor entitlements:

- Disable entitlements if a portal is not using any security policies.
- If a portal is using security policies, set the value for the <control-resource-cache-size=nn> attribute to equal the number of desktops + number of books + number of pages + number of portlets + number of buttons (max, min, help, edit) used in a portal. Use the default value if you are concerned about available memory.
- Limit your entitlement request to only one resource at a time. Bundling a larger number of resources (portlets, pages, books) with one entitlement request can cause a degradation in performance.
- If your portal uses more than 5000 entitlements, customize the cache settings for WebLogic Entitlements Engine.

CHAPTER 9

Deploying Security Components

This chapter describes how to deploy the components of your portal application that relate to security. It contains the following sections:

- Deploying the Enterprise Archive File
- Using the Propagation Utility

For more detailed information on deployment and propagation, see the Production Operations Guide.

Deploying the Enterprise Archive File

To bring your portal online in a production environment, it is first necessary to prepare your portal application. Typical preparation steps include modifying deployment descriptors for the product, building the enterprise archive (EAR) with all its pre-compiled classes, and deciding if you want to compress that EAR into an archive or leave it exploded.

Similar to any J2EE application, a portal application has a number of deployment descriptors that you may want to tune for your production environment.

Modifying Enterprise Application Deployment Descriptors

Deployment descriptors contain the settings for cache configuration, behavior tracking, campaign, and commerce tax information. If these values are different for your production environment than for your existing development settings, use a deployment plan, described in the Production Operations Guide, to modify appropriately before building the portal application.

Modifying Web Application Deployment Descriptors

The portal application has a `/WEB-INF` directory that contains a number of deployment descriptors you may need to modify for your production environment.

A J2EE standard deployment descriptor is `web.xml`. Among other settings, it has a set of elements for configuring security for the web application. You must use J2EE security to restrict access to JSPs and page flows in your portal applications; otherwise, a user can access those resources directly by typing the URL to those resources. For information on how to secure the JSPs and page flows, see Chapter 4, "Preventing Direct Access to Portal Application Resources".

Using the Propagation Utility

WebLogic Portal supports the use of multiple authentication providers in a portal domain, which means that users in external providers can log in to your portal applications. It also means that in your code you potentially have access to multiple user stores.

Because the propagation utility does not propagate some portal resources from the source to the destination system, there might be cases where propagated data depends on other data that is not propagated. For example, delegated administration and visitor entitlement roles and policies are propagated to the destination but the related users and groups are not, so you must manually add those related users, user profiles, and groups on the destination system.

For more information about the propagation utility, including which resources are propagated and which are not, see the Production Operations Guide.

Part IV Production

The production phase is the time that you make small changes to your production environment. Security tasks might include changing your authentication provider settings or making modifications to visitor entitlement roles, adding a small number of users or groups, and other maintenance tasks.

Some security tasks you can do in the production phase might include the following:

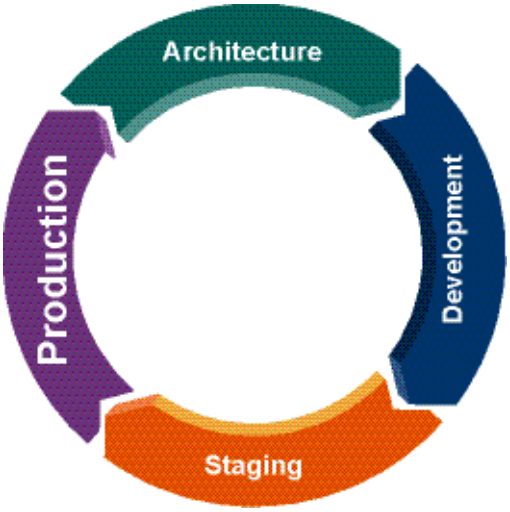
- Changing authentication providers, as described in Chapter 6, "Managing Security Providers"
- Creating a new group to reflect an organizational change, as described in the User Management Guide
- Creating or updating a role to modify visitor access to a portal resource, as described in Chapter 8, "Configuring Visitor Entitlements"

If you add or modify visitor entitlements to your portal application in the production phase, you must return to the staging phase to test the functionality you created. This staging environment simulates a production environment.

The production phase can be ongoing; some changes to your portal's functionality might even require you to return to the development phase and redeploy your portal application. Then you can move to the staging phase to test your changes and back to the production phase.

In the production phase, you might also modify users or groups or interaction management settings and queries. For information on configuring users and groups, see the User Management Guide. For information on interaction management settings and queries, see the Interaction Management Guide.

For a description of the production phase of the portal life cycle, see the WebLogic Portal Overview. The portal life cycle is shown in the following graphic:



CHAPTER 10

Implementing Authorization Programmatically

You create visitor entitlement roles using the WebLogic Portal Administration Console, as described in Chapter 8, "Configuring Visitor Entitlements". In addition, there are specific runtime checks that you can perform in your portal applications to customize a visitor's path through a portal based on their roles. You can use the JSP tags described in this chapter to perform authorization checks dynamically in portal applications. The access privileges are defined depending on the roles the user is assigned.

The `<auth:isAccessAllowed>` and `<auth:isUserInRole>` JSP tags enable you to customize a user's path through a portal by determining what their access privileges are.

The `<auth:isUserInRole>` JSP tag evaluates the current user's role at runtime so you can selectively authorize access to an application resource. The `<auth:isAccessAllowed>` JSP tag performs fine-grained entitlement-checking on application resources for which entitlements are not available by default.

This chapter includes the following sections:

- Verifying Whether a User Is Assigned a Specific Role
- Verifying Whether a User Has Access to a Resource

Verifying Whether a User Is Assigned a Specific Role

The `<auth:isUserInRole>` tag enables you to evaluate the current user's role at runtime so you can selectively authorize access to an application resource. By requiring authorization of the user accessing the JSP, you can restrict the display of application content wrapped by the tag. If used within an entitled portlet, this allows multiple levels of authorization.

The role being verified is compared to the set of valid roles for the user. Enterprise-application and web-application scoped visitor entitlement roles created using the WebLogic Portal Administration Console and global roles created with the WebLogic Server Administration Console are evaluated. Any role mapping provider that has roles mapped to the portal resource hierarchy is also evaluated.

Note: Each call to `<auth:isUserInRole>` causes all visitor roles for the current web application to be evaluated. This has performance implications if the role set is large. The map of computed roles is evaluated at most once per request, but is not cached across requests.

See the Javadoc for detailed instructions on using this tag.

Verifying Whether a User Has Access to a Resource

The `<auth:isAccessAllowed>` tag performs fine-grained entitlement-checking on application resources for which entitlements are not available by default. These are application-defined (non-portal) resources for which you have created your own security policies.

This tag provides a runtime check for authorizing access to an application resource. If access is allowed, the **id** return value is set to `true`. If access is denied, the **id** return value is set to `false` and the body of the tag is skipped.

Perform the following steps to use this tag:

1. Identify the taxonomy of the resource to be entitled. For example, if you are entitling a link on a JSP, the taxonomy is: JSP > link. Set the **resourceId** attribute to this value.
2. Create a security policy using the SecurityPolicyManager API using the **resourceId** from step 1. The policy can be predicated on WebLogic Portal roles.
3. In the WebLogic Portal Administration Console, create and define a visitor role that can access the resource you are entitling.
4. Add the `<auth:isAccessAllowed>` tag to your JSP, wrapped around the resource you want to entitle, and set the appropriate tag attributes. You can use an empty body form of this tag.

Attributes

If the attribute's value can be evaluated when the JSP tag is rendered at runtime, the attribute contains a **Yes** in the following table.

Table 10-1 isAccessAllowed Tag Attributes

id	**Required** | **String** | Can use runtime expressions: **No** The name of a variable that holds the result of the tag evaluation. Set to `true` or `false`.
resourceId	**Required** | **String** | Can use runtime expressions: **Yes** The application-defined taxonomy (hierarchy of resources) including the resource being requested. For example, if you are entitling a link on a JSP, the taxonomy is: JSP > link.
capability	**Optional** | **String** | Can use runtime expressions: **Yes** The requested capability for the resource. If no capability is specified (there is no capability name on the policy), a general security policy is searched for and used.
inheritSecurityPolicy	**Optional** | **Boolean** | Can use runtime expressions: **No** Allows a hierarchical taxonomy evaluation of security policies for authorization, instead of a single, first-found, evaluation. If this attribute is set to `false`, the first security policy found is used. If this attribute is set to `true`, any inherited security policies are used. The default is `false`.
needContextHandler	**Optional** | **Boolean** | Can use runtime expressions: **No** Must be set to `true` if there is the possibility that an in-scope role policy will be predicated on user profile attributes. The default is `true`.

Table 10-1 isAccessAllowed Tag Attributes

roleScope	**Optional** | **Integer** | Can use runtime expressions: **Yes** Determines the level in the taxonomy at which role policies are evaluated to allow or deny access to the resource. If you do not use this attribute, the search for the role is up to the `enterprise-application` scope (ENT_APP_ROLE_INHERITANCE). Possible values include the following: • `<%=EntitlementConstants.GLOBAL_ROLE_INHERITANCE%>` – Looks for role policies that are defined in the WebLogic Server Administration Console at a global scope. • `<%=EntitlementConstants.APPLICATION_INHERITANCE%>` – Looks for role policies that are defined in the WebLogic Server Administration Console at an application scope. • `<%=EntitlementConstants.ENT_APP_ROLE_INHERITANCE%>` – Looks for role policies at an enterprise application and global scope. • `<%=EntitlementConstants.WEBAPP_ROLE_INHERITANCE%>` – Looks for role policies at a web application, enterprise application, and global scope. • `<%=EntitlementConstants.LEAF_NODE_ROLE_INHERITANCE%>` – Looks for role policies at a resource leaf node and global scope only. • `<%=EntitlementConstants.HIERARCHICAL_ROLE_INHERITANCE%>` – Looks for role policies at each level up the taxonomy: leaf node, web application, enterprise application, and global scope.
subject	**Optional** | **Subject object** | Can use runtime expressions: **Yes** The subject (which maps to a user) for which the request is evaluated. If no subject is provided, the subject on the current request is used.

For more information on the `IsAccessAllowedTag` class, see the Javadoc.

Example

This example checks entitlements for a link in a JSP. The **resourceId** and **id** attribute values are read from variables declared earlier in the code.

The code in Listing 10-1 shows how to use `<auth:isAccessAllowed>` to check role policies. Because the **roleScope** attribute value is set to `HIERARCHICAL_ROLE_INHERITANCE`, the tag looks for existing role policies starting at the leaf node and up each level in the resource taxonomy. If the user does not belong to a role allowing access to this resource, the user will not see the link.

Listing 10-1 Sample Code That Checks Entitlements on a JSP

```
<%@ taglib "http://www.bea.com/servers/p13n/tags/auth" prefix="auth" %>
...
<auth:isAccessAllowed resourceId="<%=resourceId%>" id="<%=evalResult%>"
   roleScope="<%=EntitlementConstants.HIERARCHICAL_ROLE_INHERITANCE%>" >
<p>
<a href="HRpersonnel.jsp">Click here for secure personnel information.</a>
</auth:isAccessAllowed>
```

Other Tools

The following tools are alternatives to the `<auth:isAccessAllowed>` tag:

- Java – Work directly with the `com.bea.p13n.entitlements.servlets.jsp.taglib.IsAccessAllowedTag` class. For more information, see the Javadoc.
- In the WebLogic Portal Administration Console, create and define a visitor role that will be able to access the resource you are entitling. You can also view the policies for each role.