Oracle® Fusion Middleware

Administrator's Guide for Oracle Entitlements Server

11*g* Release 1 (11.1.1)

E14096-04

August 2011

ORACLE®

Oracle Fusion Middleware Administrator's Guide for Oracle Entitlements Server 11*g* Release 1 (11.1.1)

E14096-04

Primary Author: Michael Teger

Contributing Author:

Contributor: Subbu Devalpalli, Michael Khalandovsky, Akila Natarajan

Contents

Index

List of Examples

Preface

The *Oracle Fusion Middleware Administrator's Guide for Oracle Entitlements Server* provides information on system configuration administration and management of policy objects using the Oracle Entitlements Server Administration Console. Manual configuraton using back end files is also covered.

Audience

The intended audience of this guide is security administrators.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible to all users, including users that are disabled. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Accessibility standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/accessibility/.

Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/support/contact.html or visit http://www.oracle.com/accessibility/support.html if you are hearing impaired.

Related Documents

For more information, see the following documents:

- *Oracle Fusion Middleware Developer's Guide for Oracle Entitlements Server*
- *Oracle Fusion Middleware Installation Guide for Oracle Identity Management*
- *Oracle Fusion Middleware Release Notes*
- *Oracle Fusion Middleware Application Security Guide*
- *Oracle Fusion Middleware High Availability Guide*
- *Oracle Fusion Middleware Integration Guide for Oracle Access Manager*
- *Oracle Fusion Middleware Securing Oracle WebLogic Server*
- *Oracle Fusion Middleware Administrator's Guide*

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1

Introducing Oracle Entitlements Server

This chapter contains an overview of the features and architecture of Oracle Entitlements Server 11gR1. It contains the following sections:

1.1 About Access Control

Access control is a system used to grant or deny access to an enterprise's information, systems or resources. For the purposes of this documentation, the entity that is being protected is referred to as the *protected resource* while the entity to which access is granted or denied is referred to as the *subject* (most often a real person). A *policy* matches a subject with a set of *operations* that determine what the subject is allowed to see and do within the protected resource. The protected operations are dependent on the type of resource. For example, the protected operations attached to a text file (read, modify, delete) are different from those that can be attached to a banking application (view account, transfer money, modify profile).

Access control assures that only authorized subjects can access protected resources thus preventing the resources from unauthorized or inadvertent modification. Authorization of a subject typically occurs after authentication. In general, an access control system may comprise two types of authorization:

- Coarse grained authorization is perimeter authorization that uses technology principally focused on "keeping the bad guys out." Generally, it is performed by interceptors outside the application making the authorization call. It takes into account a URL and the policies regarding the subject requester.
- Fine grained authorization is more detailed, and primarily controlled by the application making the authorization call. It takes into account the URL of the protected resource and its configured policies as well as information that may include resource-specific or user attributes and the context of the request. For example, granting an employee access to a portal during normal business hours and denying this access during the weekend hours would call for fine grained authorization.

Thus, an access control system must support a policy model that is easy to administer yet allows for complex sets of conditions under which access can be granted (or denied). Oracle Entitlements Server is a product that provides centralized policy

management with centralized *or* distributed access control enforcement for all types of resources including software components and application business objects.

1.2 Overview of Oracle Entitlements Server

Oracle Entitlements Server is a fine-grained authorization product that allows an organization to protect its resources by defining and managing policies that control access to, and usage of, these resources. Access privileges are defined in a policy by specifying who can do what to which resource, when it can be done, and how. The policy can enforce controls on all types of resources including software components (URLs, Java Server Pages, Enterprise JavaBeans, methods, servlets and the like used to construct an application) and business objects (representations of user accounts, personal profiles and contracts such as bank accounts in a banking application, patient records in a health care application, or anything used to define a business relationship).

Oracle Entitlements Server supports the creation of role policies and access control policies. Role policies are used to define constraints regarding which users are assigned roles. (This is accomplished directly or indirectly using enterprise groups.) Access control policies define access to the software components and business objects. The following sections contain information on the previous releases of Oracle Entitlements Server and the features developed for this release, Oracle Entitlements Server 11gR1.

- Section 1.2.1, "Understanding Oracle Entitlements Server Releases"
- Section 1.2.2, "Using the Authorization Policy Manager Console"
- Section 1.2.3, "Features of Oracle Entitlements Server 11gR1"

1.2.1 Understanding Oracle Entitlements Server Releases

Oracle Entitlements Server 11gR1 represents a consolidation of Oracle Platform Security Services with Oracle Entitlements Server 10g (formerly BEA AquaLogic Enterprise Security). While Oracle Platform Security Services is a Java Authentication and Authorization Services (JAAS) security provider that offers coarse-grained authorization, Oracle Entitlements Server 11gR1 is an end-to-end enterprise solution that includes multiple technologies including Java Standard Edition (SE) and Enterprise Edition (EE), service-oriented architecture (SOA) and .NET. Oracle Entitlements Server 11gR1 offers fine-grained authorization in which a context for the authorization request is provided and access is granted or denied based on this. The core functionalities of Oracle Entitlements Server 11gR1 are based on the eXtensible Access Control Markup Language (XACML) specifications.

1.2.2 Using the Authorization Policy Manager Console

For this release, Authorization Policy Manager is the administration console for Oracle Entitlements Server. For purposes of the Oracle Entitlements Server documentation set, Authorization Policy Manager and variations of the Oracle Entitlements Server Administration Console (Administration Console, Console and the like) may be used interchangeably.

Note: Oracle Entitlements Server is not meant to contain the functionality of the product released as Oracle Authorization Policy Manager product. When referred to in this documentation set, Authorization Policy Manager is simply the Administration Console.

1.2.3 Features of Oracle Entitlements Server 11gR1

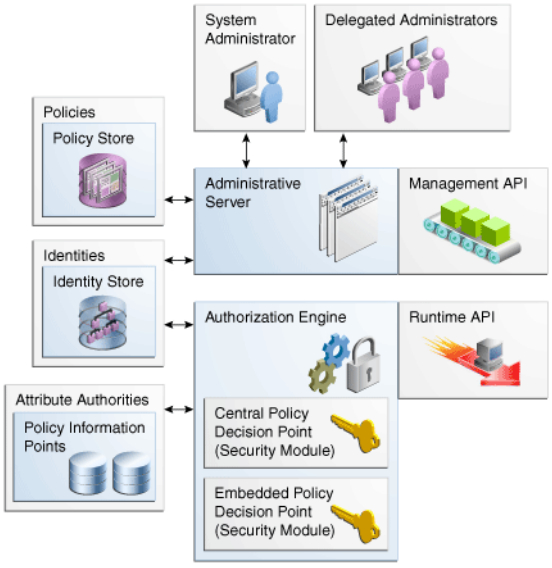
Oracle Entitlements Server 11gR1 offers fine-grained authorization and centralized entitlements management across heterogeneous application environments. Additionally, Oracle Entitlements Server:

- Distributes policies from the Administration Server to the decision endpoints.
- Caches policies and authorization decisions for performance.
- Updates security policies at run time.
- Offers a flexible architecture that supports both embedded and remote decision points (for centralized or distributed policy decisions).
- Separates security decision making from application logic.
- Audits all access decisions and management operations.
- Supports the eXtensible Access Control Markup Language (XACML) request/response protocol for authorization inquiries.
- Integrates with existing security and identity systems by leveraging enterprise data in relational databases and LDAP directories.

1.3 Overview of the Oracle Entitlements Server Architecture

From a high-level, Oracle Entitlements Server comprises centralized policy management with centralized *or* distributed policy decision making. The architecture of Oracle Entitlements Server is based on the interaction model of entities discussed in the XACML specifications. This model defines entities that provide a flexible architecture that can adapt to many components and deployments. Figure 1–1 illustrates the components of the Oracle Entitlements Server. Each of these components corresponds to one of the XACML entities.

Figure 1–1 Components of Oracle Entitlements Server



The following sections contain information on the Oracle Entitlements Server components and how they conform with the XACML entities.

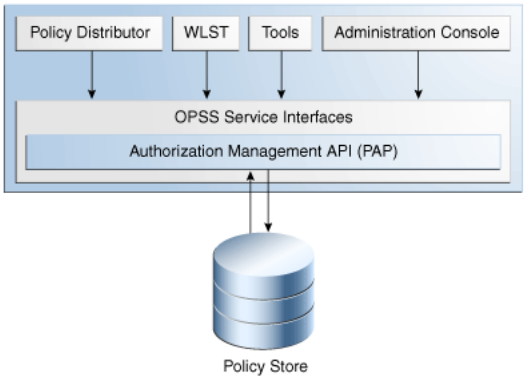
- Section 1.3.1, "The Policy Administration Point"

- Section 1.3.2, "The Policy Decision Point and the Policy Enforcement Point"
- Section 1.3.3, "The Policy Information Point"

1.3.1 The Policy Administration Point

The Policy Administration Point (PAP) is where policies used to protect a specified protected resource are created and managed. (In Figure 1–1, Administrative Server and Management API represent the PAP.) The PAP makes these rules available to the Policy Decision Point in order for that entity to reach a *grant* or *deny* decision for a request to access the protected resource. The Oracle Entitlements Server PAP is comprised of the Administration Console, management application programming interfaces (APIs) and management command line utilities. Figure 1–2 illustrates the architecture of these management and administration tools.

Figure 1–2 Oracle Entitlements Server PAP Architecture



1.3.2 The Policy Decision Point and the Policy Enforcement Point

When Oracle Entitlements Server is deployed, a Policy Decision Point (PDP) receives a request for authorization, evaluates it based on applicable policies, reaches a decision and returns the decision to the Policy Enforcement Point (PEP), the entity which first made the authorization call.

Note: The PDP can also retrieve additional subject, resource, action and environment attributes from a Policy Information Point to add contextual information to the request. See Section 1.3.3, "The Policy Information Point" for more information.

The PEP then enforces the decision. The PEP is a software component that intercepts the request to the protected application, passes it to the PDP and enforces the security decision returned from the PDP. This software component can be the protected application itself or a Security Module. The PEP is always integrated within a Jave Standard Enterprise (JSE) application or a Java Enterprise Edition (JEE) web container.

Note: The PDP may also return information with the decision - referred to as an *obligation* - that allows the decision to be enforced within a particular context. The application is not forced to act upon these obligations. See *Oracle Fusion Middleware Developer's Guide for Oracle Entitlements Server* for more information.

Oracle Entitlements Server offers two types of Security Modules. One type acts purely as a PDP by receiving requests and reaching decisions. The other type combines this PDP functionality with that of the PEP. The following sections illustrate how the Security Modules work.

- Section 1.3.2.1, "Security Module as PDP"
- Section 1.3.2.2, "Security Module as Combination PDP / PEP"
- Section 1.3.2.3, "Understanding the Types of Security Modules"

1.3.2.1 Security Module as PDP

When a Security Module acts purely as a PDP, its only functionality is decision making. It receives an authorization request and returns its decision to the PEP that originally made the authorization call. With the Security Module acting solely as the PDP, an external entity must act as the PEP - make the authorization call (using the Oracle Entitlements Server authorization API) and enforce the returned decision.

Figure 1–3 illustrates the process when the PEP entity making the call is the resource (application) itself. It receives the request for access, initiates authorization (through communication with the Security Module) and enforces the returned decision.

Figure 1–3 Application Acting as PEP Requests Decision from PDP

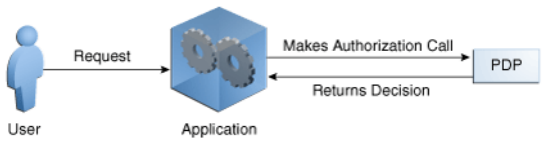
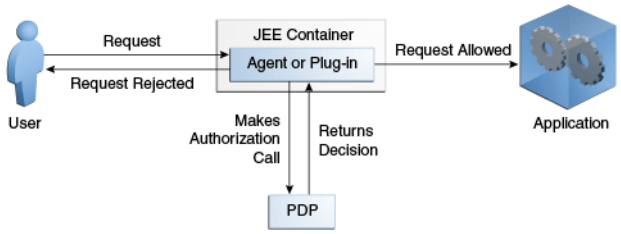


Figure 1–4 illustrates the process when the PEP entity is an agent or a plug-in (or similar software component) that intercepts the request before it reaches the application. The software component intercepts the request for access, initiates authorization (through communication with the Security Module) and forwards the decision returned from the Security Module to the application.

Figure 1–4 Agent Acting as PEP Intercepts Request and Makes Decision



Working together, these scenarios can offer a flexible authorization service. For example, the intermediary Web Services/XML gateway can request an authorization decision for a subject to access a portal. Assuming this primary decision is granted, the Web Services/XML gateway itself can then request secondary authorization decisions used to personalize the portal for the user that has been granted access.

1.3.2.2 Security Module as Combination PDP / PEP

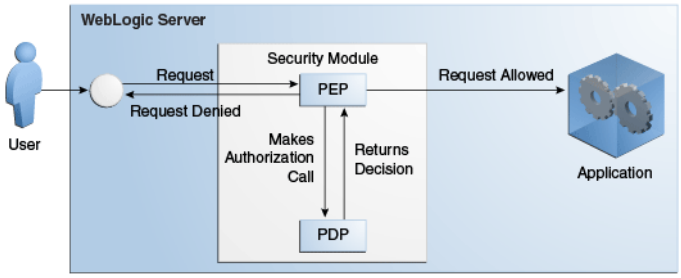
When a Security Module acts in tandem as a PDP and a PEP, it intercepts authorization requests, makes a decision, and enforces the decision. With this release of Oracle Entitlements Server, there is one Security Module that works in this manner: the WebLogic Server Security Module.

The WebLogic Server Security Module plugs directly into an Oracle WebLogic Server container that executes the protected application and will automatically request an authorization. In this scenario, a subject-initiated request to the application is intercepted by the WebLogic Server for authorization. The WebLogic Server, after successful authentication, attempts to authorize the request by making a call to a set of authorization providers configured during the Security Module's installation.

> **Note:** For more information on Oracle WebLogic Server, see the Oracle WebLogic Server Documentation Library at `http://download.oracle.com/docs/cd/E21764_01/wls.htm`.

The Role Mapping and Authorization Proxy providers communicate with the Oracle Entitlements Server authorization engine (calling the PEP API which, in turn, calls the PDP). The PDP computes a decision and returns the decision to the PEP which returns an appropriate response to the WebLogic Server. (Optionally, the PDP may return an obligation with the decision.) If access is denied, the WebLogic Server throws a security exception and prevents access. If access is permitted, the WebLogic Server allows access. Figure 1–5 illustrates this scenario.

Figure 1–5 Security Module as PDP and PEP



The benefits of using the providers is to allow fine-grained component level authorization. For example, you can use the providers to protect access to a servlet URL while allowing the servlet itself to make additional PEP API calls to decide which elements should be rendered on the returned page. By default, the Role Mapping and Authorization Proxy providers are not enabled.

> **Note:** See Section 11.1, "Integrating with WebLogic Server" for the procedure to enable the authorization providers using the WebLogic Server console. Post-configuration parameters are documented in Section A.2.4, "WebLogic Server Security Module."

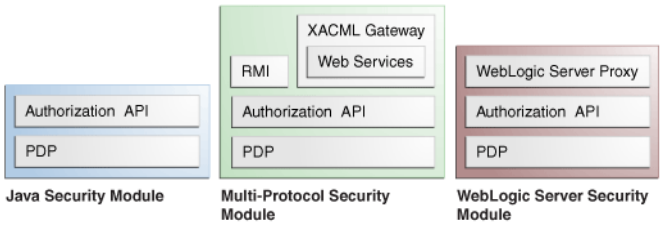
As mentioned in the previous paragraph's example, the application protected by the WebLogic Server can still make direct calls to Oracle Entitlements Server using the PEP API (as illustrated by Figure 1–3, "Application Acting as PEP Requests Decision

from PDP") for purposes similar to those discussed in Section 1.3.2.1, "Security Module as PDP."

1.3.2.3 Understanding the Types of Security Modules

Figure 1–6 illustrates how the various types of Security Modules have been developed.

Figure 1–6 Security Module Architecture



Based on this topology, the services of a Security Module can be invoked in several ways.

- The Java Security Module is a generic PDP that provides authorization decisions using Java API. This Security Module is supported on the following containers:
  - Java, Standard Edition (JSE)
  - WebSphere
  - JBoss
- The Multi-Protocol Security Module is an authorization service (based on service-oriented architecture principles) wrapped around a generic Java Security Module. It provides authorization decisions using RMI, Web Services and XACML (request and response). The Multi-Protocol Security Module is usually deployed on a central service rather then individual machines hosting the applications.

  Note: Figure 1–4, "Agent Acting as PEP Intercepts Request and Makes Decision" works similarly to the Multi-Protocol Security Module with an XML gateway; it intercepts requests and forces authorization before sending them to on to the destination.

- The WebLogic Security Module is a custom made Java Security Module that includes both a PDP and a PEP. It can receive requests directly from the WebLogic Server without the need for explicit authorization API calls. It only runs on the WebLogic Server container.

  Caution: Security Modules as central PDPs are supported for RMI, Web Services or XACML calls but Oracle Entitlements Server and the Security Modules cannot be in the same WebLogic Server domain.

See *Oracle Fusion Middleware Developer's Guide for Oracle Entitlements Server* for more information on requesting authorization decision and how Security Modules get updated policy information using the Policy Distribution Service.

1.3.3 The Policy Information Point

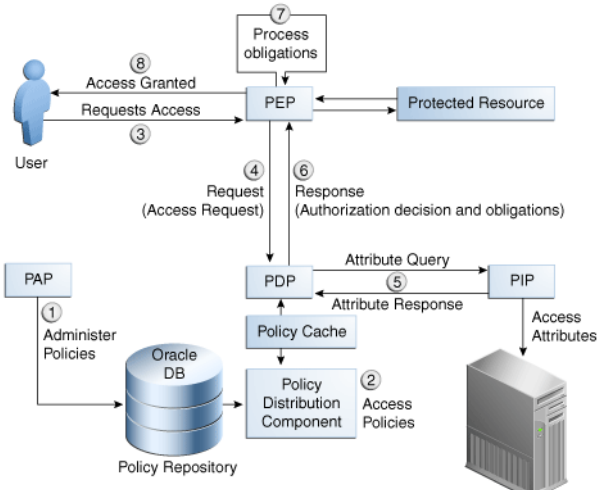
As defined in the XACML specifications, the Policy Information Point (PIP) is a data repository - a source from which information can be retrieved for use when evaluating policies for an authorization decision. This allows policies to be data-driven in that the value of an attribute can impact the access decision. For example, if access to transfer money from a bank account is based on how much money is currently in the account, an attribute retriever can be used to get a value for the current balance.

In an Oracle Entitlements Server deployment, *attribute retrievers* serve the PIP thus, the terms PIP and attribute retriever may be used interchangeably. The Attribute Authorities component illustrated in Figure 1–1, "Components of Oracle Entitlements Server" would be considered the PIP. Out of the box, an attribute retriever is available for both, LDAP and relational database data sources. See *Oracle Fusion Middleware Developer's Guide for Oracle Entitlements Server* for more information on attribute retrievers.

1.4 How Oracle Entitlements Server Processes Authorization Policies

The Oracle Entitlements Server authorization process involves the components described in Section 1.3, "Overview of the Oracle Entitlements Server Architecture". When a policy decision is requested, the PDP evaluates all policies related to the request and returns a *grant* or *deny* decision to the calling application. Figure 1–7 illustrates how the data flows during the policy authorization process.

Figure 1–7 How Data Flows in the Policy Authorization Process



1. Oracle Entitlements Server (acting as a PAP) is used to create and manage policies to protect a particular resource.
2. Policies in the policy repository are pushed to a policy cache, local to the PDP, by the Policy Distribution Service. The PDP reads policies from this cache. See *Oracle Fusion Middleware Developer's Guide for Oracle Entitlements Server* for more information.
3. A request for a resource is received by the PEP protecting it. The PEP can be the application itself or a Security Module - whichever makes the authorization call to Oracle Entitlements Server.
4. The PEP makes an authorization call to the PDP.

5. The Security Module PDP queries for additional subject, resource, action and environment attributes from the appropriate data source PIP.
6. The Security Module PDP evaluates the request and returns a response (and applicable obligations) to the PEP in the form of an authorization decision to grant or deny access.
7. The PEP fulfills any obligations, if applicable. An *obligation* is information returned with the decision upon which the PEP may or may not act. For example, an obligation may contain additional information concerning a decision to deny. The PEP entity is responsible for obligation fulfillment based on its settings. Oracle Entitlements Server is only responsible for forwarding the obligation based on policy configuration.
8. If access is permitted, the PEP grants the requester access to the resource; otherwise, access is denied.

Section 2.2, "How Oracle Entitlements Server Evaluates Policies" contains more details.

1.5 About the Supported Access Control Standards

Oracle Entitlements Server supports a number of access control models. Many access control products support only one of these models but Oracle Entitlements Server has implemented a policy model with the flexibility to support many of them. You can deploy strictly based on one model or mix and match pieces of different models. The following sections contain information on the access control models.

- Section 1.5.1, "Role-based Access Control (RBAC)."
- Section 1.5.2, "Attribute-Based Access Control (ABAC)."
- Section 1.5.3, "Java Permissions."
- Section 1.5.4, "XACML 2.0."
- Section 1.5.5, "PEP (Open Az) API"

1.5.1 Role-based Access Control (RBAC)

The Role-Based Access Control (RBAC) authorization model uses *roles* to define the privileges of a user. First, roles are created. Following, *permissions* to perform certain operations are assigned to the roles and, finally, *users* or *enterprise groups* are assigned to the roles. Through role assignment, the assignee acquires the right to perform the assigned operations. Thus, RBAC makes management of individual permissions simply a matter of assigning the appropriate roles to the appropriate entity. Roles can also be combined in a hierarchy where higher-level roles subsume permissions owned by sub-roles. Application Roles are used when modeling Oracle Entitlements Server deployments based on RBAC.

1.5.2 Attribute-Based Access Control (ABAC)

Attribute-Based Access Control (ABAC) provides the capability to define fine grained authorization using attributes. Roles need not be created. An ABAC policy specifies one or more *claims* that need to be satisfied before a user is granted access; for example, the user must be a certain age. If the user can prove this claim, access is granted.

Tip: Constraints are used when modeling Oracle Entitlements Server deployments based on ABAC. For more information, see *Oracle Fusion Middleware Developer's Guide for Oracle Entitlements Server*.

1.5.3 Java Permissions

Oracle Entitlements Server is built as an extension of Oracle Platform Security Services (OPSS). The OPSS security model is based on Java Authentication and Authorization Service (JAAS) security. JAAS institutes a permission based authorization system that implements Java-based security standards to support principal based and code based policies. A Java `Permission` object represents permission to access a resource. For example, the following code creates a `FilePermission` object representing read access to a file named `abc` in the `/tmp` directory.

```
perm = new java.io.FilePermission("/tmp/abc", "read");
```

Oracle Entitlements Server 11*g*R1 supports the Java Development Kit developer version 1.6 on either the Standard Edition or Enterprise Edition platforms

For more information, see the Java documentation at `http://www.oracle.com/technetwork/indexes/documentation/index.html`.

1.5.4 XACML 2.0

The eXtensible Access Control Markup Language (XACML) is an access control model that describes how to interpret policies, and an access control policy language (written using XML). Oracle Entitlements Server implements the XACML 2.0 request and response standard as well as the architecture model (described in Section 1.3, "Overview of the Oracle Entitlements Server Architecture.") It also implements how XACML defines policies as a collection of principals, resources, actions and attributes. For more information, see the XACML specifications at `http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml`.

1.5.5 PEP (Open Az) API

Oracle Entitlements Server has implemented the PEP Decision API, a part of the Open Az framework (`http://www.openliberty.org`). The `org.openliberty.openaz.azapi` package provides access from a PEP to a remote or embedded PDP. The `org.openliberty.openaz.azapi.pep` package (PEP API) has been implemented by Oracle Entitlements Server to be used by the PEP to issue authorization requests to a PDP. More information on the Open Az API can be found at `http://openaz.svn.sourceforge.net/viewvc/openaz/test/doc/index.html?org/openliberty/openaz/azapi/pep/package-summary.html`. More information on the PEP API implementation can be found in *Oracle Fusion Middleware Developer's Guide for Oracle Entitlements Server*.

Note: As implemented by Oracle Entitlements Server, the PEP API do not support authorization decisions for resources protected by Java permissions.

2

Understanding the Policy Model

A Policy specifies the criteria that must be satisfied in order to grant a requesting party access to a particular protected resource or assignment to a particular role. This chapter contains an overview of the Oracle Entitlements Server policy model, the elements that comprise a Policy and how the elements are organized in the policy store. It contains the following sections:

- Section 2.1, "Understanding Oracle Entitlements Server Policies"
- Section 2.2, "How Oracle Entitlements Server Evaluates Policies"
- Section 2.3, "The Policy Object Glossary"
- Section 2.4, "Implementing a Policy Use Case"

2.1 Understanding Oracle Entitlements Server Policies

Oracle Entitlements Server supports the creation of the following types of policies. The referenced sections contain detailed information regarding these policy types including how they are used.

- An *Authorization Policy* defines rules that control access to an organization's resources. See Section 2.1.1, "Understanding the Authorization Policy."

  Note: Resources may include software components or business objects. For more information, see Section 2.4, "Implementing a Policy Use Case."

- A *Role Mapping Policy* defines rules that control how principal users are granted or denied role memberships. See Section 2.1.2, "Understanding Role Assignments and the Role Mapping Policy."

2.1.1 Understanding the Authorization Policy

An *Authorization Policy* is created to grant or deny access to a particular resource based on the profile of the requesting user. From a high level, the Authorization Policy defines an association between an effect (GRANT or DENY), a principal (requesting user), the target resource, the resource's allowed actions and an optional condition. An Authorization Policy is applicable to a request for access if the parameters in the request match those specified in the policy. Consider this Authorization Policy definition:

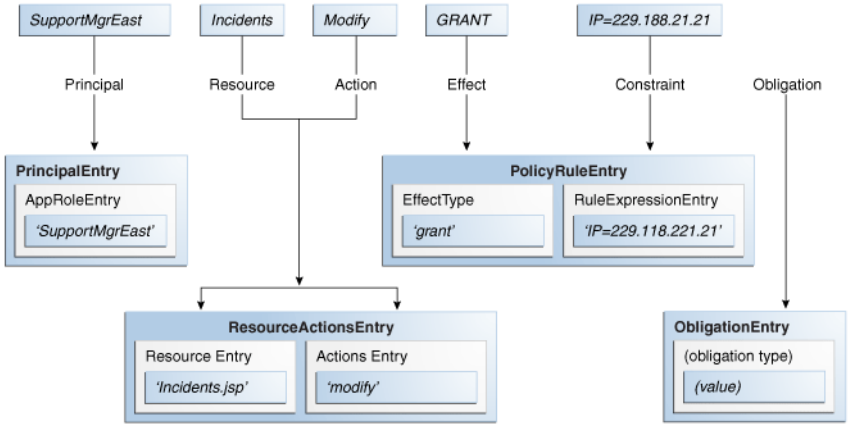
```
GRANT the SupportManagerEast role MODIFY access to the Incidents servlet
  if the request is made from an IP address of 229.188.21.21
```

This Authorization Policy will GRANT any user that is a member of the `SupportManagerEast` role access to the Incidents servlet for the purpose of modifying it. The policy is also constrained by a condition - the request must be made from IP address 229.188.21.21. Thus, if the parameters in the request match the parameters in the policy (a member of the `SupportManagerEast` role wants to modify the Incidents servlet), and the request is made from IP address 229.188.21.21, the request is granted. If the parameters in the request match the parameters in the policy (a member of the `SupportManagerEast` role wants to modify the Incidents servlet) but the request is NOT made from IP address 229.188.21.21, the policy is ignored. The following list of terms and values are extracted from this policy definition and comprise the components of the Authorization Policy.

- Effect: GRANT
- Action: MODIFY
- Target Resource: Incidents servlet
- Principal: member of SupportManagerEast role
- Condition: IP address 229.188.21.21

Figure 2–1 illustrates how the components of this policy map to the Oracle Entitlements Server Authorization Policy objects.

Figure 2–1 Policy Components Mapped to Authorization Policy Objects



For information on how to create, update and delete Authorization Policies, see Section 4.5.5, "Managing Authorization Policies."

2.1.2 Understanding Role Assignments and the Role Mapping Policy

An *Application Role* is a collection of users, groups, and roles. It can be assigned to an enterprise user, group, or external role in an identity store, or another Application Role in the policy store. (Assigning one Application Role to another Application Role allows you to build an Application Role hierarchy.) Application Roles can be assigned to a user in either of the following ways:

- By statically granting a specific user membership in the role.
- By referencing the Application Role in a *Role Mapping Policy* that will be used to dynamically assign role membership.

A Role Mapping Policy allows you to dynamically assign (GRANT) role membership to a user or dynamically revoke (DENY) role membership from a user. Consider the following Role Mapping Policy definition:

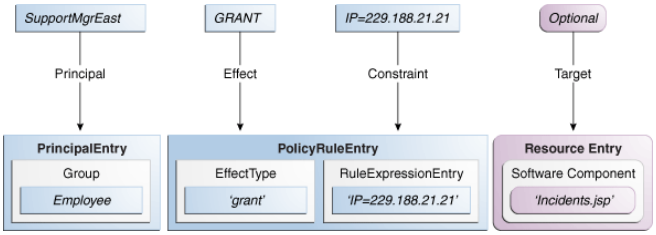
```
GRANT the Employee group application role SupportManageEast
   if the request is made from an IP address of 229.188.21.21
```

This policy grants the `SupportManagerEast` Application Role to any user that is a member of the group `Employee`. The policy is constrained by a condition though - the request must be made from IP address 229.188.21.21. Thus, if the parameters in the request match the parameters in the Role Mapping Policy (the requesting user is a member of the `Employee` group), and the request is made from IP address 229.188.21.21, the Application Role is granted. If the request is not made from the defined IP address, the Role Mapping Policy is ignored. The following terms and values are applicable to this Role Mapping Policy definition.

- Effect: GRANT
- Application Role: SupportManagerEast
- Principal: member of Employee group
- Condition: IP address 229.188.21.21

Figure 2–2 illustrates how the components of this policy map to the Oracle Entitlements Server Role Mapping Policy objects.

Figure 2–2 Policy Components Mapped to Role Mapping Policy Objects



A Role Mapping Policy can also be used to prevent specific users from being assigned an Application Role. Consider the following Role Mapping Policy definition:

```
DENY the Customers group application role GoldCircle
   if the account balance is less then $10,000
```

This policy denies the `GoldCircle` Application Role to any members of the group Customers IF their account balance is less then $10,000. For information on how to create, update and delete Role Mapping Policies, see Section 4.5.7, "Managing Role Mapping Policies."

2.2 How Oracle Entitlements Server Evaluates Policies

During Oracle Entitlements Server runtime evaluation, the following occurs:

1. Based on the subject, a list of Application Roles is determined by:
   a. Retrieving the user's static role membership.
   b. Evaluating all applicable Role Mapping Policies with a GRANT effect and adding them to the list of roles previously determined.
   c. Evaluating all applicable Role Mapping Policies with a DENY effect and removing them from the list of roles previously determined.

2. Based on the subject and list of retrieved Application Roles, a list of Authorization Policies is evaluated to find any that might be applicable based on the grantee, target matching and conditions. (The actions allowed on the resource are defined by the Authorization Policy.)
3. A final authorization decision is based on the "DENY overrides" combining algorithm and returned to the calling application.

Section 1.4, "How Oracle Entitlements Server Processes Authorization Policies" contains additional details on this process.

2.3 The Policy Object Glossary

The policy objects defined in this section can be created, provisioned and managed using the Authorization Policy Manager Administration Console.

- **Policy Store**

  The policy store is where all Oracle Entitlements Server policy objects (including, but not limited to, Applications, Resources and various role types) are stored. A policy store can be a relational database (preferred) or an LDAP-based directory. For more information, see Section 3.2.3, "Accessing the Policy Store."

- **Application**

  An Application is a high-level container for managing roles, policies, resource definitions, and other policy objects; in effect, all objects needed to define secure access to a particular application. An Application may correspond to a single deployed software application, a set of deployed software applications, or components of a software application (such as an Enterprise Java Bean). You can have more than one Application managed by Oracle Entitlements Server. For more information, see Section 4.5.1, "Managing Applications."

- **Application Role**

  An Application Role is a collection of users, groups, and other Application Roles; it can be assigned to an enterprise user, group, or external role in an identity store, or another Application Role in the policy store. For example, when creating an Application Role you might grant it all privileges necessary to access a given target Resource. Then, it can be assigned statically to a user by granting the user membership in the role, or dynamically by referencing the role in a Role Mapping Policy which will, in turn, grant the policy's principals the permissions granted or denied in the policy itself. One target application may have several different roles, with each role assigned a different set of privileges for more fine-grained access.

  Application Roles can be many-to-many mapped to external roles. For example, the external group `employee` (stored in LDAP-based identity store) can be mapped to the application role `customersupport member` (defined in one application) and to the application role `IT member` (defined in another application). For more information, see Section 4.5.6, "Managing Application Roles in the Role Catalog."

  Note: Search for Application Roles in the Role Catalog node of the Oracle Entitlements Server Administration Console. See Chapter 5, "Querying Security Objects" for more information.

- **External Role**

An External Role is a collection of users and groups defined in an external identity store such as an LDAP server or a database. The term *external role* is often synonymous with the terms enterprise role or enterprise group, and it is typically implemented as LDAP groups in the identity store. For information on adding an External Role to a policy, see Section 4.5.5.1, "Creating an Authorization Policy."

> **Note:** Within Oracle Entitlements Server, external roles and users are read-only and can be viewed. They are typically managed with a different tool, such as Oracle Identity Manager. For more information, see *Oracle Fusion Middleware System Administrator's Guide for Oracle Identity Manager*.

- **Principal**

  A principal is the identity to which the access rights defined in the policy are granted. A principal can be a user, an External Role, or an Application Role. Most frequently, it is an Application Role. For information on adding a principal to a policy, see Section 4.5.5.1, "Creating an Authorization Policy."

- **Authorization Policy**

  An Authorization Policy is a policy that specifies a set of rights that an entity (the grantee, a principal or code source) is allowed on a protected resource, such as viewing a web page or modifying a report. In short, it specifies who can do what in (or to) the protected resource. For more information, see Section 4.5.5, "Managing Authorization Policies."

  > **Note:** Search for Authorization Policies in the Default Policy Domain (or a custom Policy Domain, if applicable) node of the Oracle Entitlements Server Administration Console. See Chapter 5, "Querying Security Objects" for more information.

- **Role Mapping Policy**

  A Role Mapping Policy defines which users or External Roles are mapped to an Authorization Policy. Role Mapping Policies, at a minimum, are written to define what subjects (user and external roles) are assigned to the applicable role. They may also include conditions. For more information, see Section 4.5.7, "Managing Role Mapping Policies."

  > **Note:** Search for Role Mapping Policies under the Role Catalog node of the Oracle Entitlements Server Administration Console. See Chapter 5, "Querying Security Objects" for more information.

- Resource Type

  A Resource Type represents the type of a secured object. Protected software application components that share common characteristics can be represented by particular Resource Type. For example, a set of pages can be represented by one Resource Type and bank accounts by another Resource Type. A Resource Type defines *resource attributes* and possible valid actions that are applicable to the protected component. It also defines how to match a resource passed by the softare application to a Resource defined in an Authorization Policy. For more information, see Section 4.5.2, "Managing Resource Types."

- **Resource**

  A Resource is a protected component or object to which access is granted or denied. A Resource represents the application component or business object that can be secured by an Authorization Policy. At runtime, the application passes the Resource name to check for access definitions to determine whether a principal is authorized access. A Resource requires an associated Resource Type. For more information, see Section 4.5.3, "Managing Resources."

- **Policy Domain**

  A Policy Domain is a container under an Application object that can serve as a partition to facilitate management of Resources, Entitlements and Authorization Policies. The Policy Domain is an optional management construct that can restrict an administrator's right to a particular subset of Resource, Entitlements, and Authorization Policies. The Policy Domain has no effect upon runtime policy evaluation. Multiple Policy Domains can be created and are hierarchical. A *default policy domain* is added to each Application upon its creation. For more information, see Section 9.4, "Using Policy Domains to Delegate."

- **Entitlement**

  An Entitlement (also known as a *permission set*) represents a small set of Resources and the associated actions needed to perform a task. It groups related resources, possibly of different types, needed to perform a business function. An Entitlement is a reusable collection of access rights that can be granted to multiple principals. For more information, see Section 4.5.4, "Managing Entitlements."

- **Attributes and Functions**

  An *Attribute* represents data that can be used in a policy *condition*, or returned with the policy determination as an *obligation*. It is defined by its name, the type of data it takes as a value, and whether the value is single or multiple. An attribute value can either be passed by the protected application as part of an authorization request, or retrieved by Oracle Entitlements Server.

  A *Function* represents custom code that can be invoked as part of the evaluation of a policy condition; the returned value will affect the evaluation of the condition. For more information, see Section 4.5.9, "Managing Attributes and Functions as Extensions."

- **Condition**

  A Condition is one or more constraints that must be evaluated to true in order for the policy to be included in the authorization decision. Adding a Condition to a policy is optional and when used, further restricts access to the protected resource. In general, conditions consist of boolean expressions that test the value of some user, resource, or system attribute. Individual conditions can be combined with the following logical operators: AND, OR, and NOT. Conditions can define constraints based on date, time, a time range, a day of week, and so forth. For more information, see Section 4.6, "Using the Condition Builder."

- **Obligation**

  An Obligation specifies optional information that is returned together with an authorization decision. When used, an Obligation may impose an additional requirement for the policy enforcing component, or simply contain useful information. For example, the reason a request for access has been denied might be returned as an Obligation. For information on adding an Obligation to a policy, see Section 4.5.5.1, "Creating an Authorization Policy."

- **Role Category**

A Role Category is an optional tag that can be associated with an Application Role; it can be used for searching. Role Categories enable administrators to organize roles in arbitrary flat collections. They have no effect upon runtime policy evaluation. For more information, see Section 4.5.8, "Managing a Role Category."

2.4 Implementing a Policy Use Case

Oracle Entitlements Server provides the ability to externalize policy management and policy decision making logic from an organization's resources. It secures access to the organization's resources by implementing policies that specify the users, groups, and roles that can access them. Resources can be application software components (URLs, Enterprise JavaBeans, JavaServer Pages) or enterprise business objects (customer accounts, patient records). This use case considers how the policy model can be used to secure the financial services offered by Acme Investment Bank. It is based on the concept of *hierarchical resources* - resources are organized as a tree and inherit from their parent elements.

Note: Oracle Entitlements Server also supports the concept of non-hierarchical (flat) resources. See Section 4.5.2, "Managing Resource Types" for more information.

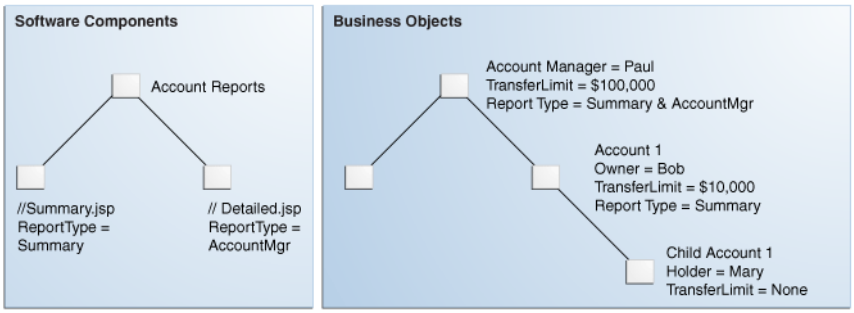
In this use case, the following conditions apply:

- A customer may open a family account and is considered an owner of the account.
- The customer may open a child account for family members and set transfer limits for each member. Transfer limits must be lower than the customer's own transfer limit.
- Each account has a bank employee that acts as an account manager and sets the transfer limits for the account.
- Both a Summary report and an Account Manager report are associated with each family account.

Figure 2–3 illustrates the financial services scenario by organizing the protected resources into business objects and software components. Paul is a bank employee who is an account manager and can set transfer limits up to $100,000 on the accounts that he manages. Paul manages account owner Bob's family account which has a transfer limit of $10,000. Bob manages his family account and a child account he created for Mary who may not transfer money. These accounts are considered business objects and are protected as such.

Account owner Bob has access to a Summary report generated for his family account. Account manager Paul has access to Bob's Summary report and his own Account Manager report. Child account holder Mary has access to no reports. These reports, generated as JavaServer Pages, are considered software components and are protected as such.

Figure 2–3 Use Case for Software Components and Business Objects



For any given user, a request for access to a financial services account (business object) or account report (software component) generates a decision based on the following questions:

1. Is this user an account holder, account owner or an account manager?
2. Can this user transfer funds from this account (subject to the role, transfer limit, and time of transaction)?
3. What reports can this user access?

The first two questions can be decided using policies created to protect business objects, while the last can be decided using policies created to protect software components. The following sections illustrate how to conceptualize the policies.

- Section 2.4.1, "Protecting Software Components"
- Section 2.4.2, "Protecting Business Objects"

2.4.1 Protecting Software Components

The Account Reports node in Figure 2–3, "Use Case for Software Components and Business Objects" represents the reporting application. `Summary.jsp` and `Detailed.jsp` are the software components. There are several options from which to choose when deciding how to model policies for securing these software components. One option is to set an Authorization Policy for the top node reporting application by using group membership. The following example illustrates how access is explicitly granted by naming the resource and the group of users that can access it.

```
GRANT the BankManagers group access to the AccountReports node
```

This top down Authorization Policy grants access to the Account Reports node for anyone in the BankManagers group. Because these resources are hierarchical, anyone in the allowed group has access to both the Summary and Detailed reports. But this access may be restricted using system-based or attribute-based conditions. For example, adding a condition based on time or based on the value of a specific user, group, or resource attribute would further limit access. The following example illustrates how a time-based condition restricts access of the reports to typical office hours.

```
GRANT the BankManagers group access to the AccountReports node
  IF the request is made between 09:00 and 17:00
```

Another option can set the top down Authorization Policy by defining the principal as a role rather than a group. A role is comprised of users or groups. (An LDAP role

would be granted enterprise wide whereas an Application Role is specific to the Application for which it was configured.)

> **Note:** A user can be assigned to a role through membership or a Role Mapping Policy as discussed in Section 2.1, "Understanding Oracle Entitlements Server Policies."

In the following example, the resource is explicitly named and access is implicitly granted to a user if the user is assigned the defined role.

```
GRANT access to AccountReports node if user has BankManagers role
```

You can also dynamically assign the BankManagers role to users accessing the reporting application if they are a member of the BankManagers group (as illustrated below).

```
GRANT BankManagers role to members of BankManagers group
  FOR access to AccountReports node
```

Another way to define the previous Authorization Policy is to assign the role based on a user attribute value rather than group membership. (In a large enterprise, it is typically more efficient to assign users based on attributes than on group membership.) The following example assigns the BankManagers role to the requesting user if the value of the UserType attribute in the user's profile is BankManager.

```
GRANT BankManagers role to anyone defined by UserType 'BankManager'
  FOR access to the AccountReports node
```

The previous examples represent Authorization Policies that scope from the top node reporting application down to the reports but Authorization Policies can also be defined for the specific report nodes. The following example grants access to the `Summary.jsp` report to all assignees of the BankManagers and AccountOwners roles. The additional condition is that the principal requesting access must be listed on the account to which the report pertains.

```
GRANT Summary.jsp access to all members of BankManagers and AccountOwners role
  IF the requesting assignee is listed as OWNER or MANAGER on specified account
```

Another example illustrates how access to the `Detailed.jsp` report can be granted to anyone who is assigned the BankManager role.

```
GRANT Detailed.jsp access to all assignees of BankManagers role
```

The previous examples show how an Authorization Policy can be modeled for specific application software components. In a real enterprise scenario, each application may have tens or hundreds of resources so it might not be practical to write an Authorization Policy for each one. The concept of *resource attributes* has been implemented by Oracle Entitlements Server to address this proliferation of application software component resources and associated Authorization Policies.

By associating a resource with an attribute, you can grant access based on the value of the attribute. For example, *filetype* could be a resource attribute that is used to define an HTML page, an image, or a PDF. By defining a condition as `if filetype=pdf`, you can grant access to all PDF files that are associated with the resource. The following example uses a resource attribute; it allows users assigned the BankManagers and AccountOwners role access to all reports although access is granted only if the report type being requested matches the value of the UserReportType attribute in the specific user's profile.

```
GRANT users assigned BankManagers or AccountOwners roles
     access to AccountReports
   IF requested ReportType matches UserReportType attribute value
     in user profile
```

This policy grants BankManagers and AccountOwners access to all reports although access is constrained based on matching resource attribute values with user attribute values. An advantage of this approach is that the policy governing access need not change as resources are added to, or removed from, an application. As resources change, the ReportType resource attribute attached to the application continues to govern access.

2.4.2 Protecting Business Objects

There are several options from which to choose when deciding how to model Authorization Policies for securing business objects. In this banking scenario, business objects are bank accounts. Figure 2–3, "Use Case for Software Components and Business Objects" illustrates the Acme bank account structure.

Each bank account can have a manager, an owner, and a holder with each *scope* assigned a certain set of privileges (or *entitlements*). The policy evaluating what a user can do on the bank account is then based on the user's attributes rather than the resource. The following example allows anyone to transfer money but that privilege is only granted if the user is defined as owner of the account requested and the amount of money being transferred is less than or equal to the limit defined for the user.

```
GRANT anyone transfer privileges only
   IF the user is listed as OWNER on specified account
  AND transfer amount is equal to or less than the transfer limit
```

There is another option to acquire a user's entitlements. Rather than comparing a transfer request to a transfer limit, Oracle Entitlements Server can return the transfer limit amount as the output of evaluation. In this scenario, the user's ability to access the account is verified but the transfer amount is returned to the caller (in a Java object) as an *obligation*. This leaves verification that the transfer amount is within the transfer limit up to the application. The following example illustrates this model.

```
GRANT anyone transfer privileges only
   IF the user is listed as OWNER on specified account
  THEN RETURN transfer limit to calling application
```

A model where the bank account corresponds to an individual resource instance can also be used; however, this would yield a proliferation of policies (one for each account) and become unmanageable. For example, if Acme Investment Bank had 100,000 accounts, it would need at least 100,000 policies just to manage transfers. For more information on adding obligations, see Section 4.5.5, "Managing Authorization Policies."

3

Getting Started With Oracle Entitlements Server

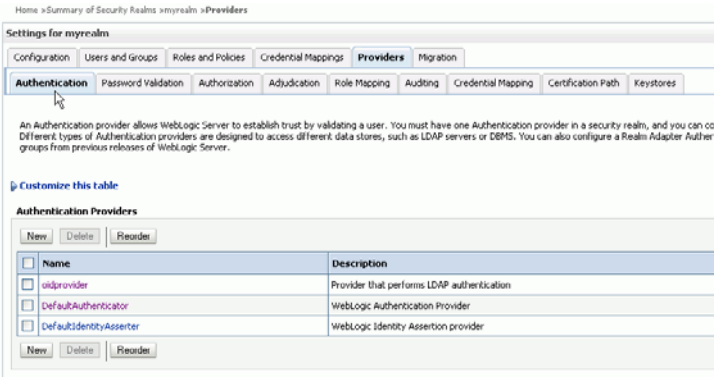
This chapter describes how to get started using Oracle Entitlements Server, including information about how to use and navigate the graphical interface. It contains the following sections.

3.1 Before You Begin

Before getting started using Oracle Entitlements Server, the following tasks must be done. They include installing the product and its components (for example, remote Security Modules), and configuring features like high availability and Secure Sockets Layer (SSL), if applicable. After finishing with these tasks, you can begin with Section 3.2, "Understanding The Graphical Interface."

- Install and configure Oracle Entitlements Server according to the instructions in *Oracle Fusion Middleware Installation Guide for Oracle Identity Management*.
  - For this release, the policy store managed by Oracle Entitlements Server can be a relational database (preferred) or an LDAP-based directory.
  - The identity store associated with Oracle Entitlements Server must be an LDAP-based directory.
- After installation, the Oracle Entitlements Server identity store is associated with the WebLogic Server embedded LDAP directory. While this embedded LDAP directory is fine for development purposes, a supported LDAP directory must be used in production. The following procedure reconfigures the default identity store settings. More specific information on configuring LDAP authentication providers can be found in the *Oracle Fusion Middleware Securing Oracle WebLogic Server*.
  1. Launch the WebLogic Server console.
  2. Click Security Realms.
  3. Click the settings for *myrealm*.
  4. Click the Provider tab.
  5. Click the Authentication tab as displayed in Figure 3–1.

Figure 3–1 The Authentication Provider Tab



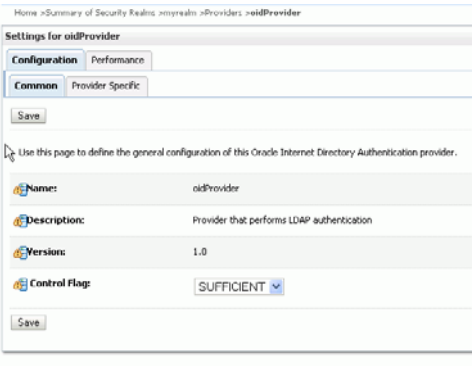
6. Click the New button to create a new provider.
7. Enter a name and select the type of LDAP-based directory.

   For example, *OracleInternetDirectoryAuthenticator*.
8. Configure the provider-specific attributes of the LDAP-based directory.

   This might include the host name and port, credentials, group search base, user search base and the like.
9. Save the provider information.
10. Change the order of the providers so that the LDAP-based directory is first.

    DefaultAuthenticator and *DefaultIdentityAsserter* will follow.
11. Click the new provider name to configure it.
    a. Click the Configuration tab.
    b. Click the Common tab.
    c. Set the Control Flag to SUFFICIENT and click Save as displayed in Figure 3–2.

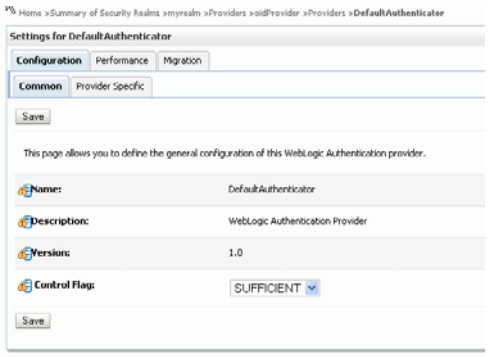
Figure 3–2 SUFFICIENT Control Flag



    d. Click the Provider Specific tab.
    e. Enter the LDAP configuration information for your identity store and click Save.

12. Return to the Providers tab.
13. Click *DefaultAuthenticator* to change its configuration.
14. Set the Control Flag to SUFFICIENT and click Save as displayed in Figure 3–3.

Figure 3–3 DefaultAuthentciator Tab in WebLogic Server Console



15. Restart WebLogic Server.

- For information about configuring high availability for Oracle Entitlements Server, see *Oracle Fusion Middleware High Availability Guide*
- For information regarding the authentication of users, see *Oracle Fusion Middleware Securing Oracle WebLogic Server*.

> **Note:** Oracle Entitlements Server is not involved in the authentication of users. This is normally done as part of the WebLogic Server security realm configuration.

- For information about configuring one-way SSL for connections that Oracle Entitlements Server establishes with the policy store, the identity store, and the database, see *Oracle Fusion Middleware Securing Oracle WebLogic Server*. Access to Oracle Entitlements Server using a browser can also be secured through one-way SSL. These settings are similar to those of any other application running in the Oracle WebLogic Server.
- Refer to the system requirements and certification documentation for information about hardware and software requirements, platforms, databases, and other information.
  - The system requirements document covers information such as hardware and software requirements, minimum disk space and memory requirements, and required system libraries, packages, or patches:

    `http://www.oracle.com/technology/software/products/ias/files/fusion_requirements.htm`
  - The certification document covers supported installation types, platforms, operating systems, databases, JDKs, and third-party products:

    `http://www.oracle.com/technology/software/products/ias/files/fusion_certification.html`

  These documents are available on Oracle Technology Network (OTN).

3.2 Understanding The Graphical Interface

Oracle Authorization Policy Manager is a sub-component of Oracle Entitlements Server that is the graphical console for administrators. It is a browser-based, graphical interface for managing policies and related policy objects. The following sections contain information to help understand the Authorization Policy Manager Administration Console.

- Section 3.2.1, "Assigning Oracle Entitlements Server Administrators"
- Section 3.2.2, "Using the Identity Store"
- Section 3.2.3, "Accessing the Policy Store"

3.2.1 Assigning Oracle Entitlements Server Administrators

Only users with sufficient privileges can log in to the Oracle Entitlements Server Administration Console or use administrative command-line tools such as the WebLogic Scripting Tool (WLST). An Oracle Entitlements Server system-level Administrator Role named `SystemAdmin` is created during installation and is mapped to the WebLogic Server administrator user (`weblogic`). The password is set during installation. SystemAdmin has extensive privileges that includes the rights to create additional Administrative Roles and delegating administrative rights to others.

Note: At first log in to the Oracle Entitlements Server Administration Console, SystemAdmin must use the credentials set during installation. The identifier and password can be changed by using your identity store's management tool.

You can create separate administrative users with different access rights for administering Oracle Entitlements Server and your environment. For more information, see Section 9.6, "Managing System Administrators Using Administrator Roles.".

3.2.2 Using the Identity Store

Oracle Entitlements Server administrator and user identities are stored in an identity store, typically an LDAP directory server. Users and external roles defined in the identity store are read-only during authorization policy definition. Oracle Entitlements Server reads and displays the data; it does not perform any management operations. Management of the identity data is accomplished using the identity store's tools or an identity management product such as Oracle Identity Manager.

3.2.3 Accessing the Policy Store

For this release, Oracle Entitlements Server the policy store used to maintain policy objects and defined policies can be a relational database (preferred) or an LDAP-based directory. (Oracle Internet Directory can be used as the policy store but has limited capabilities.) For links regarding hardware requirements, see Section 3.1, "Before You Begin." Instructions for creating and initializing the policy store can be found in *Oracle Fusion Middleware Installation Guide for Oracle Identity Management*.

3.3 Accessing the Administration Console

The following sections contain information on how to access the Authorization Policy Manager graphical interface (also referred to as the Administration Console).

3.3.1 Signing In to the Administration Console

Follow this procedure to sign in to the Authorization Policy Manager Administration Console.

1. Enter the Authorization Policy Manager Administration Console URL in the address bar of your browser. For example:

   https://hostname:port/apm/

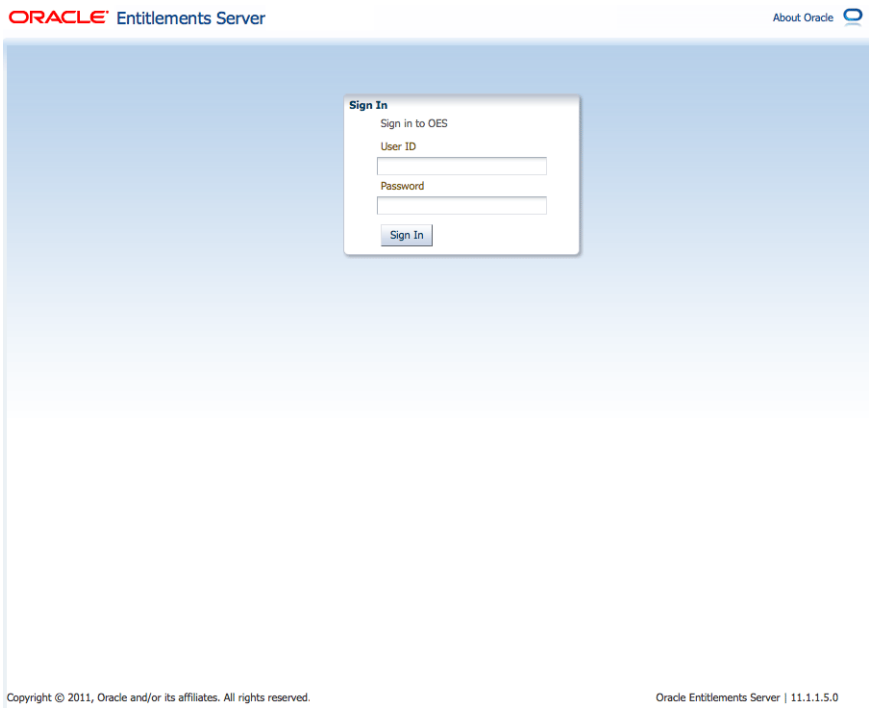
   where:

   - HTTPS represents the Hypertext Transfer Protocol (HTTP) with the Secure Socket Layer (SSL) enabled to encrypt and decrypt user page requests and the pages returned by the Web server.
   - *hostname* refers to the fully qualified domain name of the computer hosting the Oracle Authorization Policy Manager Administration Console.
   - *port* refers to the designated bind port for the Authorization Policy Manager Administration Console. (This is the same as the bind port for the WebLogic Server Administration Console.)
   - `/apm/` refers to the Authorization Policy Manager Log In page

2. Enter the System Administrator credentials.

   The default system administrator identifier is `weblogic`. The password is the same one supplied during installation. Figure 3–4 is a screenshot of the Sign In page.

Figure 3–4 Administration Console Sign In Page



3. Click **Sign In**.

3.3.2 Signing Out of the Administration Console

Follow this procedure to sign out of the Authorization Policy Manager Administration Console.

1. Click the **Sign Out** link located in the upper right corner of the Administration Console.

   Figure 3–5 is a screenshot of the Sign Out link.

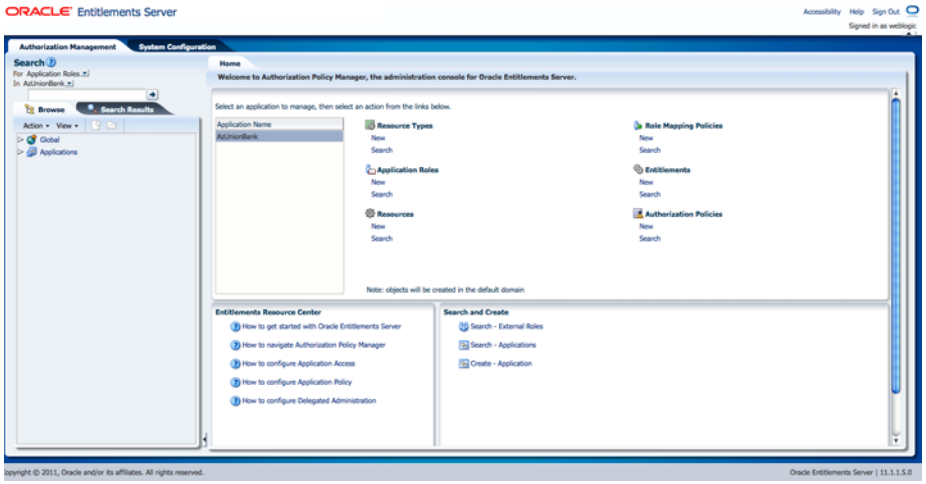
Figure 3–5 Administration Console Sign Out Link

2. Close the browser window.

3.4 Navigating the Administration Console

After a successful log in, the Authorization Policy Manager Administration Console is displayed with the Authorization Management Tab active. The Navigation Panel is on the left side and the Home area on the right side. Objects selected in the Navigation Panel are opened in tabs and displayed in the Home area. Figure 3–6 is a screenshot of the Administration Console after an administrative user has successfully signed in.

Figure 3–6 Oracle Entitlements Server Administration Console



The following list contains descriptions of the top-level items displayed in Figure 3–6. See the appropriate links for more information.

- Section 3.4.1, "Understanding the Main Tabs"
- **Section 3.4.2, "Using The Navigation Panel"**
- **Section 3.4.3, "The Home Area"**
- Section 3.4.4, "Online Help"

3.4.1 Understanding the Main Tabs

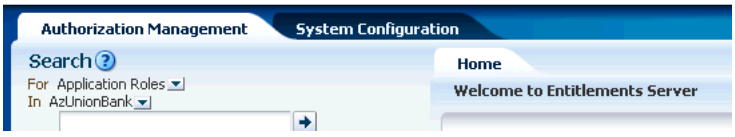
See the following sections for information on the organizational tabs used in the Administration Console. Each tab is comprised of a Navigation Panel and Home area.

- Section 3.4.1.1, "Authorization Management Tab"
- Section 3.4.1.2, "System Configuration Tab"

3.4.1.1 Authorization Management Tab

The Authorization Management tab is used to search and manage policy objects. This tab is active upon successful log in to the Administration Console. Figure 3–7 is a screenshot of the Authorization Management tab.

Figure 3–7 Authorization Management Tab

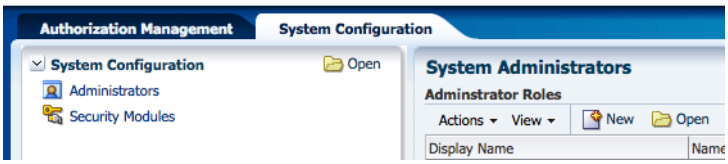


Under Authorization Management, the left side is the Navigation Panel and the right side is Home. The Home display changes based on what is selected from the Navigation Panel. For more information, see Section 3.4.2, "Using The Navigation Panel" and Section 3.4.3, "The Home Area."

3.4.1.2 System Configuration Tab

The System Configuration tab is used to manage administrative and system type objects for the Oracle Entitlements Server deployment. Figure 3–8 is a screenshot of an active System Configuration tab. The object selected in the Navigation Panel is displayed using tabs in the Home area.

Figure 3–8 System Configuration Tab



The following tasks are performed under System Configuration:

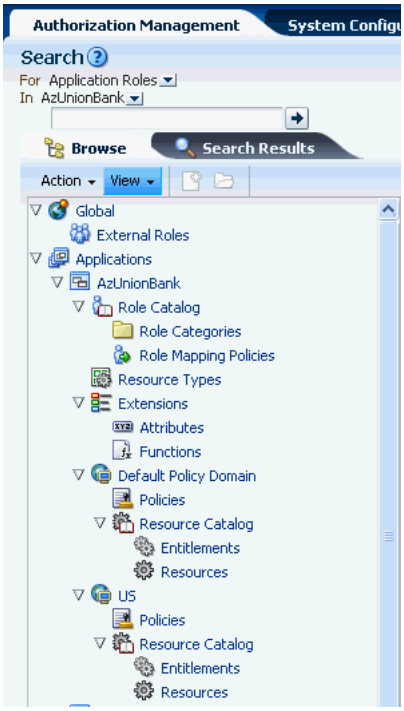
- Creating Security Modules
- Binding Security Modules to applications
- Managing system administrators (for example, creating additional system administrator roles, assigning users to system administrator roles, and assigning rights to system administrator roles)

For more information, see Chapter 8, "Managing System Configurations".

3.4.2 Using The Navigation Panel

The Navigation Panel is used to find security objects by browsing the Global or Applications information trees, or by conducting a simple search. It lists all Global and Application policy objects in a navigatable tree. You can browse the tree or display objects as Search Results based on defined search criteria. Figure 3–6 is a screenshot that displays the Navigation Panel with its nodes collapsed. Figure 3–9 displays the Navigation Panel with its nodes expanded and many policy objects in view.

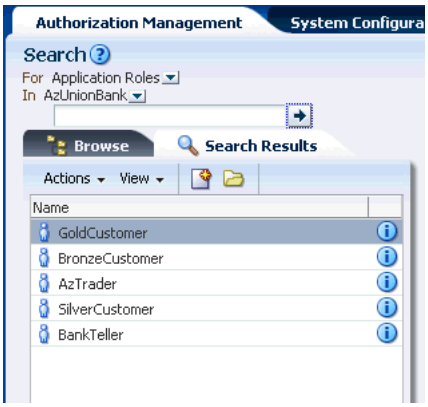
Figure 3–9 Navigation Panel Browse Tab with Nodes Expanded



The Navigation Panel contains, from top to bottom, the following elements:

- A pull-down list to select the policy object for a simple search. For more information, see Section 5.2, "Finding Objects with a Simple Search."
- A pull-down list to select the scope of a simple search. For more information, see Section 5.2, "Finding Objects with a Simple Search."
- A text box to enter the simple search string. The string is compared against both the Name and Display Name of policy objects; those that match are displayed in the Search Results tab.
- The **Browse** tab displays the following expandable and collapsible nodes:
  - The **Global** node collects global objects such as external roles.
  - The **Applications** node contains one or more Applications being managed by the administrator that is logged in. (Only Applications which the logged in user is authorized to access are displayed.) From any of those displayed, the administrator can access application-specific policy objects such as resource types, entitlements, resources, policies, and roles. For more information, see Chapter 8, "Managing System Configurations".
- The **Search Results** tab displays the results of the last simple search as seen in Figure 3–10.
- Action and View drop downs to select operations on the chosen policy object.

Figure 3–10 Navigation Panel Search Tab



From the Navigation Panel, there are two methods for displaying the **New** and **Open** options comprised in the Actions drop-down list.

- Locate the desired application, expand the node, and select the desired object. Click the Actions drop-down and select New.
- Locate the desired application, expand the node, and select the desired object. Right-click the object from the application node.

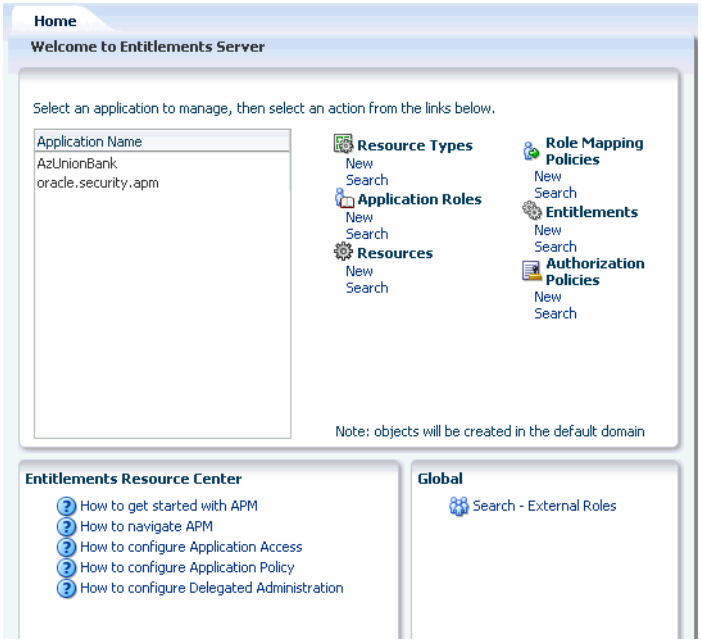
Select **New** to create a new object of the same type and select **Open** to display a search tab in the Home area. Double-clicking an object from the node also opens a Search tab in the Home area.

3.4.3 The Home Area

The Home area displays on the right side of the Navigation Panel and contains quick access links to New and Search screens for the most commonly used policy objects. As displayed in Figure 3–11, the Home area of the Administration Console is divided into the following sections.

- The **Application** area is the upper region of the Home area. The Application Name pane displays all applications available to the logged in user. To the right of this pane are links to screens for performing common operations such as creating new policy objects (entitlements, resources, resource types, application roles, and authorization policies) or searching defined policy objects.
- The **Global** section is the lower right region of the Home area. This section is for objects shared across all applications and includes external role search.
- The **Entitlements Resource Center** section is the lower left region of the Home area. It contains links to information regarding the most commonly used procedures.

Figure 3–11 The Home Area



3.4.4 Online Help

To get more information while using the Administration Console, click the Help link located in the upper right corner (as seen in Figure 3–5). A separate window opens. From this window you can access both the online help and an embedded version of this book in HTML. After the window displays, select either Oracle Entitlements Server Administration Console *Online Help* or *Administrator's Guide for* Oracle Entitlements Server from the drop-down Book list. The help topics link to the corresponding section of the embedded book as do the links in the book's Table of Contents.

4

Managing Policies and Roles

The Oracle Entitlements Server Administration Console is used to manage authorization policies and the policy objects from which they are created. This chapter contains the following sections:

4.1 Introducing Policy and Policy Object Management

Oracle Entitlements Server allows administrators to perform create, read, update, and delete (CRUD) operations on all policy and global objects. Tasks performed in the Administration Console typically require that an administrator identify an object (by browsing or searching), select it, and choose one of the operations available for it. Objects are organized into groupings that are displayed in the Navigation Panel: Global and Applications.

- Application objects include the objects used to create authorization policies (resources, application roles and the like). They apply to, and can only be used for policies within, the Application under which they are defined. The Applications node in the Navigation Panel is the branch under which all configured Applications (and their respective objects) are organized. This chapter contains information on managing Applications and their objects.
- Global objects include users, external roles, and system configurations for attribute retrievers, administrators and the like. These objects may apply to all configured Applications throughout the system. The Global node in the Navigation Panel is the branch under which all systemwide objects are organized. These obejcts are discussed in Chapter 8, "Managing System Configurations."

Note: Within Oracle Entitlements Server, external roles (and users) are read only; they are typically managed with a different tool, such as Oracle Identity Manager. For more information, see *Oracle Fusion Middleware System Administrator's Guide for Oracle Identity Manager*.

Oracle Entitlements Server supports the mapping of policies to individual users, External Roles, and Application Roles. However, mapping policies to Application Roles is recommended because of the following:

- Managing authorization based on grants to individual users and external roles can quickly become unmanageable as the number increases.
- If the identity management source changes (for example, when a move between development, test and production environments results in a new LDAP server), no changes to policy definitions are needed. All that is required is a re-mapping of the application roles to the users and external roles available in the target environment.

By default, all access to a resource is denied until an Authorization Policy is written and deployed that explicitly grants access action. If the Authorization Policy only grants an entitlement on a Resource to a role, the user must be statically assigned to it or a Role Mapping Policy must be written and deployed that assigns a user or a group to the defined role. If an Authorization Policy denies a previously granted entitlement, it takes precedence over the grant. Explicit DENY authorization policies cannot be overruled. A practical use of a DENY policy is to explicitly deny an entitlement to ensure that a user or group can never gain access to a specific resource.

4.2 Defining an Authorization Policy And Its Components

Defining a policy requires that the objects be created in a particular order. For example, a Resource can only be created after defining a Resource Type. A policy can be composed by following the sequence described below.

1. Create an Application.

   In the Navigation Panel, an Application should be created as the overall container for policies and related information that secure the components of a particular resource. You may create as many Applications as needed although it is recommended that only one is created for each application to be secured. For more information, see Section 4.5.1, "Managing Applications."

2. Create a Resource Type.

   A Resource Type specifies one or more resource attributes, and definitions of all possible valid actions that can be performed on a particular kind of resource. The actions can be standard actions (GET and POST to a URL) or custom actions on a business object (transfer to or from a bank account). Consider the following Resource Types and their valid actions:

   - A text file may support Read, Write, Copy, Edit, and Delete.
   - A checking account application may support deposit, withdrawal, view account balance, view account history, transfer to savings, and transfer from savings.

   Resource instances are created from Resource Types. Actions defined by the Resource Type are granted or denied when accessing a protected Resource instance created from the Resource Type.

   Note: A Resource instance is defined in a Policy Domain and references the Resource Type. For more information, see Section 4.5.3, "Managing Resources."

   For more information, see Section 4.5.2, "Managing Resource Types."

3. Instantiate a Resource from the Resource Type.

   A specific protected target (Resource) is instantiated from a Resource Type. A Resource represents a secured target (for example, an application) and is created under a Policy Domain in the Resource Catalog. If no Policy Domain is specified, it is created under the Default Policy Domain. For more information, see Section 4.5.3, "Managing Resources."

   Note: A Policy Domain is an optional object that is created for purposes of delegated administration and organization. See Chapter 9, "Delegating With Administrator Roles."

4. Build the Authorization Policy.

   This entails specifying the effect (GRANT or DENY), adding a user, group or role as the policy principal and the Resource and actions as the policy target. Optionally, you can add an Obligation or build a Condition. For more information, see Section 4.5.5, "Managing Authorization Policies."

4.3 Adding Fine-Grained Elements to an Authorization Policy

Section 4.2, "Defining an Authorization Policy And Its Components" documented the minimum components needed to create an authorization policy. The following fine-grained elements can be added to a simple policy.

- Entitlements

  An Entitlement associates an instantiated Resource with the applicable actions that can be performed on it. The set of actions for a Resource are a subset of the set of legal actions already defined in its corresponding Resource Type. For more information, see Section 4.5.4, "Managing Entitlements."

- Application Roles

  An Application Role can be assigned statically or dynamically to an enterprise user, group, or external role in an identity store, or another Application Role in the policy store. One target application may have several different Application Roles, with each role assigned a different set of privileges for more fine-grained access. For more information, see Section 4.5.6, "Managing Application Roles in the Role Catalog."

- Role Mapping Policy

  Membership to an Application Role can be granted dynamically with a Role Mapping Policy. An Application Role, referenced as a Principal in a Role Mapping Policy, could grant a user access to the defined resources but the Role Mapping Policy must be resolved before an authorization decision is reached. The resolution answers the question *Can the user requesting access be assigned this Application Role?* During runtime evaluation of a Role Mapping Policy, the following occurs:

  1. Based on the subject, a list of application roles is determined by retrieving static role membership and evaluating any applicable role mapping policies.
  2. Based on the subject and list of application roles, a list of Authorization Policies is evaluated to find any that might be applicable based on the grantee, target matching and constraints evaluation. The actions allowed on the Resource are defined by the Authorization Policy.

3. Final authorization decision is based on the “DENY overrides” combining algorithm.

For more information, see Section 4.5.7, "Managing Role Mapping Policies."

- A Condition can be added to a policy as a way of setting an additional contigency on the policy. It is applicable to either an Authorization Policy or a Role Mapping Policy. A Condition is written in the form of an expression that resolves to true or false and has one of the following outcomes:
  - If the expression resolves to true, the policy condition is satisfied and the effect defined in the PolicyRuleEntry is applicable.
  - If the expression does not resolve to true, the policy is not applicable.

  A Condition must be true for the policy to evaluate to true. Conditions can be complex combinations of boolean expressions that test the value of some user, resource, or system attribute or they can be custom Java evaluation functions that evaluate complex business logic. For more information, see Section 4.6, "Using the Condition Builder."
- An Obligation specifies optional information to be evaluated during the policy enforcement phase of authorization. The obligation is returned with the corresponding policy effect (GRANT or DENY). This information may or may not be taken into account during policy enforcement based on settings defined by the application. For example, the reason a request for access has been denied might be returned as an obligation. A different type of obligation might involve sending a message; for example, if a certain amount of money is withdrawn from a checking account, send a text message to the account holder's registered mobile phone. For more information, see Section 4.5.5, "Managing Authorization Policies."

4.4 Implementing An Authorization Policy Step by Step

In Section 2.4, "Implementing a Policy Use Case," several use cases for creating a policy are discussed. This section documents the step by step procedure to create an Authorization Policy (and the policy objects from which it is comprised) using the Administration Console. This procedure assumes you have installed Oracle Entitlements Server and a Java Security Module to protect an application.

1. Create an Application.

   The Application Name must match what is used in the application code. For example, create a `HelloOESworld` Application object to map to a `HelloOESworld` Application. See Section 4.5.1.1, "Creating an Application."

2. Create a Resource Type.

   The Resource Type Name must match what is used in the application code. For example, create a `Files` Resource Type object for use in collecting files that will be protected. Associate the *write* and *read* actions with the Resource Type. See Section 4.5.2.1, "Creating a Resource Type."

3. Create a Resource.

   A Resource Name must match what is used in the application code. Additionally, the Resource is created from the Resource Type. For example, create a `FinanceFile` Resource from the `Files` Resource Type. See Section 4.5.3.1, "Creating a Resource."

4. Create the Authorization Policy.

In the `HelloOESworld` Application, create an Authorization Policy. Add one or more Principals (Roles or Users), one or more targets (Resources or Entitlements) and confirm the actions for the target. Optional conditions or obligations can also be added before saving. See Section 4.5.5.1, "Creating an Authorization Policy."

5. Create a Security Module definition and bind it to the Application.

   This step defines the Security Module to which this Authorization Policy is distributed once binded. See Section 8.2, "Configuring Security Module Definitions."

6. Distribute the Authorization Policy to the Security Module.

   See Chapter 7, "Managing Policy Distribution."

4.5 Managing Policy Objects in An Application

The following sections describe how to manage policy objects specific to the Applications.

- Section 4.5.1, "Managing Applications"
- Section 4.5.2, "Managing Resource Types"
- Section 4.5.3, "Managing Resources"
- Section 4.5.4, "Managing Entitlements"
- Section 4.5.5, "Managing Authorization Policies"
- Section 4.5.6, "Managing Application Roles in the Role Catalog"
- Section 4.5.7, "Managing Role Mapping Policies"
- Section 4.5.8, "Managing a Role Category"
- Section 4.5.9, "Managing Attributes and Functions as Extensions"

4.5.1 Managing Applications

An Application is created as the overall container for policies and related artifacts that secure the components of a particular application. These artifacts include (but are not limited to) roles, resources, attributes and functions. You may create as many Application instances as needed although it is recommended that only one is created for each application to be secured. The following sections describe management operations on Application instances.

- Creating an Application
- Modifying an Application
- Deleting an Application

4.5.1.1 Creating an Application

To create an Application, proceed as follows:

1. Right-click **Applications** in the Navigation Panel and select **New** from the menu.

> **Note:** Alternately, click Create Application under Search and Create in the Home area.

An Untitled page with several tabs displays in the Home area. The General tab is active. You can only configure the Delegated Administrators and Policy Distribution details after the Application has been created. See Section 4.5.1.2, "Modifying an Application" for information.

2. Provide the following information for the application being created under the General tab.
   - **Display Name:** The Display Name is optional and case insensitive. Specifying a meaningful value, though, is recommended as it is displayed in the Administration Console and can be used as a search parameter.
   - **Name**: The name is required and case insensitive. It must match what is used in the application code.
   - **Description**: Although optional, it is recommended to provide useful information about the Application.
3. Select one of the following from the Save menu.
   - **Save and Close** saves the configuration, renames the tab with the value provided for the Application's Display Name and activates the Delegated Administrators and Policy Distribution tabs.
   - **Save and Create Another** saves the configuration to the information tree in the Navigation Panel but leaves the Untitled area open for you to create another Application.

4.5.1.2 Modifying an Application

To modify an Application, proceed as follows:

1. Expand the **Applications** node in the Navigation Panel.
2. Select the name of the Application to modify.
3. Right-click the Application name and select **Open** from the menu.

   Alternately, double-click the Application name. The Application page is displayed and the General tab, the Delegated Administrators tab and the Policy Distribution tab are all active.
4. Select the tab you want to modify or configure and see the appropriate section for parameter details.
   - **General** : Section 4.5.1.1, "Creating an Application"
   - **Delegated Administrators :** Chapter 9, "Delegating With Administrator Roles"
   - **Policy Distribution**: Chapter 7, "Managing Policy Distribution"
5. Apply or save as necessary.

4.5.1.3 Deleting an Application

To delete an Application instance, proceed as follows:

1. Find the Application to delete using an advanced search (as documented in Section 5.3.2, "Searching Applications").

   The Search Applications page is displayed.
2. Enter query parameters and click Search.

   The results are displayed.
3. Select the Application name from the results and click Delete.

4. Choose one of the following methods to search for the Application:

   A Delete Warning is displayed.

5. Click Delete.

   The Application is deleted.

> **Note:** Alternately, expand the Applications information tree in the Navigation Panel and double click the name of the Application to delete. The Application is displayed in the Home area. Click Delete in the upper right corner.

4.5.2 Managing Resource Types

Resource Types specify the full scope of traits for a particular kind of protected resource. It contains one or more resource attributes, and definitions of all possible valid actions that can be performed on the particular kind of resource. An *action* represents an activity or task in your business process that can be executed on a resource. Actions can be standard (GET and POST to a URL) or custom on a specific business object (transfer to or from a bank account). A Resource instance for a specific target is created from a Resource Type. The following sections describe management operations on Resource Types.

- Creating a Resource Type
- Modifying a Resource Type
- Deleting a Resource Type

4.5.2.1 Creating a Resource Type

To create a Resource Type, proceed as follows:

1. Display the page for creating a Resource Type by choosing from the following methods:

   - Expand the information tree in the Navigation Panel, right-click Resource Types under the particular Application in which the Resource Type will be created and select from the menu.
   - In the Home area, select the Application Name under which the Resource Type will be created and click New under Resource Types.

   An Untitled page is displayed in the Home area.

2. Provide the following information for the Resource Type.

   - **Display Name** : The display name is optional and case insensitive. Specifying a meaningful value, though, is recommended as it is displayed in the Administration Console and can be used as a search parameter.
   - **Name :** The name is required and case insensitive.
   - **Resource Finder** : An (optional) class that implements the `oracle.security.jps.service.policystore.entitymanager.ResourceFinder` interface. It allows resources managed outside of the Policy Store to be consumed. *(Reserved for future use.)*
   - **Description** : Although optional, it is recommended to provide useful information. The description string is case insensitive.

3. Add actions allowed by the Resource Type in the Actions section.

a. Click **New** to display the New Action dialog

b. Enter the name of the action.

   The string entered must match the actions for which your application is asking for authorization. If a Permission class is added, the action must be meaningful to it.

c. Click **Save**.

The Action list is updated with the new action.

4. Choose one of the following methods to add attributes to the Resource Type being created.

   - Drag and drop

     a. Use the Navigation Panel to list the Application's available attributes by performing a simple search on configured Resource instances. For more information, see Section 5.2, "Finding Objects with a Simple Search".

     b. Drag and drop attributes from the **Search Results** tab into the area labeled **Attributes**.

   - Find Existing Attribute dialog

     a. In the Attributes section, click **Add** to display the Find Existing Attribute dialog.

     b. Select the attribute **Type** from the list.

     c. Enter an (optional) string to match in the **Search** text box.

     d. Click the arrow icon next to the Search text box to begin the search.

     e. Select the attributes to add and click **Add**.

        Use **Ctrl+click** to select multiple items from the list.

   These attributes are used when instantiating a Resource. See Section 4.5.3.1, "Creating a Resource."

5. Configure the remaining fields.

   The selection changes according to the Resource Type being created.

   - Supports Resource Hiearchy - Select Yes or No to set the Resource Type as hierarchical. This means the following when the Resource Type is used to instantiate a Resource:
     - A policy applicable to a Resource created from a hierarchical Resource Type is also applicable to Resources that are its children.
     - Any attribute defined for a Resource created from a hierarchical Resource Type is inherited by Resources that are its children.
   - Resource Name Delimiter - Only valid when Supports Resource Hierarchy is enabled. The default delimiter is `Slash (/)`.
   - Evaluation Logic - Evaluation logic for a Resource Type can be either a permission class or a default matching algorithm. Define the algorithm here or the permission class below.
   - Permission Class - When the evaluation logic is a Permission class, a class name is required and is case sensitive.
   - Action Name Delimiter - The specified character is used to separate actions in a list when the Resource Type represents a permission.

- All Action Keyword - If the policy's target contains the defined keyword as an action, the policy will match any action passed in with the authorization request. For example, assume that this parameter is set to `ANY` and you create the following policy:

  ```
  GRANT user "Michael" action:"ANY" on resource:"Resource1
  ```

  The decision for authorization requests like *Can Michael do 'write' on Resource1?* or *Can Michael do 'transfer' on Resource1?* will return ALLOW. The use of this parameter allows you to create a single Authorization Policy that would be applicable to any valid action for that Resource Type.

6. Select one of the following from the Save menu.
   - **Save and Close** saves the configuration, renames the tab with the value provided for the Application's Display Name and activates the Delegated Administrators and Policy Distribution tabs.
   - **Save and Create Another** saves the configuration to the information tree in the Navigation Panel but leaves the Untitled area open for you to create another Application.

4.5.2.2 Modifying a Resource Type

To modify a Resource Type, proceed as follows:

1. Choose from the following methods to display the desired Resource Type.
   - Expand the information tree in the Navigation Panel to find the Resource Types node under the appropriate Application and double click it. A search dialog opens in the Home area. For information about searching in the Home area, see Section 5.3.3, "Searching Resource Types."
   - Search for Resource Types using the Navigation Panel's search function and double-click the Resource Type name in the Search Results tab. For information about searching in the Navigation Panel, see Section 5.2, "Finding Objects with a Simple Search."
   - In the Home area, select the Application Name under which the Resource Type was created and click **Search** under **Resource Types**. A search dialog opens in the Home area. For information about searching in the Home area, see Section 5.3.3, "Searching Resource Types."

   When the correct Resource Type name is displayed, select it and click Open to display the details.
2. Modify as necessary.
3. Click **Apply**.

4.5.2.3 Deleting a Resource Type

To delete a Resource Type, proceed as follows:

1. Choose from the following methods to delete the desired Resource Type.
   - Expand the information tree in the Navigation Panel to find the Resource Types node under the appropriate Application and double click it. A search dialog opens in the Home area. Enter criteria for the lookup and click Search. Select the appropriate Resource Type from the search results and click Delete. For information about searching in the Home area, see Section 5.3, "Finding Objects with an Advanced Search."

- Search for Resource Types using the Navigation Panel's search function and double-click the Resource Type name in the Search Results tab. A search dialog opens in the Home area. Enter criteria for the search and click Search. Select the appropriate Resource Type from the search results and click Delete. For information about searching in the Navigation Panel, see Section 5.2, "Finding Objects with a Simple Search."
- Select the appropriate Application Name in the Home area and click Search under Resource Types. A search dialog opens in the Home area. Enter criteria for the lookup and click Search. Select the appropriate Resource Type from the search results and click Delete. For information about searching in the Home area, see Section 5.3, "Finding Objects with an Advanced Search."

A Delete Warning is displayed.

2. Click Delete.

   The Resource Type is deleted.

4.5.3 Managing Resources

A Resource represents a specific, secured target in a protected application. Each Resource belongs to a defined Resource Type and can represent software components managed by a container (URLs, EJBs, JSPs) or business objects in an application (reports, transactions, revenue charts).

Note: Resources can be hierarchical (in that the child resource inherits attributes from parent resources) or non-hierarchical. When organized in a hierarchy (root down), you can add new attributes to the parent resources or overwrite any existing attributes that are inherited.

The following sections describe management operations on Resources.

- Creating a Resource
- Modifying a Resource
- Deleting a Resource

4.5.3.1 Creating a Resource

To create a Resource, proceed as follows

1. Display the page for creating a Resource by choosing from the following methods:
   - Navigate to the Resource Catalog by expanding the applicable Policy Domain node in the appropriate Application node using the Navigation Panel. Right-click **Resources** from the Resource Catalog node and select **New** from the menu.
   - Select the Application under which you will create the Resource instance from the Home area and click **New** under **Resources**.

   Note: This option creates the Resource in the Application's Default Policy Domain.

   An Untitled page is displayed in the Home area.

2. Provide the following information.
   - **Resource Type**: Select from the list. This defines what is displayed in the Instance Attributes and Overwrites table.
   - **Display Name** : The display name is optional and case insensitive. Specifying a meaningful display name is recommended since it is displayed in the Administration Console, and provides extra information to help administrators identify objects.
   - **Name :** The name is required and case insensitive. At runtime, this is the string the application passes to determine whether a user is authorized to access this Resource.
   - **Description** : Although optional, it is recommended to provide useful information about the entitlement. The description string is case insensitive.
3. Add or remove the attributes for this Resource from those displayed in the Instance Attributes and Overwrites dialog.

   The Overwrites dialog is displayed only in the case of hierarchical Resources.
4. Select the attributes from the list (use **Ctrl+click** to select multiple items from the list) and click **Add**.
5. Select one of the following from the Save menu.
   - **Save and Close** saves the configuration, renames the tab with the value provided for the Application's Display Name and activates the Delegated Administrators and Policy Distribution tabs.
   - **Save and Create Another** saves the configuration to the information tree in the Navigation Panel but leaves the Untitled area open for you to create another Application.

4.5.3.2 Modifying a Resource

To modify a resource, proceed as follows:

1. Choose from the following methods to display the desired Resource.
   - Navigate to the Resource Catalog by expanding the applicable Policy Domain node in the appropriate Application node using the Navigation Panel and double click it. A search dialog opens in the Home area. For information about searching in the Home area, see Section 5.3.6, "Searching Resources."
   - Search for Resources using the Navigation Panel's search function and double-click the Resource name in the Search Results tab. For information about searching in the Navigation Panel, see Section 5.2, "Finding Objects with a Simple Search."
   - In the Home area, select the Application Name under which the Resource Type was created and click **Search** under **Resources**. A search dialog opens in the Home area. This search dialog will only query the Default Policy Domain. For information about searching in the Home area, see Section 5.3.6, "Searching Resources."

   When the correct Resource name is displayed, select it and click Open to display the details.
2. Modify the Resource as necessary.
3. Click **Apply**.

4.5.3.3 Deleting a Resource

To delete a Resource, proceed as follows:

1. Choose from the following methods to delete the desired Resource.
   - Navigate to the Resource Catalog by expanding the applicable Policy Domain node in the appropriate Application node using the Navigation Panel and double click it. A search dialog opens in the Home area. Enter criteria for the lookup and click Search. Select the appropriate Resource from the search results and click Delete. For information about searching in the Home area, see Section 5.3, "Finding Objects with an Advanced Search."
   - Search for Resources using the Navigation Panel's search function and double-click the Resource name in the Search Results tab. Select the appropriate Resource from the search results and click Delete. For information about searching in the Navigation Panel, see Section 5.2, "Finding Objects with a Simple Search."
   - In the Home area, select the Application Name under which the Resource was created and click **Search** under **Resources**. A search dialog opens in the Home area. Enter criteria for the lookup and click Search. (This search queries only in the Default Policy Domain.) Select the appropriate Resource from the search results and click Delete. For information about searching in the Home area, see Section 5.3, "Finding Objects with an Advanced Search."

   A Delete Warning is displayed.
2. Click Delete.

   The Resource is deleted.

4.5.4 Managing Entitlements

After instantiating a Resource, define the actions that can be performed on it in an Entitlement. The actions are defined using the set of legal actions defined in the Resource's parent Resource Type. The following sections describe management operations on Entitlements.

- Creating an Entitlement
- Modifying an Entitlement
- Deleting an Entitlement

Note: An Entitlement may be created if there are plans to use the same list of Resource and Action pairs in multiple policies. Otherwise, the Resource and Action pair itself can be directly specified as a target when you create an Authorization Policy. See Section 4.5.5, "Managing Authorization Policies" for more information.

4.5.4.1 Creating an Entitlement

To create an Entitlement, proceed as follows.

1. Display the page for creating an Entitlement by choosing from the following methods:
   - Navigate to the Resource Catalog by expanding the applicable Policy Domain node in the appropriate Application node using the Navigation Panel.

Right-click **Entitlements** from the Resource Catalog node and select **New** from the menu.

- In the Home area, select the Application Name under which the Entitlement will be created and click **New** from **Entitlements**.

An Untitled page is displayed in the Home area.

2. Provide the following information.
   - **Display Name** : The display name is optional and case insensitive. Specifying a meaningful display name is recommended since it is displayed in the Administration Console, and provides extra information to help administrators identify objects.
   - **Entitlement Name :** The name is required and case insensitive. At runtime, this is the string the application passes to determine whether a user is authorized to access this Resource.
   - **Description** : Although optional, it is recommended to provide useful information about the entitlement. The description string is case insensitive.
3. Choose one of the following methods to add Resources to the Entitlement.
   - Drag and drop
     a. Use the Navigation Panel to list the Application's available Resources by performing a search on Resource instances. The Resources must be searched from the same Policy Domain in which the Entitlement is being created. For more information, see Section 5.2, "Finding Objects with a Simple Search".
     b. Drag and drop Resources from the **Search Results** tab into the area labeled **Resources**.
   - Add Targets pop up search
     a. Click **Add** in the **Targets** section.

        The **Add Targets** dialog displays. This will search in the current Policy Domain.
     b. Search for available targets by entering a string.

        The resources matching the query are displayed in **Search Results**. If no search string was entered, a list of all objects of the specified type is returned.
     c. Select your choice(s) and click Add Selected.

        The Target(s) are added to the Selected Targets. Use **Ctrl+click** to select multiple items from the list.

     Note: Alternately, you can click the Resource Expression link under the Resources tab, select a Resource Type, enter a string expression and click Add to Targets. This will search for targets, using the defined criteria, dynamically at runtime. All Resources that belong to the selected Resource Type that contain the string expression are returned, within the context of the administrator privileges.

     d. Click **Add Targets**.
4. Add actions to the Resources as follows:

a. Select an added resource from the Resources list to display the resource details in the **Resource Details** section.

b. Expand the selected row to see the range of actions.

   Only the actions allowed for the type of the selected resource are available in this area.

c. Check the desired actions for the Resource in the **Actions** section.

d. Repeat this procedure for each Resource you have added to the Entitlement being created.

5. Select one of the following from the Save menu.

   - **Save and Close** saves the configuration, renames the tab with the value provided for the Application's Display Name and activates the Delegated Administrators and Policy Distribution tabs.
   - **Save and Create Another** saves the configuration to the information tree in the Navigation Panel but leaves the Untitled area open for you to create another Application.

4.5.4.2 Modifying an Entitlement

To modify an entitlement, proceed as follows:

1. Choose from the following methods to display the desired Entitlement.

   - Navigate to the Resource Catalog by expanding the applicable Policy Domain node in the appropriate Application node using the Navigation Panel and double click it. A search dialog opens in the Home area. For information about searching in the Home area, see Section 5.3.7, "Searching Entitlements."
   - Search for Entitlements using the Navigation Panel's search function and double-click the Entitlement name in the Search Results tab. For information about searching in the Navigation Panel, see Section 5.2, "Finding Objects with a Simple Search."
   - In the Home area, select the Application Name under which the Entitlement was created and click **Search** under **Entitlements**. A search dialog opens in the Home area. For information about searching in the Home area, see Section 5.3.7, "Searching Entitlements."

   When the correct Entitlement name is displayed, select it and click Open to display the details.

2. Modify the entitlement as necessary.

3. Click **Apply**.

4.5.4.3 Deleting an Entitlement

To delete a Resource, proceed as follows:

1. Choose from the following methods to delete the desired Entitlement.

   - Expand the information tree in the Navigation Panel to find the Entitlement node under the appropriate Application's Resource Catalog and double click it. A search dialog opens in the Home area. Enter criteria for the lookup and click Search. Select the appropriate Resource from the search results and click Delete. For information about searching in the Home area, see Section 5.3, "Finding Objects with an Advanced Search."

- Search for Entitlements using the Navigation Panel's search function and double-click the Entitlement name in the Search Results tab. Select the appropriate Resource from the search results and click Delete. For information about searching in the Navigation Panel, see Section 5.2, "Finding Objects with a Simple Search."
- In the Home area, select the Application Name under which the Entitlement was created and click **Search** under **Entitlements**. A search dialog opens in the Home area. Enter criteria for the lookup and click Search. Select the appropriate Resource from the search results and click Delete. For information about searching in the Home area, see Section 5.3, "Finding Objects with an Advanced Search."

A Delete Warning is displayed.

2. Click Delete.

   The Entitlement is deleted.

4.5.5 Managing Authorization Policies

The Authorization Policy is the mechanism that defines the access rights (Grant/Deny). A user, an Application Role or an External Role is *granted* the rights of the policy. An Authorization Policy must have:

- At least one principal which can be a user, External Role or Application Role. Code sources are not allowed as a principal.
- At least one target that can be a Resource and Action association (created within the policy) or an Entitlement (created outside the policy and added to it) but not both.
- A defined effect of PERMIT or DENY.

> **Note:** Entitlement-based policies correspond closely with business functions. They are recommended in cases in which a business function considers securing a collection of resources; an entitlement can be used in one or more grants.

The following sections describe management operations on Authorization Policies.

- Creating an Authorization Policy
- Modifying an Authorization Policy
- Deleting an Authorization Policy

4.5.5.1 Creating an Authorization Policy

To create a policy, proceed as follows:

1. Display the page for creating a policy by choosing one of the following methods:
   - Navigate to the Policy Domain under the appropriate Application node in the Navigation Panel and expand it. Right-click **Authorization Policies** from the Resource Catalog node and select **New** from the menu.
   - In the Home area, select the Application Name under which the Authorization Policy will be created and click **New** from **Authorization Policies**. (When using this option, the policy will be created in the Default Policy Domain.)

An Untitled page is displayed in the Home area.

2. Provide the following information.
   - **Effect**: Select **Permit** if the policy will grant rights or **Deny** if the policy will deny rights.
   - **Display Name** : The display name is optional and case insensitive. Specifying a meaningful display name is recommended since it is displayed in the Administration Console, and provides extra information to help administrators identify objects.
   - **Name :** The name is required and case insensitive. At runtime, this is the string the application passes to determine whether a user is authorized to access this Resource.
   - **Description** : Although optional, it is recommended to provide useful information about the entitlement. The description string is case insensitive.
3. Choose one of the following methods to add Principals to the Authorization Policy.
   - Drag and drop
     a. Use the Navigation Panel to list the Application's available Principals by performing a search on Users, External Roles or Application Roles. For more information, see Section 5.2, "Finding Objects with a Simple Search".
     b. Drag and drop Principals from the **Search Results** tab into the area labeled **Principals**.
     c. Select Any or All.

        If Any, the user must match at least one of the specified principals. For example, if the principals are roles, the user must be a member of at least one of the roles for the Authorization Policy to apply. If All, the user must match all of the specified principals. For example, if the principals are roles, the user must be a member of all of them for the Authorization Policy to apply.
   - Add Principals pop-up search

     For details on how to use the pop-up search box, see Section 5.1, "Searching with the Administration Console."
     a. Click **Add** in the **Principals** section.

        The **Add Principals** dialog displays.
     b. Select the appropriate tab to search for available Principals.

        Options are Application Roles, External Roles and Users. You can navigate between tabs and add as many selected Principal types as desired.
     c. Search for the available Principals by entering a string.

        The Principals matching the query are displayed in **Search Results**.
     d. Select your choice(s) and click Add Selected.

        The Principal(s) are added to the Selected Principals. Use **Ctrl+click** to select multiple items from the list.
     e. Click **Add Principals**.
     f. Select Any or All.

If Any, the user must match at least one of the specified principals. For example, if the principals are roles, the user must be a member of at least one of the roles for the Authorization Policy to apply. If All, the user must match all of the specified principals. For example, if the principals are roles, the user must be a member of all of them for the Authorization Policy to apply.

4. Choose one of the following methods to add Targets to the Authorization Policy.

   This step adds either Resource and action associations or Entitlements or both to the Authorization Policy.

   - Drag and drop

     a. Use the Navigation Panel to list the Application's available Resources or Entitlements by performing a search. (Be sure to look for these objects in the same Policy Domain to which you are adding the Authorization Policy.) For more information, see Section 5.2, "Finding Objects with a Simple Search".

     b. Drag and drop one or more Resources or Entitlements from the **Search Results** tab into the area labeled **Targets**. Expanding the added object in **Targets** allows you to associate an action with it.

   - Add Targets pop up search

     For details on how to use the pop-up search box, see Section 5.1, "Searching with the Administration Console."

     a. Click **Add** in the **Targets** section.

        The **Add Targets** dialog displays.

     b. Select the appropriate tab to search for available Targets.

        Options are Entitlements and Resources. You can navigate between tabs and add as many selected Targets as desired.

     c. Search for available targets under the Entitlements tab by entering a string.

        The resources matching the query are displayed in **Search Results**. If no search string was entered, a list of all objects of the specified type is returned.

     d. Select your choice(s) and click Add Selected.

        The Target(s) are added to the Selected Targets. Use **Ctrl+click** to select multiple items from the list.

     e. Search for available targets under the Resources tab by entering a string.

        The resources matching the query are displayed in **Search Results**. If no search string is entered, a list of all objects of the specified type is returned.

        Alternately, you can click the Resource Expression link under the Resources tab, select a Resource Type, enter a string expression and click Add to Targets. This will search for targets, using the defined criteria, dynamically at runtime. All Resources that belong to the selected Resource Type that contain the string expression are returned, within the context of the administrator privileges.

     f. Click **Add Targets**.

5. Select the **Conditions** tab to add a condition.

   For more information, see Section 4.6, "Using the Condition Builder."

6. Select the **Obligations** tab.

   An Authorization Policy may have zero, one or more Obligations.

   a. Click **New** to display the New Obligation dialog.

   b. Provide a Name and an (optional) Display Name and Description for the New Obligation and click Add.

   c. Click **New** in the Attributes section to add an obligation attribute.

      An Obligation has a set of attributes. Each attribute is a name-value pair. The value can be either static or the value of a previously defined attribute. Each obligation should have at least one attribute. See Section 4.5.9, "Managing Attributes and Functions as Extensions" for information.

   d. Provide a Name for the attribute in the New Obligation Attribute dialog.

      If the obligation attribute is static, select either String, Integer, Boolean, Date or Time for **Data Type** and provide a Value. If the obligation is an attribute, select **Attribute** for Data Type and choose from the list of predefined attributes.

   e. Click **Add**.

7. Click **Save** to save the Authorization Policy.

4.5.5.2 Modifying an Authorization Policy

To modify a policy, proceed as follows:

1. Choose from the following methods to display the desired Authorization Policy.

   - Expand the information tree in the Navigation Panel to find the Authorization Policies node under the appropriate Application's Policy Domain and double click it. A search dialog opens in the Home area. For information about searching in the Home area, see Section 5.3.8, "Searching Authorization Policies."
   - In the Home area, select the Application Name under which the Authorization Policy was created and click **Search** under **Authorization Policies**. A search dialog opens in the Home area. For information about searching in the Home area, see Section 5.3.8, "Searching Authorization Policies."

   When the correct Authorization Policy is displayed, select it and click Open to display the details.

2. Modify the policy as necessary.

   - Select the Principal to modify.

     For more information, see Section 4.5.5.1, "Creating an Authorization Policy."

   - Select (or expand) the Target to modify.

     For more information, see Section 4.5.5.1, "Creating an Authorization Policy."

   - Click the Conditions tab to edit conditions.

     For more information, see Section 4.6, "Using the Condition Builder."

   - Click the Obligations tab to modify the Obligation or its attributes.

- To modify the obligation, click **Edit** from the Obligations table, make changes in the displayed dialog and click **Update**.
- To modify an attribute, select the attribute from the Attibutes table and click Edit. Make changes in the displayed dialog and click Update.
- To delete the Obligation, select it in the Obligations table and click Remove.

3. Click **Apply**.

4.5.5.3 Deleting an Authorization Policy

To delete an Authorization Policy, proceed as follows:

1. Choose from the following methods to display the Authorization Policy search screen.
   - Expand the information tree in the Navigation Panel to find the Authorization Policies node under the appropriate Application's Policy Domain, right-click it and select Open. A search dialog opens in the Home area.
   - In the Home area, select the Application Name under which the Authorization Policy was created and click **Search** under **Authorization Policies**. A search dialog opens in the Home area.

   For information about searching in the Home area, see Section 5.3, "Finding Objects with an Advanced Search."
2. Enter criteria for the lookup and click Search.
3. Select the appropriate Authorization Policy from the search results and click Delete.

4.5.6 Managing Application Roles in the Role Catalog

Application Roles are defined at the Application level (thus, its name). An Application Role can be assigned to an enterprise user, group, or role in an identity store, or another Application Role in the policy store. One target application may have several different roles, with each assigned a different set of privileges for more fine-grained authorization. Membership can be granted statically or dynamically with a Role Mapping Policy.

> **Note:** A Role Mapping Policy assigns the role to subjects and an Authorization Policy defines the role's access rights.

You can use Application Roles to control access by establishing relationships with the following procedure:

1. Define Application Roles to represent the functional roles users have in the application.
2. Map each Application Role to External Roles or individual Users.
3. Create Authorization Policies to provide the level of access rights (Permit/Deny) required to meet the goals of the Application Roles.
4. Add the Application Role as a Principal to one or more Authorization Policies.

Application Roles use role inheritance and hierarchy. The inheritance pattern is such that a subject assigned to a role (using a Role Mapping Policy or static role

assignments) also inherits any child roles if it is not prohibited by Role Mapping Policies. When an Application Role is referenced as a policy principal, access to the resource for all users assigned to the role is governed by the policy. The following sections describe management operations on Application Roles.

- Creating an Application Role
- Modifying an Application Role
- Mapping External Roles to an Application Role
- Mapping an External User to an Application Role
- Deleting an Application Role or Removing External Role Mappings

4.5.6.1 Creating an Application Role

The following procedure describes the steps to create a new Application Role. You are not required to add members to the role at the same time and can return to the saved role later. To create an Application Role, proceed as follows:

1. Display the page for creating an Application Role by choosing one of the following methods:
   - Navigate to the Role Catalog under the appropriate Application node in the Navigation Panel. Right-click the **Role Catalog** node and select **New** from the menu.
   - In the Home area, select the Application Name under which the Application Role will be created and click **New** from **Application Roles**.

   An Untitled page with four tabs is displayed in the Home area: General (active), Application Role Hierarchy, External Role Mapping and External User Mapping.
2. Provide the following information under the **General** tab.
   - **Display Name** : The display name is optional and case insensitive. Specifying a meaningful display name is recommended since it is displayed in the Administration Console, and provides extra information to help administrators identify objects.
   - **Role Name :** The name is required and case insensitive. At runtime, this is the string the application passes to determine whether a user is authorized to access this Resource.
   - **Description** : Although optional, it is recommended to provide useful information about the entitlement. The description string is case insensitive.
   - Role Category : A Role Category is a tag you can assign to a role for ease of management. See Section 4.5.8, "Managing a Role Category."
3. Click **Save**.

   The page is renamed to match the entry provided for Role Name and the Application Role Hierarchy, External Role Mapping and External User Mapping tabs become active. At this point, you can create a policy with this Application Role as the Principal or find a policy with this Application Role as the Principal by clicking Create Policies or Find Policies, respectively. To define the Application Role Hierarchy continue to the next step.
4. Optionally, select the **Application Role Hierarchy** tab to define from which roles this Application Role will inherit permissions (Inherits) and for which roles this Application Role will define permissions (Is Inherited By). Hierarchy is not

required but if you choose to define it, the following example sub procedure is specific to the former option.

a. Click **Inherits**.

b. Click **Add**.

c. Select the radio button that corresponds to the role to which you are adding the hierarchy.

   When you add roles to the hierarchy, you can either add the roles to the role under which you are working or to a role that you can select in the Application Role Hierarchy table.

d. Complete the criteria fields in the **Add a Role** dialog and click **Search**.

   The results display in the **Search Results** table. Empty strings fetch all roles.

e. Select the role from which this role will inherit permissions in the **Search Results** table.

   Use **Ctrl+click** to select multiple roles.

f. Click **Add**.

   The selected roles display in the **Application Role Hierarchy** tab, and the Application Role inherits permissions from them.

For information about external role mapping, see Section 4.5.6.3, "Mapping External Roles to an Application Role." For information about external user mapping, see Section 4.5.6.4, "Mapping an External User to an Application Role."

4.5.6.2 Modifying an Application Role

To modify or view an Application Role, proceed as follows:

1. Choose from the following methods to display the desired Application Role.

   - Expand the information tree in the Navigation Panel to find the Role Catalog node under the appropriate Application, right-click it and select Open. A search dialog opens in the Home area. Enter criteria for the lookup and click Search.
   - In the Home area, select the Application Name under which the Application Role was created and click **Search** under **Application Roles**. A search dialog opens in the Home area. Enter criteria for the lookup and click Search.

   When the correct Application Role is displayed, select it and click Open to display the details in the Home area. For information about searching in the Home area, see Section 5.3, "Finding Objects with an Advanced Search."

2. Select the tab that contains the parameters you want to modify and click Add.

   For information on the available tabs, see:

   - Application Role Hierarchy : Creating an Application Role
   - External Role Mapping : Mapping External Roles to an Application Role
   - External User Mapping : Mapping an External User to an Application Role

4.5.6.3 Mapping External Roles to an Application Role

To map external roles to an application role, proceed as follows:

1. Choose from the following methods to display the desired Application Role.

- Expand the information tree in the Navigation Panel to find the Role Catalog node under the appropriate Application and double click it. A search dialog opens in the Home area. For information about searching in the Home area, see Section 5.3, "Finding Objects with an Advanced Search."
- Search for Application Roles using the Navigation Panel's search function and double-click the Application Role name in the Search Results tab. For information about searching in the Navigation Panel, see Section 5.2, "Finding Objects with a Simple Search."
- In the Home area, select the Application Name under which the Application Role was created and click **Search** under **Application Roles**. A search dialog opens in the Home area. For information about searching in the Home area, see Section 5.3, "Finding Objects with an Advanced Search."

When the correct Application Role is displayed, select it and click Open to display the details in the Home area.

2. Select the **External Role Mapping** tab.
3. Click **Add** to display the **Add a Role** dialog.
4. Complete the query fields in the **Add a Role** dialog and click **Search**.

   Empty strings fetch all roles. The results display in the **External Role Search** table.
5. Select the external role to map to by clicking its name in the table.

   Use **Ctrl+click** to select multiple roles.
6. Click **Map Roles**.

   The selected roles display in the **External Role Mapping** tab.

4.5.6.4 Mapping an External User to an Application Role

To map an external user to an application role, proceed as follows:

1. Choose from the following methods to display the desired Application Role.
   - Expand the information tree in the Navigation Panel to find the Role Catalog node under the appropriate Application and double click it. A search dialog opens in the Home area. For information about searching in the Home area, see Section 5.3, "Finding Objects with an Advanced Search."
   - Search for Application Roles using the Navigation Panel's search function and double-click the Application Role name in the Search Results tab. For information about searching in the Navigation Panel, see Section 5.2, "Finding Objects with a Simple Search."
   - In the Home area, select the Application Name under which the Application Role was created and click **Search** under **Application Roles**. A search dialog opens in the Home area. For information about searching in the Home area, see Section 5.3, "Finding Objects with an Advanced Search."

   When the correct Application Role is displayed, select it and click Open to display the details in the Home area.
2. Select the **External Users Mapping** tab.
3. Click **Add** to display the **Add a User** dialog.
4. Complete the query fields in the **Add a User** dialog and click **Search**.

   Empty strings fetch all roles. The results display in the **External User Search** table.

5. Select the user to map by selecting its name in the table.

   Use **Ctrl+click** to select multiple roles.

6. Click **Map Users**.

   The selected roles display in the **External User Mapping** tab.

4.5.6.5 Deleting an Application Role or Removing External Role Mappings

To delete an Application Role or remove External Role Mapping from an Application Role, proceed as follows:

1. Choose from the following methods to display the desired Application Role.
   - Expand the information tree in the Navigation Panel to find the Role Catalog node under the appropriate Application, right-click it and select Open. A search dialog opens in the Home area. Enter criteria for the lookup and click Search.
   - In the Home area, select the Application Name under which the Application Role was created and click **Search** under **Application Roles**. A search dialog opens in the Home area. Enter criteria for the lookup and click Search.

   For information about searching in the Home area, see Section 5.3, "Finding Objects with an Advanced Search."

2. Select the Application Role in the Search Results table and:
   - Click Delete to remove the role.
   - Select the appropriate mapping in the External Role Mapping table and click Remove.

4.5.7 Managing Role Mapping Policies

Membership to an Application Role can be granted statically or dynamically with a Role Mapping Policy. An Application Role, referenced in a Role Mapping Policy, could grant a user access to the defined resources. The following sections describe management operations on Role Mapping Policies.

- Creating a Role Mapping Policy
- Modifying a Role Mapping Policy
- Deleting a Role Mapping Policy

4.5.7.1 Creating a Role Mapping Policy

To create a Role Mapping Policy, proceed as follows:

1. Display the page for creating a Role Mapping Policy by choosing one of the following methods:
   - Navigate to the appropriate Application node in the Navigation Panel and expand the Role Catalog branch. Right-click **Role Mapping Policies** and select **New** from the menu.
   - In the Home area, select the Application Name under which the Role Mapping Policy will be created and click **New** from **Role Mapping Policies**.

   An Untitled page is displayed in the Home area.

2. Provide the following information.

- **Effect**: Select **Permit** if the policy will grant rights or **Deny** if the policy will deny rights.
- **Display Name** : The display name is optional and case insensitive. Specifying a meaningful display name is recommended since it is displayed in the Administration Console, and provides extra information to help administrators identify objects.
- **Name :** The name is required and case insensitive.
- **Description** : Although optional, it is recommended to provide useful information about the policy. The description string is case insensitive.

3. Choose one of the following methods to add Application Roles.
   - Drag and drop
     a. Use the Navigation Panel to list the Application's available Application Roles by performing a search. For more information, see Section 5.2, "Finding Objects with a Simple Search".
     b. Drag and drop Application Roles from the **Search Results** tab into the area labeled **App Role**.
   - Add Application Roles dialog
     a. Click **Add** in the **App Role** section.

        The **Search Application Roles** dialog displays.
     b. Search for the available Application Roles by entering a string.

        The resources matching the query are displayed in **Search Results**.
     c. Select the principals to add and click **Add Application Roles**.

        Use **Ctrl+click** to select multiple items from the list.

Note: For this release, this dialog displays the Search Principals title and Add Principals button.

---

4. Choose one of the following methods to add Principals.
   - Drag and drop
     a. Use the Navigation Panel to list the Application's available Users and External Roles by performing a search. For more information, see Section 5.2, "Finding Objects with a Simple Search".
     b. Drag and drop Users and External Roles from the **Search Results** tab into the area labeled **Principals**.
   - Add Principals dialog
     a. Click **Add** in the **Principals** section.

        The **Search Principals** dialog displays.
     b. Search for the available Principals (in this case, Users or External Roles) by entering a string.

        The resources matching the query are displayed in **Search Results**.
     c. Select the principals to add and click **Add Principals**.

        Use **Ctrl+click** to select multiple items from the list.

5. Optionally, choose one of the following methods to add Resources (also referred to as Targets).
   - Drag and drop
     a. Use the Navigation Panel to list the Application's available Resources by performing a search. For more information, see Section 5.2, "Finding Objects with a Simple Search".
     b. Drag and drop one or more Resources from the **Search Results** tab into the area labeled **Resources**.
   - Add Targets pop up search
     a. Click **Add** in the **Resources** section.

        The **Add Targets** dialog displays.
     b. Choose the Policy Domain that contains the Resource (if applicable).
     c. Enter a string and click Search.

        The resources matching the query are displayed in **Search Results**. If no search string was entered, a list of all objects of the specified type is returned.
     d. Select the appropriate Targets to add and click **Add Selected**.

        The Target(s) are added to the Selected Targets. Use **Ctrl+click** to select multiple items from the list.
     e. Click the Resource Expression link to add an expression as a Target.

        Select a Resource Type, enter a string expression and click Add to Targets. This will search for targets, using the defined criteria, dynamically at runtime. All Resources that belong to the selected Resource Type that contain the string expression are returned, within the context of the administrator privileges.
     f. Click **Add Targets**.
6. See Section 4.6, "Using the Condition Builder" for information on using the Condition Builder.
7. Click Save.

4.5.7.2 Modifying a Role Mapping Policy

To modify a Role Mapping Policy, proceed as follows:

1. Choose from the following methods to display the desired Role Mapping Policy.
   - Expand the information tree in the Navigation Panel to find Role Mapping Policies under the Role Catalog node of the appropriate Application and double click Role Mapping Policies. A search dialog opens in the Home area. For information about searching in the Home area, see Section 5.3, "Finding Objects with an Advanced Search."
   - In the Home area, select the Application Name under which the Application Role was created and click **Search** under **Role Mapping Policies**. A search dialog opens in the Home area. For information about searching in the Home area, see Section 5.3, "Finding Objects with an Advanced Search."

   When the correct Role Mapping Policy is displayed, select it and click Open to display the details in the Home area.

2. Modify the policy as necessary.
3. Click Apply.

4.5.7.3 Deleting a Role Mapping Policy

To delete a Role Mapping Policy, proceed as follows:

1. Choose from the following methods to display the desired Role Mapping Policy.
   - Expand the information tree in the Navigation Panel to find Role Mapping Policies under the Role Catalog node of the appropriate Application and double click Role Mapping Policies. A search dialog opens in the Home area. For information about searching in the Home area, see Section 5.3, "Finding Objects with an Advanced Search."
   - In the Home area, select the Application Name under which the Application Role was created and click **Search** under **Role Mapping Policies**. A search dialog opens in the Home area. For information about searching in the Home area, see Section 5.3, "Finding Objects with an Advanced Search."

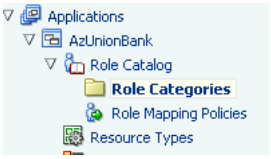
   When the correct Role Mapping Policy is displayed, select it and click Open to display the details in the Home area.
2. Double-click the Role Mapping Policy to delete.

   The Role Mapping Policy displays in the Home area.
3. Click Delete in the upper right corner of the Home area.

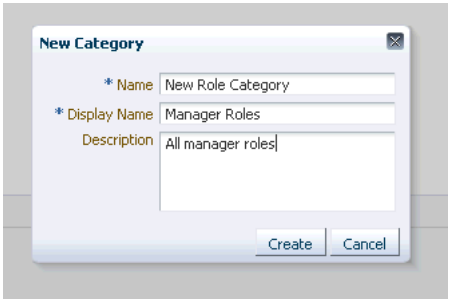
4.5.8 Managing a Role Category

A Role Category is a tag you can assign to a role for ease of management. You can create or delete a Role Category but you cannot modify them. To create a Role Category, proceed as follows. Instructions to delete a Role Category are detailed after the final step.

1. Expand the appropriate Application node in the Navigation Panel and double-click the **Roles Categories** node.



   The Role Categories page opens in the Home area.
2. Click **New** to display the **New Category** dialog.
3. Provide the following information.
   - **Name**
   - **Display Name**
   - **Description**



4. Click **Create**.

   The new category displays in the Role Categories list.

To delete a Role Category, expand the appropriate Application node in the Navigation Panel and double-click the **Roles Categories** node. Select the Role Category to delete and click Delete.

4.5.9 Managing Attributes and Functions as Extensions

Attributes and Functions are definitions organized under the Extensions node of the Application for which they were created. Attribute and function definitions can be used in a Condition or an Obligation. In regards to a Condition, attribute and function definitions can be used to make an optional expression that can be added to a policy to further restrict access to the protected resource. In regards to an Obligation, this optional set of name-value pairs returns additional information, with a policy decision, to the calling application. There are two ways to define an Obligation:

- Statically where an attribute with an absolute value is returned.
- Dynamically where an attribute value, or a custom function, is evaluated at runtime and the output is returned.

An Attribute can be a value dynamically defined at runtime (for example, the locality of the user) or a value based on the type of protected resource (for example, creation date of a text file). During policy evaluation, attribute values can be passed in by the application or Oracle Entitlements Server can retrieve it using a custom attribute retriever. Attributes must have a defined type. Boolean, integer, date, time and string are Oracle Entitlements Server predefined types. An attribute may be singular or a multi-valued list. A Function is a definition of externally implemented logic. It can be added to a policy as a condition on the policy's outcome. The following sections describe management operations on Attributes and Functions.

- Creating an Attribute
- Modifying an Attribute
- Deleting an Attribute
- Creating a Function

- Modifying a Function
- Deleting a Function

4.5.9.1 Creating an Attribute

To create an attribute, proceed as follows:

1. Navigate to, and expand, Extensions under the appropriate Application node in the Navigation Panel.
2. Right-click the **Attributes** node and select **New** from the menu.

   An Untitled page is displayed in the Home area.
3. Provide the following information for the attribute.
   - **Display Name** : The display name is optional and case insensitive. Specifying a meaningful display name is recommended since it is displayed in the Administration Console, and provides extra information to help administrators identify objects.
   - **Name :** The name is required and case insensitive. At runtime, this is the string the application passes to determine whether a user is authorized to access this Resource.
   - **Description** : Although optional, it is recommended to provide useful information about the entitlement. The description string is case insensitive.
   - **Category**: Select from Resource and Dynamic as a value for this required parameter.
   - **Type**: Select from String, Date, Integer, Boolean, Time.
   - **Input Values**: Select from Single and Multiple.
4. Select one of the following from the Save menu.
   - **Save and Close** saves the configuration and renames the page with the value provided for the Display Name.
   - **Save and Create Another** saves the configuration to the information tree in the Navigation Panel but leaves the Untitled area open for you to create another Attribute.

4.5.9.2 Modifying an Attribute

To modify an attribute, proceed as follows:

1. Choose from the following methods to display the desired Attribute.
   - Navigate to, and expand, Extensions under the appropriate Application node in the Navigation Panel. Double-click Attributes to open a search dialog in the Home area. For information about searching in the Home area, see Section 5.3.9, "Searching Attributes."
   - Search for Attributes using the Navigation Panel's search function and double-click the Attribute name in the Search Results tab. For information about searching in the Navigation Panel, see Section 5.2, "Finding Objects with a Simple Search."

   When the correct Attribute is displayed, select it and click Open to display the details in the Home area.
2. Modify the attribute as necessary.

3. Click **Apply**.

4.5.9.3 Deleting an Attribute

To delete an attribute, proceed as follows:

1. Choose from the following methods to display the Attribute.
   - Navigate to, and expand, Extensions under the appropriate Application node in the Navigation Panel. Right-click Attributes and select Open to display a search dialog in the Home area. Enter criteria for the lookup and click Search. For information about searching in the Home area, see Section 5.3, "Finding Objects with an Advanced Search."
   - Search for Attributes using the Navigation Panel's search function, right-click the Attribute name in the Search Results tab and select Open to display the Attribute in the Home area. For information about searching in the Navigation Panel, see Section 5.2, "Finding Objects with a Simple Search."
2. Select the Attribute and click **Delete**.

   A Delete Warning is displayed.
3. Click Yes.

4.5.9.4 Creating a Function

To create a function, proceed as follows:

1. Navigate to, and expand, Extensions under the appropriate Application node in the Navigation Panel.
2. Right-click the **Functions** node and select **New** from the menu.

   An Untitled page is displayed in the Home area.
3. Provide the following information for the function.
   - **Display Name** : The display name is optional and case insensitive. Specifying a meaningful display name is recommended since it is displayed in the Administration Console, and provides extra information to help administrators identify objects.
   - **Name :** The name is required and case insensitive. At runtime, this is the string the application passes to determine whether a user is authorized to access this Resource.
   - **Description** : Although optional, it is recommended to provide useful information about the entitlement. The description string is case insensitive.
   - **Function Class Name**: The name of the class that provides the functionality.
   - **Input Parameter**: A list of the types of parameters passed to the function.
   - **Return Type**: Select the data type returned by the function.
   - **Syntax Preview** displays a preview of the function's syntax.
4. Select one of the following from the Save menu.
   - **Save and Close** saves the configuration and renames the page with the value provided for the Display Name.
   - **Save and Create Another** saves the configuration to the information tree in the Navigation Panel but leaves the Untitled area open for you to create another.

4.5.9.5 Modifying a Function

To modify a function, proceed as follows:

1. Navigate to, and expand, Extensions under the appropriate Application node in the Navigation Panel.
2. Double-click Functions to open a search dialog in the Home area.
3. Enter search criteria to display the Function.

   For information about searching in the Home area, see Section 5.3.10, "Searching Functions."
4. Select the Function from the Search Results and click Open.

   The Function's details are displayed in the Home area.
5. Modify the Function as necessary.
6. Click **Apply**.

4.5.9.6 Deleting a Function

To delete a Function, proceed as follows:

1. Choose from the following methods to display the Function.
   - Navigate to, and expand, Extensions under the appropriate Application node in the Navigation Panel. Right-click Functions and select Open to display a search dialog in the Home area. Enter criteria for the lookup and click Search. Select the appropriate Function from the Search Results. For information about searching in the Home area, see Section 5.3, "Finding Objects with an Advanced Search."
   - Search for Functions using the Navigation Panel's search function, right-click the Function name in the Search Results tab and select Open to display the Function in the Home area. For information about searching in the Navigation Panel, see Section 5.2, "Finding Objects with a Simple Search."
2. Click Delete.

   A Delete Warning is displayed.
3. Click Yes.

4.6 Using the Condition Builder

An optional Condition in a policy rule can be used to further evaluate the applicability of an authorization decision returned in response to a request for access. For example, a Condition can be used to grant access to a resource only on the condition that the request was issued from a specific location or at a specific time.

> **Note:** Conditions in Role Mapping Policies provide the same functionality, and take the same format, as conditions in Authorization Policies.

A Condition is written in the form of an expression that resolves to either true or false. If the expression resolves to true, the condition is satisfied and the policy is applicable. If the expression does not resolve to true, the policy is not applicable. The expression can operate on attributes, functions or literals. Oracle Entitlements Server contains

predefined attributes and functions that can be inserted or you can create custom ones. The literals belong to the supported data types and are constants.

> **Note:** All Attributes and Functions (both custom and predefined) are created, collected and further managed under the Extensions node of the Application. For more information, see Section 4.5.9, "Managing Attributes and Functions as Extensions."

The Condition Builder allows an administrator to quickly create Condition expressions that can then be added to an Authorization Policy or a Role Mapping Policy. The following procedure illustrates how to use the Condition Builder to create a Condition for your policy. To create a Condition, you either create or modify an Authorization Policy or a Role Mapping Policy. Following one of these procedures will bring you to a step in which you can build a Condition.

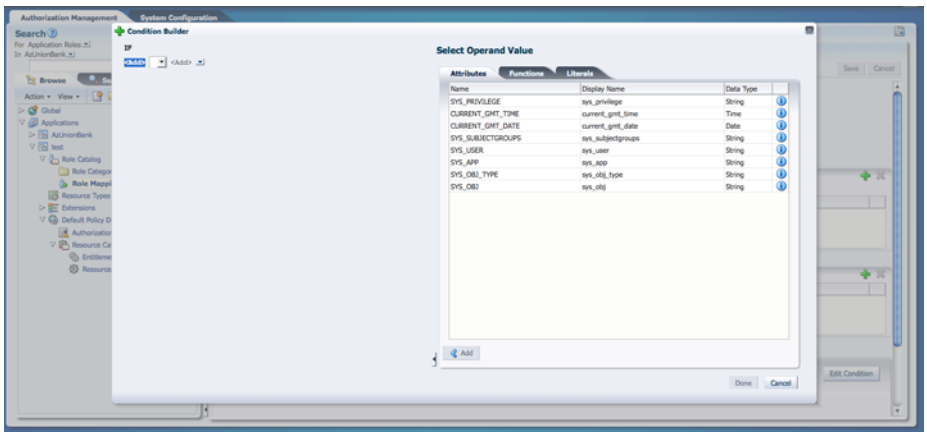
- Creating an Authorization Policy or
- Modifying an Authorization Policy
- Creating a Role Mapping Policy
- Modifying a Role Mapping Policy

When you get to the appropriate screen, follow this procedure.

1. Click the Condition tab.
2. Click Edit Condition.

   The Condition Builder (as displayed in Figure 4–1) displays. Note the frame of the Condition expression on the left. The frame contains two **Add** replaceables and an operator drop down. (The drop down is empty until an operand has been added.) The tabs for exprssion components - Attributes, Functions and Literals - are on the right. You will add components from these tabs to the Expression frame to build your Condition.

Figure 4–1 The Condition Builder

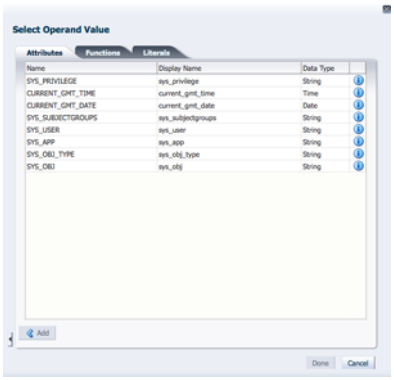


3. Click the tab that contains the component type you want to add to the Condition.

   Figure 4–2 is a screen shot of the Operand Value tabs. The Attributes and Function listed in these tabs are filtered based on the Application in which the policy is being created. For example, a custom Function created within Application 1 will

not be visible when the Condition Builder is activated to create a policy within Application 2.

Figure 4–2 Operand Value Tabs



4. Select the line that contains the component you want to add to the Condition and click Add.

   Click the blue **i** to display a Details box with more information regarding the component. Figure 4–3 is a screenshot after having added a SYS_APP attribute which takes a string value.

Figure 4–3 Adding a Literal to the Condition

5. Populate the value on the right of the expression by selecting the appropriate Operand Value and click Add.
6. Specify the operator on the right of the Condition Builder by clicking the drop down and selecting your choice.

   The operator options are dependent on the Operand Value.
7. Add additional expressions by clicking the last arrow in the expression and selecting AND, OR or NOT from the crop down menu, if applicable.

   REMOVE will clear the expression of all components so you may begin again.
8. Select components for the additional expression from the appropriate Operand Value tabs, if applicable.

   You may add as many expressions (and components) as necessary by clicking the last arrow in the current expression and selecting from the Operand Value tabs.
9. Click Done to complete the Condition.

The following points should be taken into account as you navigate the Condition Builder to create your expression.

- The Condition Builder contains Tool Tips on most fields for additional details.

- Click the appropriate blue **i** for information on the Operand Value.
- At the minimum, an expression must contain two operands and an operator.
- You can compare an Attribute and an Attribute, an Attribute and a Function, an Attribute and a Literal, a Function and a Function, and a Function and a Literal.
- The input parameters for Functions can be Attributes, Literals or Functions.
- The choice of operators displayed is directly related to the first operand chosen. For example, you cannot do less than or equal to on a string.
- The choice of a second Operand Values displayed within an expression is also directly related to the first operand chosen.
- REMOVE clears the expression to which it is tied of all components so you may begin again. It does not clear the entire Condition.
- The completed Condition (expression) is evaluated by Oracle Entitlements Server at runtime. The interpretation is governed by the rules of precedence.
- The outcome of this Condition must be a boolean.

The following sections contain procedures for more complex conditions.

- Building a Complex Expression
- Passing Parameters to Functions

4.6.1 Building a Complex Expression

This procedure explains how you might build a complex expression using parenthesis.

1. Follow one of these procedures to bring you to the Condition Builder.
   - Creating an Authorization Policy or
   - Modifying an Authorization Policy
   - Creating a Role Mapping Policy
   - Modifying a Role Mapping Policy
2. Click the Condition tab.
3. Click Edit Condition.

   The Condition Builder displays as in Figure 4–1.
4. Click the Attributes tab.
5. Select the `DateAttr` custom attribute and click Add.

   `DateAttr` is not a predefined Oracle Entitlements Server attribute so this step assumes a custom attribute has been defined as documented in Section 4.5.9, "Managing Attributes and Functions as Extensions." DateAttr is added to the left of the operator.
6. Select the equal sign (=) as the operator.
7. Select the `CURRENT_GMT_DATE` predefined attribute and click Add.

   `CURRENT_GMT_DATE` is a predefined Oracle Entitlements Server attribute and can be viewed under the Attributes tab. It is added to the right of the operator.
8. Add more complexity to the Condition by selecting the appropriate AND, OR or NOT operation at the end of the line of code.

Parentheses must match; there must be an equal number of open and closing parentheses. If you select an operation at the end of a line of code, the operation will involve the code itself. If you select an operation at the end of the entire Condition, it will allow you to add on to the Condition as a whole.

9. Add additional conditions by choosing values from Attributes, Functions or Literals as necessary.
10. Click Done when finished.

4.6.2 Passing Parameters to Functions

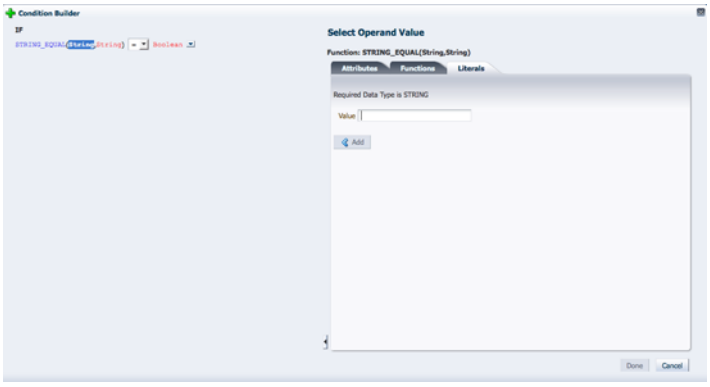
This procedure describes how to pass parameters into a Function.

1. Follow one of these procedures to bring you to the Condition Builder.
   - Creating an Authorization Policy or
   - Modifying an Authorization Policy
   - Creating a Role Mapping Policy
   - Modifying a Role Mapping Policy
2. Click the Condition tab.
3. Click Edit Condition.

   The Condition Builder displays as in Figure 4–1.
4. Click the Functions tab.
5. Select STRING_EQUAL and click Add.

   Figure 4–4 illustrates an added Function and contains placeholders for the two parameters that must be passed to it. This Function will compare the two strings (one the value of a predefined attribute).

Figure 4–4 Adding a Function



6. Select the first parameter if not already.
7. Click the Attributes tab.
8. Select SYS_USER and click Add.

   The second parameter is highlighted and the Literal tab is activated.
9. Enter a value for the second parameter and click Add.

For this example, joe. The boolean to the right of the operator is highlighted and the Literal tab is activated.

10. Choose the appropriate operator.
11. Click the Boolean replaceable and select whether this function output should be true or false.
12. Add Additional operands as you see fit.
13. Click Done when finished.

5

Querying Security Objects

Oracle Entitlements Server enables querying for policies and policy objects from within the Oracle Entitlements Server Administration Console. This chapter explains the types of search functionalities and for what purposes they can be used. It contains the following topics:

- Section 5.1, "Searching with the Administration Console"
- Section 5.2, "Finding Objects with a Simple Search"
- Section 5.3, "Finding Objects with an Advanced Search"

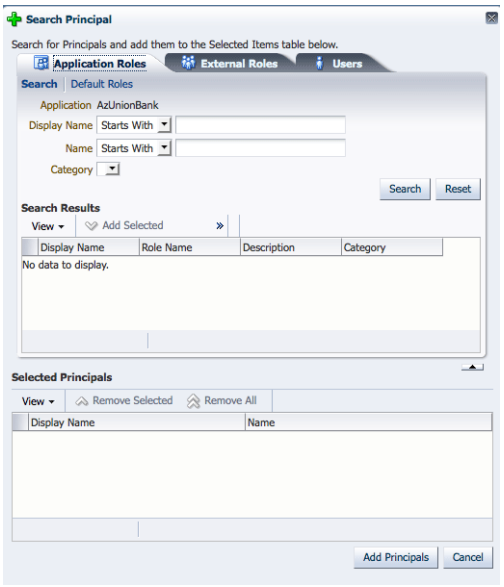
5.1 Searching with the Administration Console

Oracle Entitlements Server enables different kinds of search queries using the Administration Console.

- A *simple search* matches names and display names only. The search is generated from the top of the Navigation Panel and results are displayed in the Navigation Panel. For more information, see Section 5.2, "Finding Objects with a Simple Search."
- An *advanced search* uses operators that enable more sophisticated matching. The advanced search screen is launched by double-clicking an object in the Navigation Panel, or from the Home area. The search box opens in the Home area and results are also displayed there. For more information, see Section 5.3, "Finding Objects with an Advanced Search."
- A *pop-up search* opens from within the Authorization Policy or Role Mapping Policy screens, when the policy is being created or modified, by clicking the green Add button (plus sign). The pop-up search box uses a shopping cart paradigm. You add choices selected from the multiple, displayed tabs on the top of the search box to the Selected box on the bottom of the search box. All choices in the Selected box are added when you click Add.

  Figure 5–1 is a screen shot of the pop-up search box for adding a Principal. You can click between the three tabs (Application Roles, External Roles, and Users), selecting one or more policy subjects and adding them to the Selected Principals box. When you click Add Principals, all choices added from all tabs will then be added to the policy.

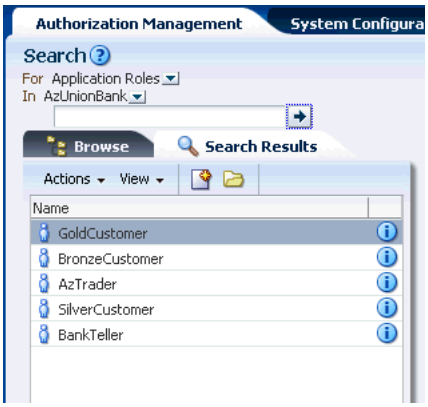
Figure 5–1 Pop-up Search Box



5.2 Finding Objects with a Simple Search

A simple query matches names and display names only. The fields in the top portion of the Authorization Management tab in the Navigation Panel, as shown in Figure 5–2, are used to specify simple queries.

Figure 5–2 Simple Search Fields in Navigation Panel



To specify a simple search, proceed as follows:

1. Select the policy object for which you are searching from the **For** list.

   The following object types are available:

   - Application Roles
   - External Roles
   - Users
   - Resources
   - Resource Types

- Entitlements
- Attributes

2. Select the search scope from the **In** list.

   The *search scope* defines the level at which the search will take place. When searching for Application Roles, Resources, Resource Types, Entitlements and Attributes, the search scope is an Application. For External Roles and Users, the search scope is Global. For Entitlements and Resources, the search scope is the Policy Domain within an Application.

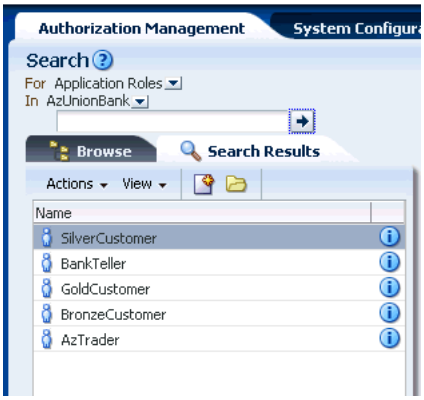
   Note: If performing a Resource search, you also select the Resource Type from the **Type** list.

3. Optionally, enter a string to match in the text box.

   Wildcard characters percent (%) and asterisk (*) are supported for a simple search.

4. Click the arrow icon next to the text box to begin the search.

   Names and display names matching the specified criteria are returned and displayed in the **Search Results** tab. If no search string was entered, a list of all objects of the specified type is returned.



5. Double-click the object to edit, right click the object and select New to create, or click the object's information icon for details.

   For more information on managing policy objects, see Chapter 4, "Managing Policies and Roles."

5.3 Finding Objects with an Advanced Search

An advanced search is generally initiated by double-clicking the object name in the Navigation Panel, or from the Search link for the object in the Home area. An advanced search can use the following operators:

- Starts with
- Ends with
- Contains
- Equal to

There is no support for wildcard characters in an advanced search. In particular, the asterisk (*) or percent (%) characters are treated as plain text in any advanced search

parameter. The following sections have information on searching for policy objects with an advanced search.

- Searching External Roles
- Searching Applications
- Searching Resource Types
- Searching Application Roles
- Searching Role Mapping Policies
- Searching Resources
- Searching Entitlements
- Searching Authorization Policies
- Searching Attributes
- Searching Functions

5.3.1 Searching External Roles

To search External Roles, proceed as follows:

1. Select from the following methods to display the Search External Roles page:
   - In the Navigation Panel, expand Global and double-click **External Roles**. (Alternately, right-click **External Roles** and select Open.)
   - In the Home area, click **Search - External Roles** from the Search and Create section.
2. Enter the following query parameters:
   - **Name**: Select an operator from the list and enter a string to match.
   - **Display Name**: Select an operator from the list and enter a string to match.

   Optionally, select from the **Saved Search** drop-down list of previously saved searches. Its query parameters automatically populate the search fields. Select **Personalize...** to set options for previously saved searches.
3. Optionally, click **Save...** to name the current query parameters.

   The named search is added to the **Saved Search** list.
4. Click **Search**.

   The results are displayed in Search Results.

5.3.2 Searching Applications

To search applications, proceed as follows:

1. Select from the following methods to display the Search Applications page:
   - In the Navigation Panel, double-click Applications to display the **Search Applications** page. (Alternately, right-click **Applications** and select Open.)
   - In the Home area, click **Search - Applications** from the Search and Create section.
2. Enter the following query parameters:
   - **Name**: Select an operator from the list and enter a string to match.

- **Display Name**: Select an operator from the list and enter a string to match.

Optionally, select from the **Saved Search** drop-down list of previously saved searches. Its query parameters automatically populate the search fields. Select **Personalize...** to set options for previously saved searches.

3. Optionally, click **Save...** to name the current query parameters.

   The named search is added to the **Saved Search** list.

4. Click **Search**.

   The results are displayed in Search Results.

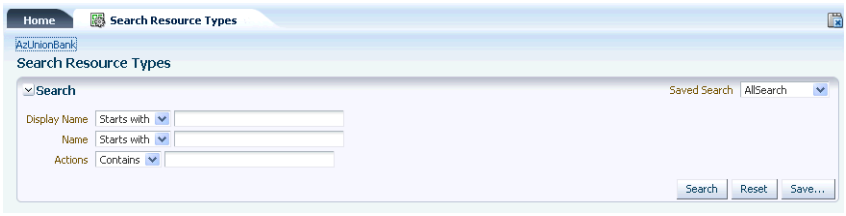
5.3.3 Searching Resource Types

To search Resource Types, proceed as follows:

1. Select from the following methods to display the Search Resource Types page as in Figure 5–3.

   - In the Navigation Panel, expand the Application node and double-click **Resource Types**. (Alternately, right-click **Resource Types** and select Open.)
   - In the Home area, select the appropriate Application Name and click **Search** under **Resource Types**.

Figure 5–3 Searching for Resource Types



2. Enter the following query parameters:

   - **Display Name:** Select an operator from the list and enter a string to match.
   - **Name**: Select an operator from the list and enter a string to match.
   - **Actions**: Select an operator from the list and enter a string to match.

   Optionally, select from the **Saved Search** drop-down list of previously saved searches. Its query parameters automatically populate the search fields. Select **Personalize...** to set options for previously saved searches.

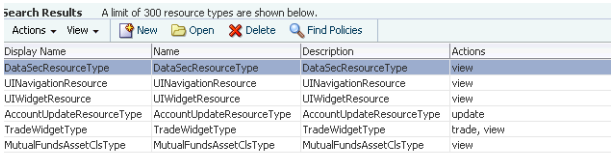
3. Optionally, click **Save...** to name the current query parameters.

   The named search is added to the **Saved Search** list.

4. Click **Search**.

   All results matching the query specifications are displayed in the **Search Results** table as illustrated in Figure 5–4.

Figure 5–4 Resource Type Search Results



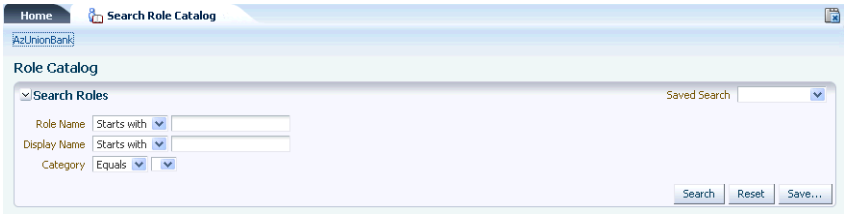
5.3.4 Searching Application Roles

To search Application Roles, proceed as follows:

1. Select from the following methods to display the Search Role Catalog page.
   - In the Navigation Panel, expand **Applications** and the named Application node applicable to the search, and double-click **Role Catalog**. (Alternately, right-click **Role Catalog** and select Open.)
   - In the Home area, select the Application Name and click **Search** from **Application Roles**.

   The Search Role Catalog tab is displayed as in Figure 5–5.

Figure 5–5 Searching for Application Roles in a Role Catalog



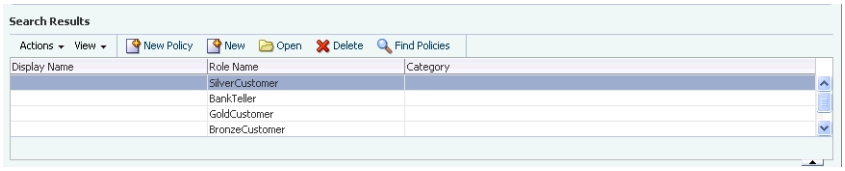
2. Enter the following query parameters:
   - **Role Name:** Select an operator from the list and enter a string to match.
   - **Display Name:** Select an operator from the list and enter a string to match.
   - **Category**: Select a Role Category from the list. (Oracle Entitlements Server only supports an *equals* search for Role Category.)

   Optionally, select from the **Saved Search** drop-down list of previously saved searches. Its query parameters automatically populate the search fields. Select **Personalize...** to set options for previously saved searches.
3. Optionally, click **Save...** to name the current query parameters.

   The named search is added to the **Saved Search** list.
4. Click **Search**.

   All results matching the query specifications are displayed in the **Search Results** table as in Figure 5–6.

Figure 5–6 Application Role Search Results



5.3.5 Searching Role Mapping Policies

1. Select from the following methods to display the Search Role Mapping Policies page:
   - In the Navigation Panel, expand **Applications** and the named Application node applicable to the search, and double-click **Role Mapping Policies**. (Alternately, right-click **Role Mapping Policies** and select Open.)
   - In the Home area, select the Application Name and click **Search** from **Role Mapping Policies**.

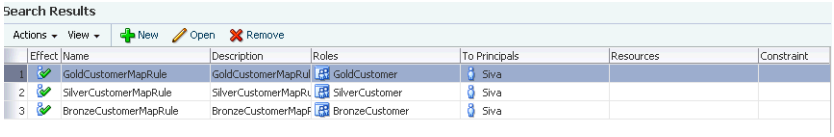
     The Search Role Policies page is displayed as in Figure 5–7.

Figure 5–7 Searching for Role Mapping Policies

2. In the **Search** section, enter the query parameters as follows:
   - **Effect**: Select the policy effect (Grant/Deny) from the list.
   - **Display Name:** Select an operator from the list and enter a string to match.
   - **Name**: Select an operator from the list and enter a string to match.
   - **Role**: Select an operator from the list and enter a string to match.
   - **Principal**: Select an operator from the list and enter a string to match.
   - **Target**: Select an operator from the list and enter a string to match.
3. Click **Search**.

   All results matching the query specifications are displayed in the **Search Results** table as in Figure 5–8.

Figure 5–8 Role Mapping Policy Search Results



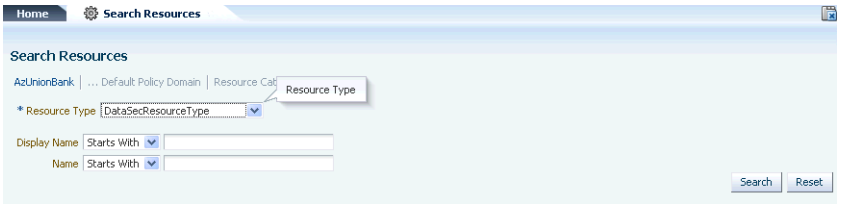
5.3.6 Searching Resources

A Resource can be hierarchical (a scenario in which the sub resource inherits attributes from the parent resource) or non-hierarchical. If a Resource is hierachical, its tiered-organization is shown in the Search results. To search Resources, proceed as follows:

1. Select from the following methods to display the Search Role Mapping Policies page:
   - In the Navigation Panel, expand **Applications** and the named Application node applicable to the search. Expand the appropriate Policy Domain and Resource Catalog and double-click **Resources**. (Alternately, right-click **Resources** and select Open.)
   - In the Home area, select the Application Name and click **Search** from **Resources**.

     The Search Resources page is displayed as in Figure 5–9.

Figure 5–9 Searching for Resources



2. Enter the following query parameters:
   - **Resource Type**: Select a resource type from the list. This parameter is required.
   - **Display Name:** Select an operator from the list and enter a string to match.
   - **Name**: Select an operator from the list and enter a string to match.
3. Click **Search**.

   All results matching the query specifications are displayed in the **Search Results** table.

5.3.7 Searching Entitlements

To search Entitlements, proceed as follows:

1. Select from the following methods to display the Search Entitlements page:
   - In the Navigation Panel, expand **Applications** and the named Application node applicable to the search. Expand the appropriate Policy Domain and Resource Catalog and double-click **Entitlements**. (Alternately, right-click **Entitlements** and select Open.)
   - In the Home area, select the Application Name and click **Search** from **Entitlements**. (In this case, the search is done only within the Default Policy Domain.)

   The Search Entitlements tab is displayed in the Home area as in Figure 5–10.

Figure 5–10 Searching for Entitlements

2. Enter the following query parameters:

   - **Entitlement Name:** Select an operator from the list and enter a string to match.
   - **Display Name:** Select an operator from the list and enter a string to match.
   - **Resource name**: Select an operator from the list and enter a string to match.

   Optionally, select from the **Saved Search** drop-down list of previously saved searches. Its query parameters automatically populate the search fields. Select **Personalize...** to set options for previously saved searches.

3. Optionally, click **Save...** to name the current query parameters.

   The name search is added to the **Saved Search** list.

4. Click **Search**.

   All results matching the query specifications are displayed in the **Search Results** table.

5.3.8 Searching Authorization Policies

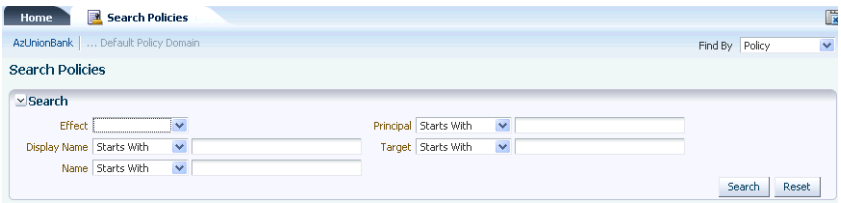
Authorization Policies can be searched by specifying a policy name, a principal, or a target. To search Authorization Policies, proceed as follows:

1. Select from the following methods to display the Search Policies page:

   - In the Navigation Panel, expand **Applications** and the named Application node applicable to the search. Expand the appropriate Policy Domain and Resource Catalog and double-click **Authorization Policies**. (Alternately, right-click **Authorization Policies** and select Open.)
   - In the Home area, select the Application Name, and click **Search** under Authorization Policies. (In this case, the search is done within the Default Policy Domain.)

   The Search Policies tab is displayed in Figure 5–11.

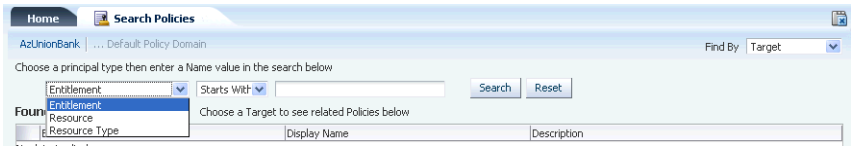
Figure 5–11 Searching Policies



2. Select the search type from the **Find By** list.

The query parameters change according to the selection. Options include **Policy**, **Principal** or **Target**. Figure 5–11 is a screenshot in which Policy is selected. Figure 5–12 is a screenshot in which Target is selected.

Figure 5–12 Searching Polices by Target



3. Search using the option based on your previous selection.
   - To Find By: Policy, enter the following query parameters.
     - **Effect:** Select the policy effect (Grant/Deny) from the list.
     - **Display Name:** Select an operator from the list and enter a string to match.
     - **Name:** Select an operator from the list and enter a string to match.
     - **Principal:** Select an operator from the list and enter a string to match.
     - **Target:** Select an operator from the list and enter a string to match.
   - To Find By: Principal or Find By: Target, select an operator from the list, and enter a string to match.

     A Resource Type must be provided if the Resource or Resource Type operator is selected.
4. Click **Search**.

5.3.9 Searching Attributes

To search Attributes, proceed as follows:

1. In the Navigation Panel, expand **Applications** and the named Application node applicable to the search.
2. Expand **Extensions** and double-click **Attributes**.

   Alternately, right-click **Attributes** and select Open. The Search Attributes page is displayed.
3. Enter the following query parameters.
   - **Display Name**: Select an operator from the list and enter a string to match.
   - **Name**: Select an operator from the list and enter a string to match.
   - **Type**: Select an operator from the list and enter a string to match.

   Optionally, select from the **Saved Search** drop-down list of previously saved searches. Its query parameters automatically populate the search fields. Select **Personalize...** to set options for previously saved searches.
4. Optionally, click **Save...** to name the current query parameters.

   The named search is added to the **Saved Search** list.
5. Click **Search**.

5.3.10 Searching Functions

To search application functions, proceed as follows

1. In the Navigation Panel, expand **Applications** and the named Application node applicable to the search.
2. Expand **Extensions** and double-click **Functions**.

   Alternately, right-click **Functions** and select Open. The Search Functions page is displayed.
3. Enter the following query parameters.
   - **Name**: Select an operator from the list and enter a string to match.
   - **Display Name**: Select an operator from the list and enter a string to match.

   Optionally, select from the **Saved Search** drop-down list of previously saved searches. Its query parameters automatically populate the search fields. Select **Personalize...** to set options for previously saved searches.
4. Optionally, click **Save...** to name the current query parameters.

   The named search is added to the **Saved Search** list.
5. Click **Search**.

6

Configuring Predefined Attribute Retrievers

As discussed in Section 1.3, "Overview of the Oracle Entitlements Server Architecture," the Policy Information Point (PIP) is a system entity that acts as a source for attribute values. Oracle Entitlements Server relies on an Attribute Retriever plug-in to get attribute values from one or more of these PIP information stores. Predefined Attribute Retrievers are shipped with Oracle Entitlements Server. This chapter documents these predefined Attribute Retrievers and related configuration requirements. It contains the following sections.

- Section 6.1, "Understanding Predefined Attribute Retrievers"
- Section 6.2, "Configuring the Predefined Attribute Retrievers"
- Section 6.3, "Modifying jps-config.xml"
- Section 6.4, "Setting Up PIP Connection Credentials"

> **Note:** See the *Oracle Fusion Middleware Developer's Guide for Oracle Entitlements Server* for information on custom Attribute Retrievers.

6.1 Understanding Predefined Attribute Retrievers

Oracle Entitlements Server contains predefined Attribute Retrievers that are used to connect to, and retrieve attribute values from, Lightweight Directory Access Protocol (LDAP) data stores and relational database management systems (RDBMS). These plug-ins can handle one or more attributes defined in the system without additional programming. They also contain a caching feature and failover.

- An in-memory cache mechanism is used to improve performance by reducing communications between Oracle Entitlements Server and the external repository. The cache holds up to 1000 entries and can be enabled for each individual attribute. The cache size is not configurable. If the limit is reached, cache items are removed randomly. Example 6–2 illustrates the definition of an individual attribute with the `cached` and `ttl` properties.
- Repository failover can also be configured. When a call for an attribute is received, Oracle Entitlements Server checks whether the primary repository is active. If it is active, the value is retrieved. If the primary repository is not active, it has failed previously and the backup repository is active. In the latter case, Oracle Entitlements Server checks to see if it is time to switch back to the active repository (based on configuration). If it is time to switch back, the switch is made and the value is retrieved from the primary repository. If the configured time has not yet passed, the value is retrieved from the active backup repository.

> **Note:** If errors occur when retrieving values from the primary repository, Oracle Entitlements Server searches the backup repositories, trying them one by one until an active one is found.

See Section 6.2.3, "Configuring Individual Attributes for Predefined Attribute Retrievers" for configuration information.

6.2 Configuring the Predefined Attribute Retrievers

Configuration information for these Attribute Retrievers is defined in the `jps-config.xml` configuration file. You must configure two types of information: attribute query information and repository connection information

- Repository connection information is used to connect to the data store and may include its location, JDBC driver and URL or LDAP URL (whichever is applicable) and the user/credential information. This connection information is related to a particular retriever instance. Repository connection information is defined in the `<serviceInstances>` section of `jps-config.xml` as illustrated in Example 6–1.

Example 6–1 Repository Connection Information Defined for Attribute Retriever

```
<serviceInstance name="policystore.rdbms" provider="policy.rdbms">
 <property name="jdbc.url"
  value="jdbc:oracle:thin:@scl58116.us.oracle.com:1521:orcl"/>
 <property name="jdbc.driver" value="oracle.jdbc.driver.OracleDriver"/>
 <property name="bootstrap.security.principal.key" value="keyname"/>
 <property name="bootstrap.security.principal.map" value="mapname"/>
 <property name="oracle.security.jps.ldap.root.name" value="cn=jpsTestNode"/>
 <property name="oracle.security.jps.farm.name"
  value="cn=wcai_view_jing.atzsrg"/>
</serviceInstance>
```

Section 6.2.1, "Configuring the LDAP Respository Attribute Retriever Parameters," Section 6.2.2, "Configuring the Database Repository Attribute Retriever Parameters," and Section 6.3, "Modifying jps-config.xml" contain information regarding a respository connection configuration.

> **Note:** The instance must also be defined in the default `<jpsContexts>` section. See Example 6–8, "Declaring the Predefined Attribute Retriever in jpsContext".

- Attribute query information is related to a particular attribute and includes its name, the name of the predefined Attribute Retriever used, the search query for retrieval (for example, a SQL query if the store is a relational database or an LDAP query if it's a directory), and any attribute caching information. Attribute query information is defined in the `<propertySets>` section of `jps-config.xml` as illustrated in Example 6–2.

Example 6–2 Attribute Query Information Defined for Attribute Retriever

```
<propertySet name="ootb.pip.attribute.age.based.on.myattr.ldap">
    <property name="ootb.pip.attr.type" value="OOTB_PIP_ATTRIBUTE"/>
    <property name="ootb.pip.ref" value="pip.service.ootb.ldap"/>
    <property name="name" value="oespipage_myattr"/>
    <property name="query" value="(cn=%MyAttr%)"/>
```

```
    <property name="cached" value="true"/>
    <property name="ttl" value="60"/>
</propertySet>
```

Section 6.2.3, "Configuring Individual Attributes for Predefined Attribute Retrievers" and Section 6.3, "Modifying jps-config.xml" contain information regarding an attribute query configuration.

Note: These predefined Attribute Retrievers can be configured with Oracle Database 11gR1, Oracle Internet Directory 11gR1, and Oracle Virtual Directory 11gR1.

The following sections contain information on the configuration parameters for each type of Attribute Retriever. As previously mentioned, these parameters are in the `jps-config.xml`, the configuration file (used by Java EE containers) located in the *$DOMAIN_HOME*`/config/fmwconfig` directory.

- Configuring the LDAP Respository Attribute Retriever Parameters
- Configuring the Database Repository Attribute Retriever Parameters
- Configuring Individual Attributes for Predefined Attribute Retrievers

6.2.1 Configuring the LDAP Respository Attribute Retriever Parameters

Table 6–1 documents the parameters that must be defined when using the LDAP Attribute Retriever. See Example 6–5, "Using the Predefined LDAP Attribute Retriever" and Example 6–10, "Configuring LDAP Failover" for sample configuration code.

Table 6–1 LDAP Attribute Retriever Parameters

Name	Usage
name	**Description:** The predefined Attribute Retriever's name **Mandatory** **Accepted Value:** String defining the Attribute Retriever service instance.
description	**Description:** A description of the predefined Attribute Retriever **Optional** **Accepted Value:** string
type	**Description:** The predefined Attribute Retriever's type **Manadatory** **Accepted Value:** LDAP_PIP
failed.server.retry.interval	**Description:** After communication with a primary repository has failed, this attribute defines the interval of time during which the backup repository is used before switching back to the primary repository. **Optional** **Accepted Value:** Takes a value equal to the number of seconds. Default value is 15.

Table 6–1 (Cont.) LDAP Attribute Retriever Parameters

Name	Usage
bootstrap.security.principal.key	**Description:** Defines the key for the password credentials to access the LDAP policy store, stored in the CSF store. Valid in JEE and JSE applications. Applies to LDAP and database stores. See Section 6.4, "Setting Up PIP Connection Credentials." **Optional:** For production mode only. **Accepted Value:** key name of the credential; for example, `oes_sm_key`.
bootstrap.security.principal.map	**Description:** Defines the map for the password credentials to access the LDAP policy store, stored in the CSF store. Valid in JEE and JSE applications. Applies to LDAP and database stores. See Section 6.4, "Setting Up PIP Connection Credentials." **Optional:** For production mode only. **Accepted Value:** map name of the credential; for example, `oes_sm_map`.
ldap.url	**Description:** Defines the URL of the LDAP policy store. Valid in JEE and JSE applications and only applies to LDAP stores. **Mandatory** **Accepted Value:** URI of the LDAP policy store in the format `ldap://host:port`.

6.2.2 Configuring the Database Repository Attribute Retriever Parameters

Table 6–2 documents the parameters that must be defined when using the RDBMS Attribute Retriever. See Example 6–6, "Using the Predefined RDBMS Attribute Retriever with JDBC" and Example 6–7, "Using the Predefined RDBMS Attribute Retriever with SQL" for sample configuration code.

Table 6–2 RDBMS Attribute Retriever Parameters

Name	Usage
name	**Description:** The predefined Attribute Retriever's name **Mandatory** **Accepted Value:** String defining the Attribute Retriever service instance.
description	**Description:** A description of the predefined Attribute Retriever **Optional** **Accepted Value:** string
type	**Description:** The predefined Attribute Retriever's type **Manadatory** **Accepted Value:** RDBMS_PIP

Table 6–2 (Cont.) RDBMS Attribute Retriever Parameters

Name	Usage
failed.server.retry.interval	**Description:** After the primary repository has failed, this attribute identifies the interval of time during which the backup repository is used before switching back to the primary repository. **Optional** **Accepted Value:** Takes a value equal to the number of seconds. Default value is 15.
bootstrap.security.principal.key	**Description:** Defines the key for the password credentials to access the database, stored in the CSF store. Valid in JEE and JSE applications. See Section 6.4, "Setting Up PIP Connection Credentials." **Optional:** For production mode only. **Accepted Value:** key name of the credential; for example, `oes_sm_key`.
bootstrap.security.principal.map	**Description:** Defines the map for the password credentials to access the database, stored in the CSF store. Valid in JEE and JSE applications. See Section 6.4, "Setting Up PIP Connection Credentials." **Optional:** For production mode only. **Accepted Value:** map name of the credential; for example, `oes_sm_map`.
jdbc.driver	**Description:** Location of the driver when using Java Database Connectivity (JDBC) API to connect to a database. **Mandatory:** When using JDBC API to connect to database. **Accepted Value:** `oracle.jdbc.driver.OracleDriver`, for example
jdbc.url	**Description:** Takes a URL that points to the database. **Mandatory:** When using JDBC API to connect to database. **Accepted Value:** A list of comma-delimited URLs. The first is treated as primary and so on. For example, `jdbc:oracle:thin:@scl58116.us.oracle.com:1521:orcl`
datasource.jndi.name	**Description:** Data source JNDI name if you want the PIP instance working through data source rather than directly through JDBC. The data source scenario is supported on WebLogic Server and WebSphere Application Server only. **Mandatory:** If you want the PIP instance working through data source rather than directly through JDBC. **Accepted Value:** JNDI name of pre-defined data source object

6.2.3 Configuring Individual Attributes for Predefined Attribute Retrievers

Table 6–3 documents the parameters to be defined for each attribute retrieved by the configured Attribute Retriever. See Example 6–9, "Enabling an Attribute's Cache" for a sample configuration.

Table 6–3 Configure Attributes to be Retrieved

Name	Usage
name	**Description:** The name of the attribute as defined in the policy store. When using the LDAP predefined Attribute Retriever, the attribute name defined for Oracle Entitlements Server must be the same as the attribute name defined in the LDAP store. Currently, there is no name mapping functionality. **Mandatory** **Accepted Value:** Attribute name
query	**Description:** The SQL command or LDAP filter used for the query. Users can use a built-in and custom attributes in the query string. For example, the built-in attribute `sys_user` can be used to define a query such as *select age from customers where name=%sys_user%;*. The token is automatically replaced by its value before sending the query to the data store. Bi-directional dependency (where, for example, AttributeA's query string contains AttributeB and AttributeB's query string contains AttributeA) can also be detected and, in such cases, an exception is thrown. **Mandatory** **Accepted Value:** SQL command or LDAP filter.
search.base	**Description:** The LDAP search base. **Mandatory:** For LDAP only. **Accepted Value:** The DN of the search base object.
ttl	**Description:** The time-to-live in seconds of any cached attribute values when cached is enabled. **Optional** **Accepted Value:** Any integer; default value is 60 seconds if cache is enabled.
cached	**Description:** Enables the caching of attribute values. **Optional** **Accepted Value:** Default value is false.
ootb.pip.attr.type	**Description:** Should be set to OOTB_PIP_ATTRIBUTE. **Mandatory** **Accepted Value:** OOTB_PIP_ATTRIBUTE.
ootb.pip.ref	**Description:** Should be set to an OOTB PIP instance. **Mandatory** **Accepted Value:** The PIP service instance name defined in the `<serviceInstance>` section of `jps-config.xml`

6.3 Modifying jps-config.xml

To configure the predefined Attribute Retriever in `jps-config.xml`, modify the elements as described in each example in this section. Example 6–3 is a sample `jps-config.xml` file. The examples following it illustrate the modifications that can be made.

Example 6–3 Sample jps-config.xml File

```
<?xml version="1.0"?>

<jpsConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="
```

```
http://xmlns.oracle.com/oracleas/schema/jps-config-11_0.xsd">

   <property name="oracle.security.jps.jaas.mode" value="off"/>
   <property name="oracle.security.jps.enterprise.user.class"
    value="weblogic.security.principal.WLSUserImpl"/>
   <property name="oracle.security.jps.enterprise.role.class"
    value="weblogic.security.principal.WLSGroupImpl"/>

<propertySets>
<!-- These are the global authenticated role properties -->
  <propertySet name="authenticated.role.properties">
     <property name="authenticated.role.name" value="authenticated-role"/>
     <property name="authenticated.role.uniquename" value="authenticated-role"/>
     <property name="authenticated.role.description"
        value="This is the authenticated role used by identity store
        service instance."/>
  </propertySet>

<!-- attribute defined for ldap retriever -->
  <propertySet name="ootb.pip.attribute.age.ldap">
     <property name="ootb.pip.attr.type" value="OOTB_PIP_ATTRIBUTE"/>
     <property name="ootb.pip.ref" value="pip.service.ootb.ldap"/>
     <property name="name" value="oespipage"/>
     <property name="query" value="(cn=%SYS_USER%)"/>
     <property name="cached" value="true"/>
     <property name="ttl" value="60"/>
  </propertySet>

  <propertySet name="ootb.pip.attribute.age.based.on.myattr.ldap">
     <property name="ootb.pip.attr.type" value="OOTB_PIP_ATTRIBUTE"/>
     <property name="ootb.pip.ref" value="pip.service.ootb.ldap"/>
     <property name="name" value="oespipage_myattr"/>
     <property name="query" value="(cn=%MyAttr%)"/>
     <property name="cached" value="true"/>
     <property name="ttl" value="60"/>
  </propertySet>

  <propertySet name="ootb.pip.attribute.gender.ldap">
     <property name="ootb.pip.attr.type" value="OOTB_PIP_ATTRIBUTE"/>
     <property name="ootb.pip.ref" value="pip.service.ootb.ldap"/>
     <property name="name" value="oespipgender"/>
     <property name="query" value="(oespipage=%oespipage%)"/>
     <property name="cached" value="true"/>
     <property name="ttl" value="60"/>
  </propertySet>

<!-- attribute defined for rdbms retriever -->
  <propertySet name="ootb.pip.attribute.age.rdbms">
     <property name="ootb.pip.attr.type" value="OOTB_PIP_ATTRIBUTE"/>
     <property name="ootb.pip.ref" value="pip.service.ootb.db"/>
     <property name="name" value="oespipage"/>
     <property name="query" value="select oespipage
         from pip_info_store where username=%SYS_USER%"/>
     <property name="cached" value="true"/>
     <property name="ttl" value="60"/>
  </propertySet>

  <propertySet name="ootb.pip.attribute.age.based.on.myattr.rdbms">
     <property name="ootb.pip.attr.type" value="OOTB_PIP_ATTRIBUTE"/>
     <property name="ootb.pip.ref" value="pip.service.ootb.db"/>
```

```
      <property name="name" value="oespipage_myattr"/>
      <property name="query" value="select oespipage
          as oespipage_myattr from pip_info_store where username=%MyAttr%"/>
      <property name="cached" value="true"/>
      <property name="ttl" value="60"/>
    </propertySet>

    <propertySet name="ootb.pip.attribute.gender.rdbms">
      <property name="ootb.pip.attr.type" value="OOTB_PIP_ATTRIBUTE"/>
      <property name="ootb.pip.ref" value="pip.service.ootb.db"/>
      <property name="name" value="oespipgender"/>
      <property name="query" value="select oespipgender
           from pip_info_store where oespipage=%oespipage%"/>
       <property name="cached" value="true"/>
      <property name="ttl" value="60"/>
    </propertySet>
  </propertySets>

  <serviceProviders>

    <serviceProvider type="CREDENTIAL_STORE" name="credstoressp"
         class="oracle.security.jps.internal.credstore.ssp.
                SspCredentialStoreProvider">
      <description>SecretStore-based CSF Provider</description>
    </serviceProvider>

    <serviceProvider class="oracle.security.jps.az.
          internal.runtime.provider.PIPServiceProvider"
          name="pip.service.provider" type="PIP"/>

    <serviceProvider type="POLICY_STORE" name="policy.rdbms"
          class="oracle.security.jps.internal.policystore.
          OPSSPolicyStoreProvider">
      <property name="policystore.type" value="DB_ORACLE"/>
      <description>DBMS based PolicyStore</description>
    </serviceProvider>

    <serviceProvider name="pdp.service.provider" type="PDP"
           class="oracle.security.jps.az.internal.
           runtime.provider.PDPServiceProvider">
        <description>OPSS Runtime PDP Service Provider</description>
    </serviceProvider>

    <serviceProvider name="idstore.xml.provider" type="IDENTITY_STORE"
           class="oracle.security.jps.internal.idstore.
           xml.XmlIdentityStoreProvider">
         <description>XML-based IdStore Provider</description>
    </serviceProvider>

    <serviceProvider name="jaas.login.provider" type="LOGIN"
           class="oracle.security.jps.internal.
           login.jaas.JaasLoginServiceProvider">
         <description>This is Jaas Login Service Provider and is used
           to configure login module service instances</description>
    </serviceProvider>

    <serviceProvider name="policy.xml" type="POLICY_STORE"
          class="oracle.security.jps.internal.
          policystore.xml.XmlPolicyStoreProvider">
      <description>XML-based PolicyStore</description>
```

```
</serviceProvider>

<serviceProvider type="POLICY_STORE" name="policy.oid"
        class="oracle.security.jps.internal.
        policystore.ldap.LdapPolicyStoreProvider">
    <description>LDAP-based PolicyStore</description>
    <property name="policystore.type" value="OID"/>
    <property name="connection.pool.maxsize" value="30"/>
    <property name="connection.pool.provider.type" value="idmpool"/>
  </serviceProvider>

  <serviceProvider type="AUDIT" name="audit.provider"
       class="oracle.security.jps.internal.audit.AuditProvider">
      <description>Audit Service</description>
  </serviceProvider>
</serviceProviders>

<serviceInstances>

  <serviceInstance name="credstore" provider="credstoressp" location="./">
        <description>File Based Credential Store Service Instance</description>
  </serviceInstance>

  <serviceInstance name="idstore.xml" provider="idstore.xml.provider">
<!-- Subscriber name must be defined for XML Identity Store -->
    <property name="subscriber.name" value="jazn.com"/>
<!-- This is the location of XML Identity Store -->
    <property name="location" value="./user-data.xml"/>
<!-- This property set defines the authenticated role -->
    <propertySetRef ref="authenticated.role.properties"/>
  </serviceInstance>
  <serviceInstance name="idstore.loginmodule"
        provider="jaas.login.provider">
        <description>Identity Store Login Module</description>
    <property name="loginModuleClassName" value="oracle.security.jps.internal.
          jaas.module.idstore.IdStoreLoginModule"/>
    <property name="jaas.login.controlFlag" value="REQUIRED"/>
    <property name="debug" value="true"/>
    <property name="addAllRoles" value="true"/>
  </serviceInstance>

  <serviceInstance name="policystore.rdbms" provider="policy.rdbms">
      <property name="jdbc.url"
         value="jdbc:oracle:thin:@scl58116.us.oracle.com:1521:orcl"/>
      <property name="jdbc.driver" value="oracle.jdbc.driver.OracleDriver"/>
      <property name="bootstrap.security.principal.key" value="keyname"/>
      <property name="bootstrap.security.principal.map" value="mapname"/>
      <property name="oracle.security.jps.ldap.root.name"
         value="cn=jpsTestNode"/>
      <property name="oracle.security.jps.farm.name"
         value="cn=wcai_view_jing.atzsrg"/>
  </serviceInstance>

  <serviceInstance name="policystore.rdbms.ds" provider="policy.rdbms">
    <property name="oracle.security.jps.ldap.root.name"
        value="cn=jpsTestNode"/>
    <property name="oracle.security.jps.farm.name"
        value="cn=wcai_view_jing.atzsrg"/>
    <property value="atzsrgds" name="datasource.jndi.name"/>
  </serviceInstance>
```

```
    <serviceInstance name="pdp.service" provider="pdp.service.provider">
       <property name="oracle.security.jps.runtime.pd.client.sm_name"
             value="${atzsrg.pdp.configuration_id}"/>
       <property name="oracle.security.jps.pdp.
             AuthorizationDecisionCacheEnabled" value="true"/>
       <property name="oracle.security.jps.pdp.
             AuthorizationDecisionCacheEvictionCapacity" value="500"/>
       <property name="oracle.security.jps.pdp.
             AuthorizationDecisionCacheEvictionPercentage" value="10"/>
       <property name="oracle.security.jps.pdp.
             AuthorizationDecisionCacheTTL" value="60"/>
       <property name="oracle.security.jps.ldap.
             policystore.refresh.interval" value="30000"/>
       <property name="oracle.security.jps.policystore.
             refresh.purge.timeout" value="600000"/> <!-- 10 minutes -->
       <property name="loading_attribute_backward_compatible" value="false"/>
  <!-- Properties for controlled mode PD -->
       <property name="oracle.security.jps.runtime.
             pd.client.policyDistributionMode" value="non-controlled"/>
       <property name="oracle.security.jps.runtime.
             instance.name" value="${atzsrg.pdp.instance_name}"/>
    </serviceInstance>

    <serviceInstance name="policystore.oid" provider="policy.oid">
      <property name="max.search.filter.length" value="4096"/>
      <property name="bootstrap.security.principal.key" value="keyname"/>
      <property name="bootstrap.security.principal.map" value="mapname"/>
      <property name="ldap.url" value="ldap://scl58126.us.oracle.com:3060"/>
      <property name="oracle.security.jps.ldap.root.name"
         value="cn=jpsTestNode"/>
      <property name="oracle.security.jps.farm.name"
         value="cn=wcai_view_jing.atzsrg"/>
      <property name="oracle.security.jps.policystore.resourcetypeenforcementmode"
         value="Lenient"/>
    </serviceInstance>

    <serviceInstance name="policystore.xml" provider="policy.xml"
                location="./system-jazn-data.xml"/>

    <serviceInstance name="user.authentication.loginmodule"
                provider="jaas.login.provider">
           <description>User Authentication Login Module</description>
           <property name="loginModuleClassName"
                value="oracle.security.jps.internal.
                jaas.module.authentication.JpsUserAuthenticationLoginModule"/>
           <property name="jaas.login.controlFlag" value="REQUIRED"/>
    </serviceInstance>

    <serviceInstance name="user.assertion.loginmodule"
                provider="jaas.login.provider">
           <description>User Assertion Login Module</description>
           <property name="loginModuleClassName"
                value="oracle.security.jps.internal.
                jaas.module.assertion.JpsUserAssertionLoginModule"/>
           <property name="jaas.login.controlFlag" value="REQUIRED"/>
    </serviceInstance>

    <serviceInstance name="pip.service.ootb.ldap" provider="pip.service.provider">
           <property name="type" value="LDAP_PIP"/>
```

```
        <property name="ldap.url" value="ldap://scl58126.us.oracle.com:3060"/>
        <property name="bootstrap.security.principal.key" value="keyname"/>
        <property name="bootstrap.security.principal.map" value="mapname"/>
        <property name="search.base" value="cn=pip_info_store,
             cn=wcai_view_jing.atzsrg,cn=JPSContext,cn=jpsTestNode"/>
        <property name="failed.server.retry.interval" value="10"/>
 </serviceInstance>
<!-- JPS Audit Service Instance-->
 <serviceInstance name="audit" provider="audit.provider">
        <property name="audit.filterPreset" value="None"/>
        <property name="audit.maxDirSize" value ="0"/>
        <property name="audit.maxFileSize" value ="104857600"/>
        <property name="audit.loader.jndi" value="jdbc/AuditDB"/>
        <property name="audit.loader.interval" value="15" />
        <property name="audit.loader.repositoryType" value="File" />
 </serviceInstance>

 <serviceInstance name="pip.service.ootb.db" provider="pip.service.provider">
        <property name="type" value="RDBMS_PIP"/>
        <property name="jdbc.url"
             value="jdbc:oracle:thin:@scl58116.us.oracle.com:1521:orcl"/>
        <property name="jdbc.driver" value="oracle.jdbc.driver.OracleDriver"/>
        <property name="bootstrap.security.principal.key" value="keyname"/>
        <property name="bootstrap.security.principal.map" value="mapname"/>
        <property name="failed.server.retry.interval" value="10"/>
 </serviceInstance>

 <serviceInstance name="pip.service.ootb.db.ds" provider="pip.service.provider">
        <property name="type" value="RDBMS_PIP"/>
        <property value="atzsrgds" name="datasource.jndi.name"/>
        <property name="failed.server.retry.interval" value="10"/>
 </serviceInstance>

</serviceInstances>

    <jpsContexts default="default">
        <jpsContext name="default">
            <serviceInstanceRef ref="policystore.oid"/>
            <serviceInstanceRef ref="pdp.service"/>
            <serviceInstanceRef ref="audit"/>
            <serviceInstanceRef ref="idstore.xml"/>
            <serviceInstanceRef ref="idstore.loginmodule"/>
            <serviceInstanceRef ref="pip.service.ootb.ldap"/>
            <serviceInstanceRef ref="pip.service.ootb.db"/>
        </jpsContext>
        <jpsContext name="smsec">
            <serviceInstanceRef ref="credstore"/>
        </jpsContext>

    </jpsContexts>

</jpsConfig>
```

Example 6–4 illustrates how the `serviceProvider` element defines the use of a predefined Attribute Retriever by defining the internal Oracle Entitlements Server class.

Example 6–4 Declaring the Predefined Attribute Retriever

```
<serviceProvider
```

```
  class="oracle.security.jps.az.internal.runtime.provider.PIPServiceProvider"
  name="pip.service.provider" type="PIP"/>
```

The following examples illustrate how to modify the `serviceInstance` element for the predefined Attribute Retriever being used.

- Example 6–5, "Using the Predefined LDAP Attribute Retriever"
- Example 6–6, "Using the Predefined RDBMS Attribute Retriever with JDBC"
- Example 6–7, "Using the Predefined RDBMS Attribute Retriever with SQL"
- Example 6–8, "Declaring the Predefined Attribute Retriever in jpsContext"
- Example 6–9, "Enabling an Attribute's Cache"
- Example 6–10, "Configuring LDAP Failover"

Example 6–5 illustrates how to modify the `serviceInstance` element when using the predefined LDAP Attribute Retriever.

Example 6–5 Using the Predefined LDAP Attribute Retriever

```
<serviceInstance name="pip.service.ootb.ldap" provider="pip.service.provider">
   <property name="type" value="RDBMS_PIP"/>
   <property name="ldap.url" value="ldap://dadvmg0065.us.oracle.com:3080"/>
   <property name="bootstrap.security.principal.key" value="keyname"/>
   <property name="bootstrap.security.principal.map" value="mapname"/>
   <property name="failed.server.retry.interval" value="10"/>
</serviceInstance>
```

The following two examples illustrate how to modify the `serviceInstance` element when using the predefined RDBMS Attribute Retriever. Example 6–6 is when using Java Database Connectivity (JDBC) API.

Example 6–6 Using the Predefined RDBMS Attribute Retriever with JDBC

```
<serviceInstance name="pip.service.ootb.db" provider="pip.service.provider">
    <property name="type" value="RDBMS_PIP"/>
    <property name="jdbc.url"
     value="jdbc:oracle:thin:@scl58116.us.oracle.com:1521:orcl"/>
    <property name="jdbc.driver" value="oracle.jdbc.driver.OracleDriver"/>
    <property name="bootstrap.security.principal.map" value="mapname"/>
    <property name="bootstrap.security.principal.key" value="keyname"/>
    <property name="failed.server.retry.interval" value="10"/>
</serviceInstance>
```

Example 6–7 is when using a SQL database.

Example 6–7 Using the Predefined RDBMS Attribute Retriever with SQL

```
<serviceInstance name="pip.service.ootb.db" provider="pip.service.provider">
    <property name="type" value="RDBMS_PIP"/>
    <property name="datasource.jndi.name" value="DB_RAC"/>
    <property name="failed.server.retry.interval" value="10"/>
</serviceInstance>
```

Example 6–8 illustrates how to declare the predefined Attribute Retriever reference in the `jpsContext` element. This sample defines a predefined RDBMS Attribute Retriever.

Example 6–8 Declaring the Predefined Attribute Retriever in jpsContext

```
  <jpsContext name="default">
```

```
        <serviceInstanceRef ref="policystore.db"/>
        <serviceInstanceRef ref="pdp.service"/>
        <serviceInstanceRef ref="audit"/>
        <serviceInstanceRef ref="idstore.xml"/>
        <serviceInstanceRef ref="idstore.loginmodule"/>
        <serviceInstanceRef ref="pip.service.ootb.db"/>
</jpsContext>
```

Example 6–9 illustrates how to configure the caching of a specific attribute value. Caching is enabled per attribute. In this example, the cache record is deleted after 60 seconds.

Example 6–9 Enabling an Attribute's Cache

```
<propertySet name="ootb.pip.attribute.gender.ldap">
        <property name="ootb.pip.attr.type" value="OOTB_PIP_ATTRIBUTE"/>
        <property name="ootb.pip.ref" value="pip.service.ootb.ldap"/>
        <property name="name" value="oespipgender"/>
        <property name="query" value="(oespipage=%oespipage%)"/>
        <property name="cached" value="true"/>
        <property name="ttl" value="60"/>
</propertySet>
```

Example 6–10 illustrates how to configure the failover behavior. In this example, the primary connection is `ldap://dadvmg0065:3080` and the backup connection is `ldap://scl58123:3060`. The failed server retry interval is 10 seconds.

Example 6–10 Configuring LDAP Failover

```
<serviceInstance name="pip.service.ootb.ldap" provider="pip.service.provider">
      <property name="type" value="LDAP_PIP"/>
      <property name="ldap.url"
         value="ldap://dadvmg0065:3080,ldap://scl58123:3060"/>
      <property name="bootstrap.security.principal.key" value="keyname"/>
      <property name="bootstrap.security.principal.map" value="mapname"/>
      <property name="failed.server.retry.interval" value="10"/>
</serviceInstance>
```

6.4 Setting Up PIP Connection Credentials

As documented in Table 6–1, " LDAP Attribute Retriever Parameters" and Table 6–2, " RDBMS Attribute Retriever Parameters", the `bootstrap.security.principal.key` and `bootstrap.security.principal.map` parameters define the key and the map (respectively) to access the data store. Oracle Entitlements Server ships with `oesPassword.sh` which sets these LDAP and database connection credentials in the bootstrap credential store. The tool is located in the *$OES_SM_INSTANCE_DIRECTORY*`/bin/` directory. Use the following command to run it.

```
./oesPassword.sh -setpass
```

It prompts for the security principal key name, the security principal map name, the username and associated password.

7

Managing Policy Distribution

Policy distribution comprises the process used to make configured policies and policy data available for evaluation. Evaluation of the policies will produce a *grant* or *deny* authorization decision in answer to an access request. This chapter contains the following sections.

7.1 Understanding Policy Distribution

Managing policies and distributing them are distinct operations. Policy management operations are used to define, modify and delete policies in the policy store. The Policy Distribution Component then makes the policies available to a Security Module where the data is used to grant or deny access to a protected resource. Policies are not enforced until they are distributed. Policy distribution may include any or all of the following actions:

- Reading policies from the policy store.
- Caching policy objects in the in-memory policy cache maintained by the Security Module for use during authorization request processing.
- Perserving policy objects in a file-based persistent cache, local to the Policy Distribution Component, that provides independence from the policy store.

Both the central Oracle Entitlements Server Administration Console and the locally-installed (to the protected application) Security Module contain the Policy Distribution Component. This architecture allows two deployment scenarios: the first involves a centralized Policy Distribution Component that can communicate with many Security Modules while the second involves a Policy Distribution Component that is local to, and communicates with, one Security Module.

> **Note:** For details on configuring a Security Module for policy distribution, see Section A.1, "Policy Distribution Configuration." For details on creating definitions and binding Security Modules, see Section 8.2, "Configuring Security Module Definitions."

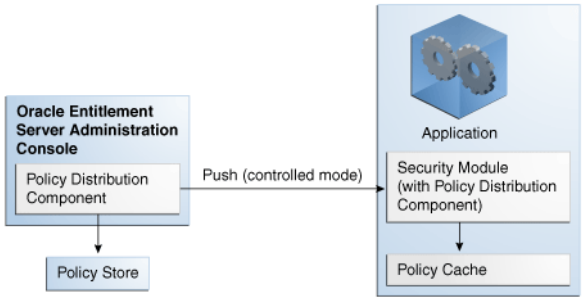
The following sections contain more information.

- Using a Central Policy Distribution Component
- Using a Local Policy Distribution Component

7.1.1 Using a Central Policy Distribution Component

The centralized Policy Distribution Component scenario involves the use of the Policy Distribution Component (within the Administration Console) to act as a server communicating with the Security Module's Policy Distribution Component client. Figure 7–1 illustrates how, in this scenario, the Security Module's Policy Distribution Component client does not communicate with the policy store. The distribution of policies is initiated by the Oracle Entitlements Server administrator and *pushed* to the Policy Distribution Component client. Currently, data can only be pushed in a *controlled* manner as described in Section 7.2.1, "Controlled Distribution." This scenario allows for a central Policy Distribution Component that can communicate with many Security Modules.

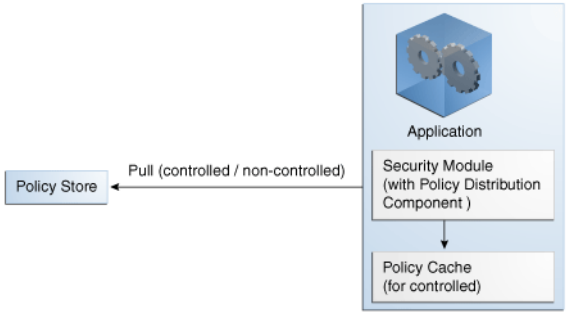
Figure 7–1 Using Oracle Entitlements Server Policy Distribution Component



7.1.2 Using a Local Policy Distribution Component

The local (to the Security Module) scenario involves the Security Module's Policy Distribution Component communicating directly with the policy store. This scenario allows for a local Policy Distribution Component to communicate with one Security Module only. The application administers management operations and decides when the Security Module instance of the Policy Distribution Component will distribute policies or policy deltas. In this deployment, as illustrated in Figure 7–2, the Policy Distribution Component *pulls* data from the policy store (by periodically checking the policy store for data to be distributed) and sends policy data from the policy store, making it available to the PDP after administrator-initiated policy distribution.

Figure 7–2 Using Security Module Policy Distribution Component



Currently, data can be pulled in either a controlled manner as described in Section 7.2.1, "Controlled Distribution" or a non-controlled manner as described in Section 7.2.2, "Non-controlled Distribution."

7.2 Defining Distribution Modes

Oracle Entitlements Server handles the task of distributing policies to individual Security Modules that protect applications and services. Policy data is distributed in either a controlled manner or a non-controlled manner. The distribution mode is defined in the `jps-config.xml` configuration file for each Security Module. The specified distribution mode is applicable for all Application objects bound to that Security Module. The following sections have more information on the distribution modes.

- Controlled Distribution
- Non-controlled Distribution

7.2.1 Controlled Distribution

Controlled distribution is the default distribution mode. It is initiated by the Policy Distribution Component, ensuring that the PDP client (Security Module) receives policy data that has been created or modified since the last distribution. In this respect, distribution is controlled by the policy administrator who takes explicit action to distribute the new or updated policy data. (The Policy Distribution Component maintains a versioning mechanism to keep track of policy changes and distribution.) When controlled distribution is enabled, the Security Module cannot request distribution of the Policy Distribution Component directly.

Note: The exception is when a Security Module starts and registers itself with the Policy Distribution Component with a Configuration ID. The policies are distributed to the Security Module based on this registration.

With controlled distribution, the Policy Distribution Component distributes new and updated policy data to the Security Module where the data is stored in a local persistent cache, a file-based cache maintained by the PDP to store policy objects and provide independence from the policy store. The Policy Distribution Component does not maintain constant live connections to its Security Module clients; it will establish a connection before distributing policy to it. Thus, the Security Module is not dependent on the policy store for making policy decisions; it can use its own local cache if the policy store is offline. When the Security Module starts, it will check if the policy store is available. If it is not available, the Security Module will use policy data from the local persistent cache.

Caution: Controlled distribution is supported only on database type policy stores - not on LDAP-based policy stores. If the distribution API is invoked for an LDAP policy store, it will be non-operable.

With controlled distribution, if any policy distribution operation fails, the entire policy distribution fails. By default, controlled distribution is disabled.

7.2.2 Non-controlled Distribution

When the PDP client (Security Module) periodically retrieves (or pulls) policies and policy modifications from a policy store, it is referred to as non-controlled distribution. Non-controlled distribution makes policy changes available as soon as they are saved to the policy store. Non-controlled distribution is initiated by the Security Module and

may retrieve policies that are not yet complete. The policy store must be online and constantly available for non-controlled distribution. Non-controlled distribution is supported on any policy store type.

7.3 Distributing Policies

From a high level, the following steps are needed to get to the point where you can distribute policies.

1. Create a Security Module definition.

   See Section 8.2.1, "Creating a Security Module Definition."

2. Bind the definition to the appropriate Application.

   See Section 8.2.2, "Binding an Application to a Security Module." To unbind the Security Module, see Section 8.2.3, "Unbinding an Application From a Security Module."

3. Open the Application in the Home area.

   See Section 7.3.1, "Distributing Policies Using the Administration Console."

4. Distribute the policies.

   See Section 7.3.1, "Distributing Policies Using the Administration Console."

7.3.1 Distributing Policies Using the Administration Console

Policies are distributed from within an Application. The following procedure documents how to distribute policies using the Administration Console.

1. Expand the **Applications** node in the Navigation Panel.
2. Select the Application to modify.
3. Right-click the Application name and select **Open** from the menu.

   The General tab, the Delegated Administrators tab and the Policy Distribution tab are all active.

4. Click the Policy Distribution tab.
5. Select the definition of the Security Module to which you will distribute policies.
6. Click Distribute.
7. Click Refresh to update the distribution progress.

8

Managing System Configurations

Security Module definitions and administrator configurations are defined within the top-level System Configuration tab in the Authorization Policy Manager Administration Console. This chapter contains the following topics:

- Section 8.1, "Delegating With Administrators"
- Section 8.2, "Configuring Security Module Definitions"

8.1 Delegating With Administrators

Administrator Roles can be created to delegate management operations for policy objects. For example, Application and Policy Domain delegating administrators can be defined by creating an Administrator Role at the appropriate level and assigning the role Administration Privileges as well as a user, group, or another role. See Chapter 9, "Delegating With Administrator Roles" for more information. It includes a section on creating System Administrator Roles which can manage other types of Administrator Roles in any Application or Policy Domain.

8.2 Configuring Security Module Definitions

A Security Module is an Oracle Entitlements Server client that plays a key role in authorization. After an authorization request is generated, the Security Module evaluates policy data to determine if access to the resource will be granted or denied. An Application (the Oracle Entitlements Server object that represents the protected reosurce) must be *bound* to the Security Module that protects it. Binding Security Modules enables policy data to be transmitted to it for evaluation. The Policy Distribution Component (discussed in Chapter 7, "Managing Policy Distribution") is the mechanism used to transmit policy data to the Security Modules.

> **Note:** For more information about the authorization process, see Section 1.4, "How Oracle Entitlements Server Processes Authorization Policies."

The following sections document how to bind (and unbind) Security Module definitions to (and from) Application objects.

- Creating a Security Module Definition
- Binding an Application to a Security Module
- Unbinding an Application From a Security Module
- Deleting a Security Module Definition

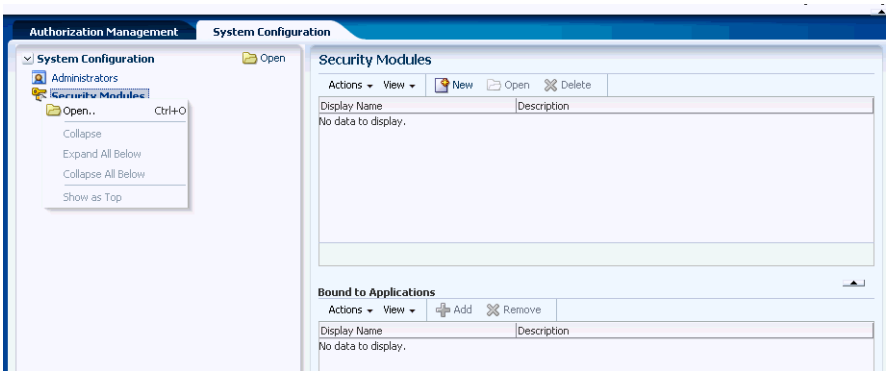
8.2.1 Creating a Security Module Definition

To create a security module, proceed as follows.

1. Select the **System Configuration** tab from the Home area.
2. Double-click **Security Modules** in the Navigation Panel.

   Alternately, right-click Security Modules and select Open. The Security Modules page is displayed as in Figure 8–1.

Figure 8–1 Security Modules in Home Area



3. Click **New** to create a new Security Module definition.

   Alternately, select **New** from the Actions menu. The Security Module dialog is displayed.
4. Provide the following values for the new Security Module.
   - **Name**: The entry must be a unique.
   - **Display Name**
   - **Description**
5. Click **Save**.

8.2.2 Binding an Application to a Security Module

To bind an Application to a Security Module, proceed as follows.

1. Select the **System Configuration** tab from the Home area.
2. Double-click **Security Modules** in the Navigation Panel.

   Alternately, right-click Security Modules and select Open. The Security Modules page is displayed.
3. Select the name of the Security Module definition from the table.
4. Click **Add** in the Bound to Applications table., either cor select **Add** from the **Actions** menu.

   Alternately, select Add from the Bound to Applications Actions menu. The Add Applications dialog displays.
5. Enter a search string in the text box and click the arrow to search.

   Alternately, click the arrow with no search string to return all available Applications.

6. Select one or more applications from the list returned.
7. Click **Add**.

   The selected applications are bound to the selected Security Module and displayed in the Bound to Applications table.

8.2.3 Unbinding an Application From a Security Module

To unbind an application from a Security Module, proceed as follows.

1. Select the **System Configuration** tab from the Home area.
2. Double-click **Security Modules** in the Navigation Panel.

   Alternately, right-click Security Modules and select Open. The Security Modules page is displayed.
3. Select the name of the applicable Security Module definition in the table.
4. Select the name of the applicable Application in the Bound to Applications table.
5. Click **Remove** or select **Remove** from the Actions menu.

   A confirmation dialog is displayed.
6. Click Unbind.

8.2.4 Deleting a Security Module Definition

To remove a Security Module definition, proceed as follows.

1. Select the **System Configuration** tab from the Home area.
2. Double-click **Security Modules** in the Navigation Panel.

   Alternately, right-click Security Modules and select Open. The Security Modules page is displayed.
3. Select the name of the applicable Security Module definition in the table.
4. Click **Delete** or select **Delete** from the Actions menu.

   A confirmation dialog is displayed.
5. Click Remove.

9

Delegating With Administrator Roles

System administrative rights and policy management permissions can be delegated from one administrator to another by creating Administrator Roles with restricted rights, or by granting an existing Administrator Role to a user. This chapter documents information on how to delegate policy and system administrative tasks. It contains the following sections:

9.1 About Delegated Administrators

Administration is when one or more authorized rights are granted to someone to do a certain job. Delegation is the ability for that someone to transfer the authorized right that has been granted them to another. In combination, we can define delegating administration as the transference of authorized rights from one to another. In Oracle Entitlements Server, administrators who are authorized to perform a task on policy objects and entities may transfer this right to others using Administration Roles. Administration Roles consist of a subject (the person to whom the role is granted), the resources (the objects to which the role pertains) and actions (view, manage/modify).

> **Note:** See Section 1.5.1, "Role-based Access Control (RBAC)" for more details on roles.

Oracle Entitlements Server allows you to define delegating Administrator Roles by assigning Administration Privileges, and mapping external roles and users, to it. When a user is logged in as an Administrator, the Navigation Panel displays only the set of Applications the logged in user is authorized to administer. In point of fact, all objects that a delegating Administrator cannot administer are hidden. Any nondefault delegating Administrator Role can perform management operations if it is granted the Admin Role with VIEW and MANAGE privileges.

Note: A nondefault Administrator Role is any Administrator Role created manually. This would not include Administrator Roles automatically created when you create an Application or a Policy Domain.

The following restrictions also apply to Administrator Roles.

- Non-system level (delegating) Administration Roles can only manage other Administration Roles within its scope. For example, an Administration Role created for Application1 can manage Administration Roles in Application 1 Policy Domains but cannot manage peer Administration Roles in Application1, or any roles in Application2 and its Policy Domains. Scope and granularity are discussed further in Chapter 9.2, "Delegating Using Scope and Granularity."
- System level Administration Roles (as discussed in Chapter 8, "Managing System Configurations") can manage delegating Administration Roles in any Application or Policy Domain.
- Nondefault Administration Roles (again, created manually) cannot manage default Administration Roles in any Application or Policy Domain.

9.2 Delegating Using Scope and Granularity

Delegated administration is all about transferring management of resources and policy objects from one person to another. The scope of the delegation (or range of objects covered by the delegation) is defined in levels. The granularity of administration defines the type of objects managed at each scope. A default Administration Role is automatically created when each scope is created; additional Administration Roles can be created later.

Note: The following is applicable to all default Administration Roles.

- Default Administrator Roles cannot be deleted individually.
- If a Policy Domain is deleted, all Administration Roles (including the default) are deleted.
- If the Application is deleted, all Administration Roles are deleted.
- Privileges assigned to default Administrator Roles cannot be modified.

From highest to lowest, the scopes and applicable granularity are as follows:

- The top-level `SystemAdmin` has privileges to manage system-level resources as well as all policy-related objects. System resources include Administrator Roles, system configurations and Security Module bindings. Policy objects include the Application objects.

Note: System Administrators have rights to all policy objects, including all Application objects and child Policy Domains but they are primarily intended to manage configurations, Application objects, and the bindings between the two.

Information on managing system level Administrator Roles is in Chapter 8, "Managing System Configurations."

- Application administrators have privileges to manage all objects in the Application to which they are assigned. One `ApplicationPolicyAdmin` is generated for each Application that is created. They are primarily intended to delegate the management of policy objects within the Application (including the Policy Domains and its children, such as Functions, Attributes, Application Roles and Resource Types). For more information, see Section 9.3, "Delegating Application Administration."
- Policy Domain administrators have privileges to manage all child objects in the Policy Domain to which they are assigned. One `PolicyDomainAdmin` is generated for each Policy Domain that is created. They are primarily created to delegate the management of policies, permissions and resources within a Policy Domain. For an overview of this concept, see Section 9.4, "Using Policy Domains to Delegate." For additional information, see Section 9.5, "Delegating Policy Domain Administration."

9.3 Delegating Application Administration

The following sections explain how to manage administrators for an Application.

- Adding a Delegated Administrator for An Application
- Modifying or Deleting an Application's Delegated Administrator

9.3.1 Adding a Delegated Administrator for An Application

This procedure documents how to create a new Administrator Role and assign it to the applicable roles or users. To add a delegated administrator to an Application, proceed as follows.

1. Expand the **Applications** node in the Navigation Panel.
2. Select the Application to modify.
3. Right-click the Application name and select **Open** from the menu.

   The General tab, the Delegated Administrators tab and the Policy Distribution tab are all active.
4. Click the Delegated Administrators tab.

   The Application name is listed in the displayed table. Click the arrow next to the Application name to see the default `ApplicationPolicyAdmin` created when the Application object was created. Click the Administrator Role name to display its details, in tabs, below the Delegated Administrators table.

   - Role Details
   - External Role Mapping
   - External User Mapping
5. Click New to create a new Administrator Role.

   Be sure to select the name of the Application to activate New. Alternately, select the Application and select New from the Actions menu. A New Administrator Role dialog is displayed.
6. Provide the following values for the new Administrator Role and click OK.

- **Name**: The entry must be a unique.
- **Display Name**
- **Description**

7. Select the new Administrator Role to activate its configuration tabs.

   The Role Details tab is active.

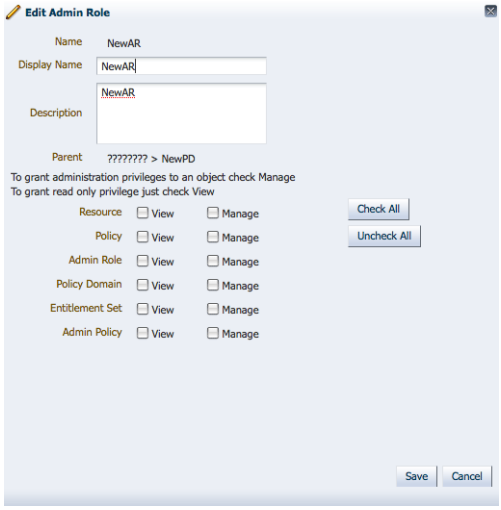
8. Click Edit to define the role details.

   An Edit Administrator Role dialog is displayed.

9. Grant View or Manage privileges for the appropriate policy objects and click Save.

   Figure 9–1 is the Edit Admin Role privileges pop up screen. Select View or Manage for the listed policy objects. For example, Admin Policy allows the administrator to assign new permissions to an Admin Role. Admin Role, however, allows the administrator to assign members to an Admin Role. See Section 2.3, "The Policy Object Glossary" for details on the other listed objects.

Figure 9–1 Edit Admin Role Pop Up Screen



10. Click the External Role Mapping tab to grant the Administrator Role to members of External Roles.
11. Click **Add** to display the **Search Principals** dialog.
12. Complete the query fields in the **External Roles** search box and click **Search**.

    Empty strings fetch all roles. The results display in the **Search** Results table.

13. Select the external role to map to by clicking its name in the table.

    Use **Ctrl+click** to select multiple roles.

14. Click **Add Principals**.

    The selected roles display in the **External Role Mapping** tab.

15. Click the External User Mapping tab to grant the Administrator Role to External Users.
16. Click **Add** to display the **Search Principals** dialog.
17. Complete the query fields in the **Users** search box and click **Search**.

Empty strings fetch all roles. The results display in the **Search** Results table.

18. Select the user to map by selecting its name in the table.

    Use **Ctrl+click** to select multiple roles.

19. Click **Add Principals**.

    The selected roles display in the **External User Mapping** tab.

9.3.2 Modifying or Deleting an Application's Delegated Administrator

To modify or delete an Application's configured Administrator Role, proceed as follows.

1. Expand the **Applications** node in the Navigation Panel.
2. Select the Application to modify.
3. Right-click the Application name and select **Open** from the menu.

   The General tab, the Delegated Administrators tab and the Policy Distribution tab are all active.
4. Click the Delegated Administrators tab.
5. Navigate to the Administrator Role you want to modify and select it.

   The Role Details, External Role Mapping and External User Mapping tabs are displayed.
6. Select the tab which contains the configuration to modify or delete.
   - To modify the configuration, see Section 9.3.1, "Adding a Delegated Administrator for An Application" for details.
   - To remove a mapping from an Administrator Role, select the applicable Administrator Role and the appropriate Mapping tab. Select the mapping and click Remove.
   - To delete an Administrator Role, select the Administrator Role and click Delete.

9.4 Using Policy Domains to Delegate

Administration of the policies securing one protected application may be delegated using one or more (optional) Policy Domains. A Policy Domain contains the components of completed policy definitions. It is the amalgamation of a target Resource (an instance of the Resource Type), an Entitlement (the actions that can be performed on the Resource), and a Policy (a rule that assembles the controls and the principals they affect).

The use of multiple Policy Domains allows policies to be partitioned according to some defined logic, such as the architecture of the protected application or how administration of the policies are delegated. For example, one Policy Domain can be used to maintain all policies securing a Resource or multiple Policy Domains can be used to reflect a particular characteristic of the Resource. Different administrators can then be placed in charge of different Policy Domains.

Note: Because the creation of a Policy Domain is optional, if there is no need to delegate policy administration, there is no need to create any Policy Domains. In this case, a default Policy Domain is created with each Application that will contain all the Application's policy objects.

The following sections contain the management procedures for Policy Domains.

- Creating a Policy Domain
- Modifying a Policy Domain
- Deleting a Policy Domain

9.4.1 Creating a Policy Domain

To create a Policy Domain, proceed as follows.

1. Right-click the name of the Application in the Navigation Panel under which the Policy Domain will be created and select **New** from the menu.

   An Untitled page displays in the Home area.

2. Provide the following information for the Policy Domain.
   - **Display Name**
   - **Name**
   - **Description**: Although optional, it is recommended to provide useful information about the entitlement.
3. Select one of the following from the Save menu.
   - **Save and Close** saves the configuration and renames the tab with the value provided for the Policy Domain's Display Name.
   - **Save and Create Another** saves the configuration to the information tree in the Navigation Panel but leaves the Untitled area open for you to create another Application.

9.4.2 Modifying a Policy Domain

To modify a Policy Domain, proceed as follows.

1. Navigate to the Application under which the Policy Domain you want to delete was created and expand the information tree.
2. Double click the name of the Policy Domain you want to modify.

   The Policy Domain configuration displays in the Home area.
3. Modify as necessary and click Apply.

9.4.3 Deleting a Policy Domain

To delete a Policy Domain, proceed as follows.

1. Navigate to the Application under which the Policy Domain you want to delete was created and expand the information tree.
2. Double click the name of the Policy Domain you want to delete.

The Policy Domain configuration displays in the Home area.

3. Click Delete.

   A confirmation dialog is displayed.

4. Click OK to delete.

9.5 Delegating Policy Domain Administration

The following sections describe how to manage administrators for Policy Domains.

- Adding a Delegated Administrator to a Policy Domain
- Modifying or Deleting a Policy Domain's Delegated Administrator

9.5.1 Adding a Delegated Administrator to a Policy Domain

This procedure documents how to create a new Administrator Role and assign it to the applicable roles or users. To add a delegated administrator to a Policy Domain, proceed as follows.

1. Expand the **Applications** node in the Navigation Panel.
2. Select the Application to modify.
3. Right-click the Application name and select **Open** from the menu.

   The General tab, the Delegated Administrators tab and the Policy Distribution tab are all active.

4. Click the Delegated Administrators tab.

   The Policy Domain names are listed in the displayed table. Clicking the arrow next to the Policy Domain expands the hierarchy and displays any Administrator Roles already configured; for example, the default `PolicyDomainAdmin`.

5. Select the Policy Domain under which you will create the Administrator Role.
6. Click New to create a new Administrator Role.

   Be sure to select the name of the Policy Domain to activate New. Alternately, select the Policy Domain and select New from the Actions menu. A New Administrator Role dialog is displayed.

7. Provide the following values for the new Administrator Role and click OK.
   - **Name**: The entry must be a unique.
   - **Display Name**
   - **Description**
8. Select the new Administrator Role to activate its configuration tabs.

   The Role Details tab is active.

9. Click Edit to define the role details.

   An Edit Administrator Role dialog is displayed.

10. Grant View or Manage privileges for the appropriate Policy Domain objects and click Save.
11. Click the External Role Mapping tab.
    a. Click **Add** to display the **Search Principals** dialog.

b. Complete the query fields in the **External Roles** search box and click **Search**.

   Empty strings fetch all roles. The results display in the **Search** Results table.

c. Select the external role to map to by clicking its name in the table.

   Use **Ctrl+click** to select multiple roles.

d. Click **Add Principals**.

   The selected roles display in the **External Role Mapping** tab.

12. Click the External User Mapping tab.

a. Click **Add** to display the **Search Principals** dialog.

b. Complete the query fields in the **Users** search box and click **Search**.

   Empty strings fetch all roles. The results display in the **Search** Results table.

c. Select the user to map by selecting its name in the table.

   Use **Ctrl+click** to select multiple roles.

d. Click **Add Principals**.

   The selected roles display in the **External User Mapping** tab.

9.5.2 Modifying or Deleting a Policy Domain's Delegated Administrator

To modify or delete a Policy Domain's configured Administrator Role, proceed as follows.

1. Expand the **Applications** node in the Navigation Panel.
2. Select the Application to modify.
3. Right-click the Application name and select **Open** from the menu.

   The General tab, the Delegated Administrators tab and the Policy Distribution tab are all active.

4. Click the Delegated Administrators tab.
5. Navigate to the Administrator Role you want to modify and select it.

   The Role Details, External Role Mapping and External User Mapping tabs are displayed.

6. Select the tab which contains the configuration to modify or delete.

   - To modify the configuration, see Section 9.5.1, "Adding a Delegated Administrator to a Policy Domain" for details.
   - To remove a mapping from an Administrator Role, select the applicable Administrator Role and the appropriate Mapping tab. Select the mapping and click Remove.
   - To delete an Administrator Role, select the Administrator Role and click Delete.

9.6 Managing System Administrators Using Administrator Roles

You can delegate system administration privileges to users by creating and configuring System Administrator Roles. By default, `SystemAdmin` is created during installation and is displayed in the System Administrators table when you navigate to System Administrators under the main System Configuration tab. `SystemAdmin`

manages system-level resources (including other Administrator Roles, and system configurations and bindings) and maps to the WebLogic Server `weblogic` user.

The following sections document the management operations for all Oracle Entitlements Server System Administrator Roles.

- Creating a New Administrator Role
- Assigning Privileges to an Administrator Role
- Modifying Administrator Role Membership
- Deleting an Administrator Role

9.6.1 Creating a New Administrator Role

To create a new Administrator Role, proceed as follows.

1. Select the **System Configuration** tab from the Home area.

   The System Administrators tab is displayed in the Home area.

2. Click **New** under Administrator Roles to create a new Administrator Role.

   A dialog is displayed.

3. Provide the following values for the new Administrator Role.
   - **Name**: The entry must be a unique.
   - **Display Name**
   - **Description**
4. Click **Create**.

9.6.2 Assigning Privileges to an Administrator Role

To assign privileges to an Administrator Role, map external roles, external users or both to the role as documented in this procedure.

1. Select the **System Configuration** tab from the Home area.

   The System Administrators tab and configured Administrator Roles are displayed in the Home area. Alternately, right-click Administrators and select Open.

2. Select the name of the Administrator Role from the table.
3. Select the Modify or View option to define the Administrator Control.

   Modify defines the administrator as having management (and by proxy viewing) privileges on all system administrator resources. View defines the administrator as having only viewing privileges.

4. Click **the External Role Mapping** tab.
   a. Click Add or select Add from the Actions menu.

      The Add Roles search dialog is displayed.

   b. Enter a search string in the text box and click the arrow to search for External Roles.

      Alternately, click Search with no search string to return all available External Roles.

   c. Select one or more roles from the results and click Add Selected.

Alternately, click Add All to add all returned results.

d. Click Add Principals.

5. Click **the External User Mapping** tab.

a. Click Add or select Add from the Actions menu.

The Add Users search dialog is displayed.

b. Enter a search string in the text box and click the arrow to search for External Users.

Alternately, click Search with no search string to return all available External Users.

c. Select one or more users from the results and click Add Selected.

Alternately, click Add All to add all returned results.

d. Click Add Principals.

9.6.3 Modifying Administrator Role Membership

To modify Administrator Role membership, delete the mappings as documented in this procedure.

1. Select the **System Configuration** tab from the Home area.
2. Double-click **System Administrators** in the Navigation Panel.

Alternately, right-click System Administrators and select Open. The System Administrators page is displayed.

3. Select the name of the Administrator Role from the table.
4. Modify the Modify or View Administrator Control as necessary.
5. Click **the External Role Mapping** tab.

a. Select the External Role to delete.

b. Click Remove.

Alternately, select Remove from the Actions menu.

6. Click **the External User Mapping** tab.

a. Select the External User to delete.

b. Click Remove.

Alternately, select Remove from the Actions menu.

9.6.4 Deleting an Administrator Role

To delete an Administrator Role, proceed as follows.

1. Select the **System Configuration** tab from the Home area.
2. Double-click **System Administrators** in the Navigation Panel.

Alternately, right-click System Administrators and select Open. The System Administrators page is displayed.

3. Select the name of the Administrator Role from the table.
4. Click **Delete**.

A confirmation dialog is displayed.

5. Click **Remove**.

10

Customizing the User Interface

This chapter explains several customizations you can make to Oracle Authorization Policy Manager, the Oracle Entitlements Server Administration Console. It contains the following sections:

10.1 Customizing Authorization Policy Manager

All customizations described in this chapter require modifying data in the following files:

```
$ORACLE_HOME$/apm/modules/oracle.security.apm_11.1.1/oracle.security.apm.ear
$ORACLE_HOME$/apm/modules/oracle.security.apm_
   11.1.1/oracle.security.apm.core.view.war
```

Customizations applied to a version of Authorization Policy Manager must be specified anew every time a new version of the tool is installed.

> **Tip:** Before you begin, it is recommended that you backup the Authorization Policy Manager EAR and WAR files listed above.

To customize Authorization Policy Manager, proceed as follows:

1. Unzip the tool's EAR and WAR files, and the view WAR file, as illustrated by the following commands:

   ```
   $ unzip -d $tempDir/ear $ORACLE_HOME$/apm/modules/oracle.security.apm_
      11.1.1/oracle.security.apm.ear
   $ unzip -d $tempDir/war $tempDir/ear/oracle.security.apm.war
   $ unzip -d $tempDir/viewWar $ORACLE_HOME$/apm/modules/
      oracle.security.apm_11.1.1/oracle.security.apm.core.view.war
   ```

2. Modify one or more unzipped files, as explained in the remaining sections of this chapter.

3. Zip anew the tool's EAR and WAR files, and the view WAR file, as illustrated by the following commands:

   ```
   $ zip $tempDir/ear/oracle.security.apm.war $tempDir/war/*
   $ zip $ORACLE_HOME$/apm/modules/oracle.security.apm_
      11.1.1/oracle.security.apm.ear $tempDir/ear/*
   ```

```
$ zip $ORACLE_HOME$/apm/modules/oracle.security.apm_
   11.1.1/oracle.security.apm.core.view.war $temp/viewWar/*
```

4. Redeploy Authorization Policy Manager.

10.2 Customizing Headers, Footers, and Logo

To customize headers, footers, and logo, proceed as follows:

1. Unzip the view WAR file. For details, see Customizing Authorization Policy Manager.
2. Open for edit the file `AuthPolicyMgr.jspx` and apply any of the following modifications, as appropriate.
3. To specify a new branding title, modify the `branding` facet as illustrated in the following snippet:

```
<f:facet name="branding">
 <af:outputText value="My Custom Application Title" noWrap="true" id="ot1"/>
</f:facet>
```

4. To specify a new footer, modify the `appAbout` and `appCopyright` facets as illustrated in the following snippet:

```
<f:facet name="appAbout">
<af:outputText value="My Custom Footer at Right" noWrap="true" id="ot2"/>
</f:facet>
<f:facet name="appCopyright">
<af:outputText value="My Custom Footer at Left"  noWrap="true" id="ot3"/>
</f:facet>
```

5. To specify a new logo image, proceed as follows:
   a. Insert your resource in the `metaContainer` facet as illustrated in the following snippet (leave all other content inside the facet as is):

```
<f:facet name="metaContainer">
....
<af:resource type="css">
.MyCustomBrandingLogo {
background-image:url(/apm/images/world_36x20.png);
 background-position:center;
 background-repeat:no-repeat; display:block;
height:2.5em; width:119px;
           }
 </af:resource>
...
</f:facet>
```

   b. Specify that style class name as the input attribute to the `pageTemplate` tag, as illustrated in the following snippet (leave all other content inside the tag as is):

```
<af:pageTemplate viewId="/templates/IdmShell.jspx"
 value="#{bindings.pageTemplateBinding}" id="pt1">
...
<f:attribute name="brandingLogoCls" value="MyCustomBrandingLogo"/>
...
```

10.3 Customizing Color Schemes

Assuming that you have a new skin available to customize the color scheme, proceed as follows:

1. Unzip the tool's EAR and WAR files. For details, see Customizing Authorization Policy Manager.
2. Open for edit the file Trinidad-config.xml, typically located in the folder WAR/WEB-INF.
3. In that file, specify the value of the new skin in the `skin-family` tag, as illustrated in the following snippet:

```
<trinidad-config xmlns="http://myfaces.apache.org/trinidad/config">
...
<skin-family>MyCustomSkin</skin-family>
...
</trinidad-config>
```

10.4 Customizing the Login Page

To customize the login page and login error page, proceed as follows:

1. Unzip the tool's EAR file. For details, seeCustomizing Authorization Policy Manager.
2. Open for edit the file web.xml, typically located in the folder EAR/WEB-INF.
3. In that file, specify the appropriate values for the `form-login-page` and `form-error-page`, under the element `form-login-config`, as illustrated in the following snippet:

```
<login-config>
<form-login-config>
<form-login-page>/MyCustomLoginPage.html</form-login-page>
<form-error-page> MyCustomLoginErrorPage.html </form-error-page>
</form-login-config>
</login-config>
```

11

Management Tasks

This chapter contains information on managing audit tasks and migrating policies from different types of stores. It contains the following sections:

11.1 Integrating with WebLogic Server

As discussed in Section 1.3.2.2, "Security Module as Combination PDP / PEP," WebLogic Server can automatically intercept authorization requests after enabling the Role Mapping and Authorization providers. The following procedure explains how to do this; it assumes the WebLogic Server is installed in the `$WLS` directory in the `$DOMAIN` domain. Replace the values from your installation when following the procedure.

1. Copy the `jps-atz-wls-proxyproviders.jar` to the WebLogic Server provider definition directory using the following command.

   ```
   cp jps-atz-wls-proxyproviders.jar $WLS/server/lib/mbeantypes
   ```

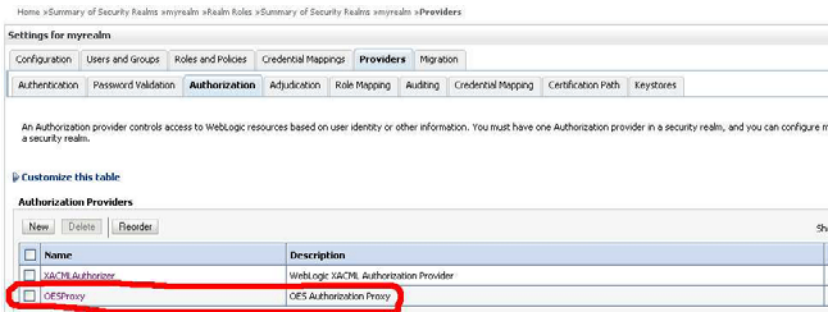
2. Start the `$DOMAIN` domain using the following command.

   ```
   $DOMAIN/startWeblogic.sh
   ```

3. Add the Authorization Proxy and Role Mapping providers to the realm that protects the domain.

   Figure 11–1 is a screenshot of the WebLogic Server console that illustrates this.

Figure 11–1 Adding Providers to the WebLogic Server Domain's Realm



4. Restart the domain.

After enabling the providers, see Section A.2.4, "WebLogic Server Security Module" for the configuration parameters.

11.2 Managing Audit Tasks

Oracle Entitlements Server audits all administrative activities and authorization requests, optionally recording the information to a file. The auditing framework is based on the framework developed for Oracle Platform Security Services. An overview of the Oracle Platform Security Services auditing framework can be found in *Oracle Fusion Middleware Application Security Guide*. The following information is more specific to the auditing functionality in Oracle Entitlements Server.

- Section 11.2.1, "Auditing Events"
- Section 11.2.2, "Configuring Auditing"
- Section 11.2.3, "Additional Auditing Information"

> **Note:** Oracle Entitlements Server will audit decisions resulting from policies configured by itself, Oracle Platform Security Services or any combination thereof.

11.2.1 Auditing Events

Table 11–1 lists the events (organized by functional category) that are audited by Oracle Entitlements Server. Audit logging is disabled by default.

Table 11–1 Events Audited in Oracle Entitlements Server

Functional Category	Functional Task
Administration Role Management	■ AdminRoleCreation ■ AdminRoleDeletion ■ AdminRoleGrant ■ AdminRoleRevoke ■ AdminRoleResActionGrant ■ AdminRoleResActionRevoke
Application Management	■ ApplicationDeletion
Grant Management	■ PermissionSetGrant ■ PermissionSetRevocation

Table 11–1 (Cont.) Events Audited in Oracle Entitlements Server

Functional Category	Functional Task
PermissionSetManagement	■ PermissionSetCreation ■ PermissionSetModification ■ PermissionSetDeletion
PolicyDomainManagement	■ PolicyDomainCreation ■ PolicyDomainDeletion
PolicyManagement	■ PolicyCreation ■ PolicyModification ■ PolicyDeletion ■ PolicyGrant ■ PolicyRevoke
ResourceManagement	■ ResourceCreation ■ ResourceModification ■ ResourceDeletion
Role Management	■ RoleCreation ■ RoleModification ■ RoleDeletion ■ RoleMembershipAdd ■ RoleMembershipRemove
RolePolicyManagement	■ RolePolicyCreation ■ RolePolicyModification ■ RolePolicyDeletion
Authorization	■ CheckPermission ■ IsAccessAllowed ■ CheckSubject
ConfigurationBindingManagement	■ SecurityModuleBinding ■ SecurityModuleUnbinding
ConfigurationManagement	■ SecurityModuleCreation ■ SecurityModuleModification ■ SecurityModuleDeletion
PolicyDistributionManagement	■ PolicyDistribution ■ PdpDeregistration ■ purgeDistributionStatus

11.2.2 Configuring Auditing

Auditing is configured in `jps-config.xml`, the configuration file used by Java EE containers. It is located in the *$DOMAIN_HOME*`/config/fmwconfig` directory. You can define a filterPreset level, a repository type and other information as illustrated in Example 11–1.

Example 11–1 Audit Service Configuration Parameters in jps-config.xml

```
<!-- JPS Audit Service Instance-->
<serviceInstance name="audit" provider="audit.provider">
```

```
        <property name="audit.filterPreset" value="None"/>
        <property name="audit.maxDirSize" value ="0"/>
        <property name="audit.maxFileSize" value ="104857600"/>
        <property name="audit.loader.jndi" value="jdbc/AuditDB"/>
        <property name="audit.loader.interval" value="15" />
        <property name="audit.loader.repositoryType" value="File" />
    </serviceInstance>
```

Table 11–2 contains details about the configuration parameters.

Table 11–2 Auditing Parameters in jps-config.xml

Parameter	Description
audit.filterPreset	None (default), Low, Medium, All or Custom
audit.maxDirSize	Controls the size of the directory in which the audit files are written. Takes an integer in bytes.
audit.maxFileSize	Controls the size of the bus stop file in which audit events are written. Takes an integer in bytes.
audit.loader.jndi	When a database is in use, takes a path to the JNDI data source to which audit events are uploaded.
audit.loader.interval	When a database is in use, controls the frequency of the audit loader's upload. Takes an integer in seconds.
audit.loader.RepositoryType	Defines the audit repository type. Takes a value of **File** or **Db**. If type is database (Db), `audit.loader.jndi` must also be defined.

11.2.3 Additional Auditing Information

The following list collects chapter links in other documents with information regarding the auditing framework.

- Introductory material can be found in *Oracle Fusion Middleware Security Guide*.
- You can manage audit policies with the Enterprise Manager user interface or with the WebLogic Scripting Tool (WLST) command-line interface. See *Oracle Fusion Middleware Application Security Guide* for guidance.
- The Oracle Fusion Middleware Audit Framework Reference is in *Oracle Fusion Middleware Security Guide*.
- Additional configuration information is in *Oracle Fusion Middleware Security Guide*.

11.3 Migrating Policies

This section contains information regarding migrating policies from one type of store to another. It contains procedures for the following:

- Section 11.3.1, "Migrating From XML to LDAP"
- Section 11.3.2, "Migrating From LDAP to XML"
- Section 11.3.3, "Migrating From XML to Database"
- Section 11.3.4, "Migrating From Database to XML"

11.3.1 Migrating From XML to LDAP

Following is the procedure to migrate policies from an XML-based policy store to an LDAP-based directory.

1. Modify `jps-config.xml` as described in this sub procedure.

   a. Create a `serviceInstance` for both the source and destination policy stores as illustrated in Example 11–2.

Example 11–2 XML to LDAP serviceInstances for Source and Destination Policy Stores

```
<!-- Source XML-based policy store instance -->
   <serviceInstance name="src.xml" provider="policystore.xml.provider"
      location="mydir/jazn-data.xml">
    <description>File Based Policy Store Service Instance</description>
   </serviceInstance>

<!-- Destination LDAP-based policy store instance -->
   <serviceInstance provider="ldap.policystore.provider"
      name="policystore.ldap.destination">
    <description>Replace: A. myDestDomain and myDestRootName to appropriate
       values according to your destination LDAP directory structure;
       B. ldap://myDestHost.com:3060 with the URL and port
       number of your destination LDAP</description>
       <property value="OID" name="policystore.type"/>
       <property value="bootstrap" name="bootstrap.security.principal.key"/>
       <property value="cn=myDestDomain" name="oracle.security.jps.farm.name"/>
       <property value="cn=myDestRootName"
         name="oracle.security.jps.ldap.root.name"/>
       <property value="ldap://myDestHost.com:3060" name="ldap.url"/>
   </serviceInstance>
```

   b. Create a `serviceInstance` corresponding to the bootstrap credential used to access the destination LDAP directory as illustrated in Example 11–3.

Example 11–3 XML to LDAP serviceInstance for Bootstrap Credential

```
<!-- Bootstrap credentials to access destination LDAP -->
   <serviceInstance location="./bootstrap" provider="credstoressp"
       name="bootstrap.cred">
     <description>Replace location with the full path of the directory
      where the bootstrap file cwallet.sso is located;
      typically found in destinationDomain/config/fmwconfig/</description>
   </serviceInstance>
```

   c. Create a `jpsContext` for both source and destination stores as illustrated in Example 11–4.

Example 11–4 XML to LDAP jpsContext for Source and Destination Policy Stores

```
<jpsContext name="sourceContext">
   <serviceInstanceRef ref="src.xml"/>
</jpsContext>

<jpsContext name="destinationContext">
   <serviceInstanceRef ref="policystore.ldap.destination"/>
</jpsContext>

<jpsContext name="bootstrap_credstore_context">
   <serviceInstanceRef ref="bootstrap.cred"/>
</jpsContext>
```

2. Start the WebLogic Scripting Tool.

There is no need to connect the WebLogic Scripting Tool to the WebLogic Server as the migration command is an offline command.

3. Run the WebLogic Scripting Tool `migrateSecurityStore` command to migrate the policy store and application as follows.

   - To migrate the policy store, run:

     ```
     migrateSecurityStore
        (type="policyStore", src="sourceContext",
         dst="destinationContext",
         configFile="myDir/jps-config.xml")
     ```

     where the following applies:

     – The name of the corresponding `jpsContext` (previously created in Step 1) should be passed to the `src` and `dst` parameters.

     – The name of the jps-config.xml file (previously modified in Step 1) should be passed to the `configFile` parameter. The value must be a fully qualified file name with complete path information.

   - To migrate the application, run:

     ```
     migrateSecurityStore
        (type="appPolicies", src="sourceContext",
         dst="destinationContext",
         configFile="myDir/jps-config.xml",
         srcApp="sourceApplication", dstApp="destinationApplication",
         overWrite="true")
     ```

     where the following applies:

     – The name of the corresponding `jpsContext` (previously created in Step 1) should be passed to the `src` and `dst` parameters.

     – The name of the `jps-config.xml` file (previously modified in Step 1) should be passed to the `configFile` parameter. The value must be a fully qualified file name with complete path information.

     – The name of the application being migrated is the value of the `srcApp` parameter. If this parameter is not passed, all applications with the same application name in both the source and destination policy stores will be migrated.

     – The name that is assigned to the application in the destination policy store is the value of the `dstApp` parameter. If this parameter is not passed, the name of the application in the destination store is the same as the name used in the source store.

     – If the `overWrite` parameter is defined as true, policies specific to the destination application are replaced by policies from the source application. The default value of this parameter is false.

11.3.2 Migrating From LDAP to XML

Following is the procedure to migrate policies from an LDAP-based directory to an XML-based policy store.

1. Modify `jps-config.xml` as described in this sub procedure.

   a. Create a `serviceInstance` for both the source and destination policy stores as illustrated in Example 11–5.

Example 11–5 LDAP to XML serviceInstances for Source and Destination Policy Stores

```
<!-- Source LDAP-based policy store instance -->
   <serviceInstance provider="ldap.policystore.provider"
      name="policystore.ldap.source">
    <description></description>
    <property value="OID" name="policystore.type"/>
    <property value="bootstrap" name="bootstrap.security.principal.key"/>
    <property value="cn=mySourceDomain" name="oracle.security.jps.farm.name"/>
    <property value="cn=mySourceRootName"
      name="oracle.security.jps.ldap.root.name"/>
    <property value="ldap://mySourceHost.com:3060" name="ldap.url"/>
   </serviceInstance>

<!-- Destination XML-based policy store instance -->
   <serviceInstance name="dst.xml" provider="policystore.xml.provider"
    location="/scratch/divyasin/WithPSR/jazn-data-fscm.xml">
    <description>File Based Policy Store Service Instance</description>
   </serviceInstance>
```

b. Create a `serviceInstance` corresponding to the bootstrap credential used to access the destination LDAP directory as illustrated in Example 11–6.

Example 11–6 LDAP to XML serviceInstance for Bootstrap Credential

```
<!-- Bootstrap credentials to access source LDAP -->
   <serviceInstance location="./bootstrap" provider="credstoressp"
      name="bootstrap.cred">
   <description>Replace location with the full path of the directory where the
      bootstrap file cwallet.sso is located; typically found in
      destinationDomain/config/fmwconfig/</description>
   </serviceInstance>
```

c. Create a `jpsContext` for both source and destination stores as illustrated in Example 11–7.

Example 11–7 LDAP to XML jpsContext for Source and Destination Policy Stores

```
<jpsContext name="sourceContext">
   <serviceInstanceRef ref="policystore.ldap.source"/>
</jpsContext>

<jpsContext name="destinationContext">
   <serviceInstanceRef ref="dst.xml"/>
</jpsContext>

<jpsContext name="bootstrap_credstore_context">
   <serviceInstanceRef ref="bootstrap.cred"/>
</jpsContext>
```

2. Start the WebLogic Scripting Tool.

There is no need to connect the WebLogic Scripting Tool to the WebLogic Server as the migration command is an offline command.

3. Run the WebLogic Scripting Tool `migrateSecurityStore` command to migrate the policy store and application as follows.

- To migrate the policy store, run:

```
migrateSecurityStore
   (type="policyStore", src="sourceContext",
```

```
    dst="destinationContext",
    configFile="/scratch/divyasin/WithPSR/jps-config.xml")
```

where the following applies:

- The name of the corresponding `jpsContext` (previously created in Step 1) should be passed to the `src` and `dst` parameters.
- The name of the jps-config.xml file (previously modified in Step 1) should be passed to the `configFile` parameter. The value must be a fully qualified file name with complete path information.

- To migrate the application, run:

```
migrateSecurityStore
   (type="appPolicies", src="sourceContext",
    dst="destinationContext",
    configFile="/scratch/divyasin/WithPSR/jps-config.xml",
    srcApp="sourceApplication", dstApp="destinationApplication",
    overWrite="true")
```

where the following applies:

- The name of the corresponding `jpsContext` (previously created in Step 1) should be passed to the `src` and `dst` parameters.
- The name of the `jps-config.xml` file (previously modified in Step 1) should be passed to the `configFile` parameter. The value must be a fully qualified file name with complete path information.
- The name of the application being migrated is the value of the `srcApp` parameter. If this parameter is not passed, all applications with the same application name in both the source and destination policy stores are migrated.
- The name that is assigned to the application in the destination policy store is the value of the `dstApp` parameter. If this parameter is not passed, the name of the application in the destination store is the same as the name used in the source store.
- If the `overWrite` parameter is defined as true, policies specific to the destination application are replaced by policies from the source application. The default value of this parameter is false.

11.3.3 Migrating From XML to Database

Following is the procedure to migrate policies from an XML-based policy store to a database.

1. Modify `jps-config.xml` as described in this sub procedure.
   a. Create a `serviceInstance` for both the source and destination policy stores as illustrated in Example 11–8.

Example 11–8 XML to Database serviceInstances for Source and Destination Policy Stores

```
<!-- Source XML-based policy store instance -->
   <serviceInstance name="src.xml" provider="policystore.xml.provider"
     location="/scratch/divyasin/WithPSR/jazn-data-fscm.xml">
    <description>File Based Policy Store Service Instance</description>
</serviceInstance>
```

```
<!-- Destination DB-based policy store instance -->
   <serviceInstance provider="ldap.policystore.provider"
     name="policystore.db.destination">
    <description>DB Based Policy Store Service Instance</description>
     <property name="policystore.type" value="DB_ORACLE"/>
     <property name="jdbc.url"
       value="jdbc:oracle:thin:@sc.us.oracle.com:1722:orcl"
    <jdbc:oracle:thin:@sc.us.oracle.com:1722:orcl>/>
     <property name="jdbc.driver" value="oracle.jdbc.driver.OracleDriver"/>
     <property name="bootstrap.security.principal.key"
        value="bootstrap_DWgpEJgXwhDIoLYVZ2OWd4R8wOA=" />
     <property name="oracle.security.jps.ldap.root.name" value="cn=jpsTestNode"/>
     <property name="oracle.security.jps.farm.name" value="cn=view_steph.atz"/>
   </serviceInstance>
```

b. Create a `serviceInstance` corresponding to the bootstrap credential used to access the destination LDAP directory as illustrated in Example 11–9.

Example 11–9 XML to Database serviceInstance for Bootstrap Credential

```
<!-- Bootstrap credentials to access source DB -->
   <serviceInstance location="./bootstrap" provider="credstoressp"
     name="bootstrap.cred">
     <description>Replace location with the full path of the directory
        where the bootstrap file cwallet.sso is located;
        typically found in destinationDomain/config/fmwconfig/</description>
   </serviceInstance>
```

c. Create a `jpsContext` for both source and destination stores as illustrated in Example 11–10.

Example 11–10 XML to Database jpsContext for Source and Destination Policy Stores

```
<jpsContext name="sourceContext">
   <serviceInstanceRef ref="src.xml"/>
</jpsContext>

<jpsContext name="destinationContext">
   <serviceInstanceRef ref="policystore.db.destination"/>
</jpsContext>

<jpsContext name="bootstrap_credstore_context">
   <serviceInstanceRef ref="bootstrap.cred"/>
</jpsContext>
```

2. Start the WebLogic Scripting Tool.

   There is no need to connect the WebLogic Scripting Tool to the WebLogic Server as the migration command is an offline command.

3. Run the WebLogic Scripting Tool `migrateSecurityStore` command to migrate the policy store and application as follows.

   - To migrate the policy store, run:

     ```
     migrateSecurityStore
        (type="policyStore", src="sourceContext",
         dst="destinationContext",
         configFile="/scratch/divyasin/WithPSR/jps-config.xml")
     ```

     where the following applies:

- The name of the corresponding `jpsContext` (previously created in Step 1) should be passed to the `src` and `dst` parameters.
- The name of the jps-config.xml file (previously modified in Step 1) should be passed to the `configFile` parameter. The value must be a fully qualified file name with complete path information.

- To migrate the application, run:

```
migrateSecurityStore
   (type="appPolicies", src="sourceContext",
    dst="destinationContext",
    configFile="/scratch/divyasin/WithPSR/jps-config.xml",
    srcApp="sourceApplication", dstApp="destinationApplication",
    overWrite="true")
```

where the following applies:

- The name of the corresponding `jpsContext` (previously created in Step 1) should be passed to the `src` and `dst` parameters.
- The name of the `jps-config.xml` file (previously modified in Step 1) should be passed to the `configFile` parameter. The value must be a fully qualified file name with complete path information.
- The name of the application being migrated is the value of the `srcApp` parameter. If this parameter is not passed, all applications with the same application name in both the source and destination policy stores are migrated.
- The name that is assigned to the application in the destination policy store is the value of the `dstApp` parameter. If this parameter is not passed, the name of the application in the destination store is the same as the name used in the source store.
- If the `overWrite` parameter is defined as true, policies specific to the destination application are replaced by policies from the source application. The default value of this parameter is false.

11.3.4 Migrating From Database to XML

Following is the procedure to migrate policies from a database to an XML-based policy store.

1. Modify `jps-config.xml` as described in this sub procedure.
   a. Create a `serviceInstance` for both the source and destination policy stores as illustrated in Example 11–11.

Example 11–11 Database to XML serviceInstances for Source and Destination Policy Stores

```
<!-- Source DB-based policy store instance -->
   <serviceInstance provider="ldap.policystore.provider"
     name="policystore.db.source">
    <description>DB Based Policy Store Service Instance</description>
    <property name="policystore.type" value="DB_ORACLE"/>
    <property name="jdbc.url"
      value="jdbc:oracle:thin:@sc.us.oracle.com:1722:orcl"
<jdbc:oracle:thin:@sc.us.oracle.com:1722:orcl>/>
    <property name="jdbc.driver" value="oracle.jdbc.driver.OracleDriver"/>
    <property name="bootstrap.security.principal.key"
```

```
        value="bootstrap_DWgpEJgXwhDIoLYVZ2OWd4R8wOA=" />
    <property name="oracle.security.jps.ldap.root.name" value="cn=jpsTestNode"/>
    <property name="oracle.security.jps.farm.name" value="cn=view_steph.atz"/>
</serviceInstance>

<!-- Destination XML-based policy store instance -->
   <serviceInstance name="dst.xml" provider="policystore.xml.provider"
      location="/scratch/divyasin/WithPSR/jazn-data-fscm.xml">
    <description>File Based Policy Store Service Instance</description>
   </serviceInstance>
```

b. Create a `serviceInstance` corresponding to the bootstrap credential used to access the destination LDAP directory as illustrated in Example 11–12.

Example 11–12 Database to XML serviceInstance for Bootstrap Credential

```
<!-- Bootstrap credentials to access source and destination LDAPs -->
   <serviceInstance location="./bootstrap" provider="credstoressp"
      name="bootstrap.cred">
     <description>Replace location with the full path of the directory where
       the bootstrap file cwallet.sso is located; typically found in
       destinationDomain/config/fmwconfig/</description>
   </serviceInstance>
```

c. Create a `jpsContext` for both source and destination stores as illustrated in Example 11–13.

Example 11–13 Database to XML jpsContext for Source and Destination Policy Stores

```
<jpsContext name="sourceContext">
   <serviceInstanceRef ref="policystore.db.source"/>
</jpsContext>

<jpsContext name="destinationContext">
   <serviceInstanceRef ref="dst.xml"/>
</jpsContext>

<jpsContext name="bootstrap_credstore_context">
   <serviceInstanceRef ref="bootstrap.cred"/>
</jpsContext>
```

2. Start the WebLogic Scripting Tool.

There is no need to connect the WebLogic Scripting Tool to the WebLogic Server as the migration command is an offline command.

3. Run the WebLogic Scripting Tool `migrateSecurityStore` command to migrate the policy store and application as follows.

- To migrate the policy store, run:

```
migrateSecurityStore
   (type="policyStore", src="sourceContext",
    dst="destinationContext",
    configFile="/scratch/divyasin/WithPSR/jps-config.xml")
```

where the following applies:

– The name of the corresponding `jpsContext` (previously created in Step 1) should be passed to the `src` and `dst` parameters.

  - The name of the jps-config.xml file (previously modified in Step 1) should be passed to the `configFile` parameter. The value must be a fully qualified file name with complete path information.

- To migrate the application, run:

```
migrateSecurityStore
   (type="appPolicies", src="sourceContext",
    dst="destinationContext",
    configFile="/scratch/divyasin/WithPSR/jps-config.xml",
    srcApp="sourceApplication", dstApp="destinationApplication",
    overWrite="true")
```

  where the following applies:

  - The name of the corresponding `jpsContext` (previously created in Step 1) should be passed to the `src` and `dst` parameters.
  - The name of the `jps-config.xml` file (previously modified in Step 1) should be passed to the `configFile` parameter. The value must be a fully qualified file name with complete path information.
  - The name of the application being migrated is the value of the `srcApp` parameter. If this parameter is not passed, all applications with the same application name in both the source and destination policy stores are migrated.
  - The name that is assigned to the application in the destination policy store is the value of the `dstApp` parameter. If this parameter is not passed, the name of the application in the destination store will be the same as the name used in the source store.
  - If the `overWrite` parameter is defined as true, policies specific to the destination application are replaced by policies from the source application. The default value of this parameter is false.

11.4 Configuring Cache

Oracle Entitlements Server offers caching capabilites. The cache settings are configured in the `jps-config.xml` file. The following sections contain the appropriate information.

- Section 11.4.1, "Configuring Decision Caching"
- Section 11.4.2, "Configuring Attribute Caching"

11.4.1 Configuring Decision Caching

Authorization decision caching allows Oracle Entitlements Server to cache the result of an authorization call and use that decision in the future, if an identical call is made. The decision cache consists of two hierarchical levels.

- The first level (L1) caches subjects used in the authorization calls.
- The second level (L2) caches authorization and role mapping decisions for the given subject.

Note: The decision cache automatically invalidates itself if there is a change in the policy.

The key of the cache is the incoming Subject, Permission and attributes used during policy evaluation. The value of the cache is the decision and obligations.

All parameter names are prefixed with `oracle.security.jps.pdp`. Example 11–14 illustrates how the decision cache parameters might be set in `jps-config.xml`.

Example 11–14 XML To Configure Decision Caching

```
<serviceInstance name="pdp.service" provider="pdp.service.provider">
          …
 <property name="oracle.security.jps.pdp.AuthorizationDecisionCacheEnabled"
   value="true"/>
 <property name="oracle.security.jps.pdp.
   AuthorizationDecisionCacheEvictionCapacity"
   value="1000"/>
 <property name="oracle.security.jps.pdp.
   AuthorizationDecisionCacheEvictionPercentage"
   value="15"/>
 <property name=" oracle.security.jps.pdp.AuthorizationDecisionCacheTTL"
   value="180"/>
          …
</serviceInstance>
```

Table 11–3 documents the decision caching parameters.

Table 11–3 Decision Caching Parameters

Name	Description	Accepted Values
`oracle.security.jps.pdp.AuthorizationDecisionCacheEnabled`	Optional parameter that specifies whether the policy decision cache should be enabled.	true (default) false
`oracle.security.jps.pdp.AuthorizaionDecisionCacheEvictionCapacity`	Optional parameter that specifies the maximum capacity of the L1 cache. If the number of entries exceeds the value, some entries are evicted.	Integer representing number of entries 500 (default)
`oracle.security.jps.pdp.AuthorizationDecisionCacheEvictionPercentage`	Optional parameter that specifies the percentage of entries in L1 cache that have to be evicted when the maximum capacity has been reached. For example, if the maximum capacity is 200 and the value of this parameter is 10 then 20 entries are evicted from the cache.	Integer representing percent of entries 10 (default equals 10%)
`oracle.security.jps.pdp.AuthotizationDecisionCacheTTL`	Optional parameter that specifies a time-to-live value (in seconds) for entries in the L2 cache. It defines how long an authorization decision is cached.	Integer representing time in seconds 60 (default equals 1 minute)

11.4.2 Configuring Attribute Caching

Each passed attribute can be cached if the `cached` property is defined for it. A corresponding time-to-live (TTL) value must also be defined if `cached` is enabled. The key of the cache is the attribute URI. The value of the cache is the attribute object. Example 11–15 illustrates how the attribute cache might be set in `jps-config.xml`.

Example 11–15 XML To Configure Attribute Caching

```
<propertySet name="ootb.pip.attribute.age.based.on.myattr.rdbms">
    <property name="ootb.pip.attr.type" value="OOTB_PIP_ATTRIBUTE"/>
    <property name="ootb.pip.ref" value="pip.service.ootb.db"/>
```

```
    <property name="name" value=" myattr"/>
    <property name="query" value="select value from table"/>
    <property name="cached" value="true"/>
    <property name="TTL" value="60"/>
</propertySet>
```

Note: If `cached` is not defined for the attribute, it will not be cached.

11.5 Debugging

The following sections contain information on how to debug Authorization Policies created using Oracle Entitlements Server as well as the Policy Distribution Component.

- Configuring Logging for Debugging
- Searching Logs to Debug Authorization Policies
- Debugging Policy Distribution

11.5.1 Configuring Logging for Debugging

Oracle Entitlements Server uses the standard Java logging framework. Logging is the process of notifying an entity of a particular event. In the case of Oracle Entitlements Server, the entity can be a file or the Administration Console, and the event can be debugging information, runtime exceptions, or a record of actions taken by a user. The logging framework is configured based on the Oracle Entitlements Server deployment. More information is in the following sections.

- Configuring Logging for a Java Security Module Deployment
- Configuring Logging for a WebLogic Server Security Module Deployment

Note: The `java.util.logging` package provides the classes and interfaces of the platform's core logging facilities.

11.5.1.1 Configuring Logging for a Java Security Module Deployment

The following configurations must be made to enable logging when using the Java Security Module in your deployment.

- Run the following command when you start the Security Module to specify the logging configuration file:

  ```
  -Djava.util.logging.config.file=logging.properties
  ```

- Set the logging level by adding the following lines to the configuration file:

  ```
  oracle.jps.authorization.level=FINEST
  oracle.jps.openaz.level=FINEST
  ```

  Logging levels define the complexity of the logging record and include (from least to most) VERBOSE (simple information), WARNING, INFO, CONFIG, FINE, FINER and FINEST (complex information).

If you don't specify a configuration file, the `logging.properties` file in *$JAVA_HOME*`/jre/lib/` is used. Example 11–16 illustrates how to configure `logging.properties` to log information to the Administration Console.

Example 11–16 Configuration for Administration Console Logging

```
#The messages will we printed to the standard output
handlers=java.util.logging.ConsoleHandler

#The default level for all loggers is INFO
.level=INFO

#Override the default level for OES authorization to FINEST
oracle.jps.authorization.level=FINEST
oracle.jps.openaz.level=FINEST

#Use default formatter to print the messages
java.util.logging.ConsoleHandler.formatter = java.util.logging.SimpleFormatter
```

Example 11–17 illustrates how to configure `logging.properties` to log information to a file.

Example 11–17 Configuration for File Logging

```
#The messages will be written to a file
handlers=java.util.logging.FileHandler

#The default level for all loggers is INFO
.level=INFO

#Override the default level for OES authorization to FINEST
oracle.jps.authorization.level=FINEST
oracle.jps.openaz.level=FINEST

#Configure file information. %h - is the user home directory
java.util.logging.FileHandler.pattern = %h/java%u.log
java.util.logging.FileHandler.limit = 50000
java.util.logging.FileHandler.count = 1
java.util.logging.FileHandler.formatter = java.util.logging.SimpleFormatter
```

11.5.1.2 Configuring Logging for a WebLogic Server Security Module Deployment

To enable logging when using the WebLogic Server Security Module in your deployment, run the following command to specify the logging configuration file when you start the WebLogic Server domain.

```
startWeblogic.sh -Djava.util.logging.config.file=logging.properties
```

> **Tip:** If you specify a relative path, the base directory is the domain home - not the directory where `startWeblogic.sh` is located

Other configurations relevant to the WebLogic Server Security Module are similar to those defined in Section 11.5.1.1, "Configuring Logging for a Java Security Module Deployment."

11.5.2 Searching Logs to Debug Authorization Policies

The following sections explain how to search for information recorded to the logging file. They include the commands to be run and, in many sections, sample output.

- Searching for PEP Request Information
- Searching for Security Module Cache Configuration Parameters

- Searching for Principals
- Searching for Resources and Actions
- Searching for the Value of an Attribute
- Searching for an Authorization Decision
- Searching for the Value of an Obligation
- Searching for Static Application Roles

11.5.2.1 Searching for PEP Request Information

Run the following command against the logging file to output PEP Request related information (including the Authentic Identity, the Runtime Resource, the Runtime Action and the Application Context).

```
grep "PepRequestImpl"
```

11.5.2.2 Searching for Security Module Cache Configuration Parameters

Run the following command against the logging file to output the cache configuration parameters for a particular Security Module.

```
grep "AuthotizationDecisionCacheTTL"
```

The following properties may be returned for this search. If a property does not appear in the log, it is not specified in `jps-config.xml`. In cases like this, the default value of the property is used.

- `AuthorizationDecisionCacheTTL` defines the time-to-live (in seconds) for the Authorization Decision cache. The default value is 60.
- `AuthorizationDecisionCacheEvictionPercentage` defines the percentage of authorization decisions to drop when the Authorization Decision cache has reached maximum capacity. The default value is 10.
- `AuthorizationDecisionCacheEvictionCapacity` defines the number used to evict the Authorization Decision cache if the decision cache size reaches this size. The default value is 500.
- `AuthorizationDecisionCacheEnabled` specifies whether the Authorization Decision cache is enabled. The default value is *true*.

Example 11–18 illustrates output for this search.

Example 11–18 Sample Output for Cache Configuration Parameters Search

```
oracle.security.jps.az.internal.runtime.service.AbstractPDPService

FINE: properties : {
oracle.security.jps.pdp.AuthotizationDecisionCacheTTL=60,
oracle.security.jps.pdp.AuthorizationDecisionCacheEvictionPercentage=10,
oracle.secuirty.jps.pdp.AuthorizationDecisionCacheEvictionCapacity=1000,
oracle.security.jps.pdp.AuthorizationDecisionCacheEnabled=true}
```

11.5.2.3 Searching for Principals

Run the following command against the logging file to output the names of Principals that have been received by Oracle Entitlements Server in the form of an authorization request.

```
grep "Principal:"
```

Example 11–19 illustrates the output for a Principal search.

Example 11–19 Sample Output for Principal Search

```
com.bea.security.providers.authorization.asi.AuthorizationProviderImpl
isAccessAllowed

FINE:subject: Subject:
     Principal: John
     Principal: Employee
     Principal: Administrator
     Principal: Principal Developer
```

11.5.2.4 Searching for Resources and Actions

Run the following command against the logging file to output the Resources and Actions that have been received by Oracle Entitlements Server in the form of an authorization request.

```
grep "Resource:"
```

Example 11–20 illustrates how the information is returned. The defined values are the name of the policy object.

- Application = Lib
- Resource Type = libraryresourcetype
- Resource = Book
- Action = borrow

Example 11–20 Sample Output for Resource and Action Search

```
com.bea.security.providers.authorization.asi.AuthorizationProviderImpl
isAccessAllowed

FINE: Resource: resource=Lib/libraryresourcetype/Book, action=borrow
```

11.5.2.5 Searching for the Value of an Attribute

Run the following command against the logging file to output the value of an attribute that has been received by Oracle Entitlements Server in the form of an authorization request.

```
grep "<name-of-the-attribute>:"
EXAMPLE: grep "getAttributeInternal:"
```

Example 11–21 illustrates the returned information where the name of the attribute is `NumberOfBorrowedBooksAttribute` and the value is 2.

Example 11–21 Sample Output for the Value of an Attribute Search

```
com.bea.security.providers.authorization.asi.ARME.evaluator.EvalSession logDebug
FINE: getAttributeInternal: name: NumberOfBorrowedBooksAttribute; value: 2; type:
3
```

11.5.2.6 Searching for an Authorization Decision

Run the following command against the logging file to retrieve an authorization decision that has been stored.

```
grep "AccessResultLogger"
```

Example 11–22 illustrates the returned information and confirm that the authorization decision was affirmative.

Example 11–22 Sample Output for Authorization Decision Search

```
com.bea.security.providers.authorization.asi.AccessResultLogger log
FINE: Subject Subject:
Principal: John
Principal: Employee
Principal: Administrator
Principal: Principal Developer
 privilege borrow resource //app/policy/Lib/Book result PERMIT
```

11.5.2.7 Searching for the Value of an Obligation

Run the following command against the logging file to output the value of a specific obligation.

```
grep "adding response attribute:" | grep  "obligations"
```

Example 11–23 illustrates the returned information indicating that the obligation (named `DenyObligation`) denies the request when the amount of library books the Principal currently has checked out is more than three; in this case, the Principal has five books checked out.

Example 11–23 Sample Output for Obligation Value Search

```
com.bea.security.providers.authorization.asi.AuthorizationProviderImpl
ARMEisAccessAllowed
FINE: adding response attribute: namespace=oracle.security.oes.authorization.
 name=obligations value={DenyObligation=
   { reason_part1=Too many borrowed books (max=3), reason_part2=5, }}
```

11.5.2.8 Searching for Static Application Roles

Run the following command against the logging file to output the names of Application Roles granted statically.

```
grep "AbstractRoleManager" | grep "getGrantedStaticAppRoles"
```

Example 11–24 illustrates how two static roles are added to the list of principals: an authenticated-role – build-in role and Reader, an Application Role defined in the Application named Library.

Example 11–24 Sample Output for Static Role Search

```
oracle.security.jps.az.internal.runtime.entitymanager.AbstractRoleManager
getGrantedStaticAppRoles(Set)
FINER: RETURN [authenticated-role,
  ApplicationRoleLibrary/Readeruname:
  cn=Writer,cn=Roles,cn=Lib,cn=akapisni_dwps1_
  view1.atzsrg,cn=JPSContext,cn=jpsTestNode,guid:
  411EBF807CD411E0BF887FB1A0F3878F]
```

11.5.3 Debugging Policy Distribution

The Policy Distribution Component uses the policy management `Logger` interface. To enable debugging for the Policy Distribution Component, change the logging level of the `oracle.jps.policymgmt.level` property in the logging configuration file to FINEST. The procedure is documented in Section 11.5.1, "Configuring Logging for Debugging."

A

Installation and Configuration Parameters

This Appendix lists the parameters and accepted values that may be defined for Oracle Entitlements Server services using `jps-config.xml`, the configuration file used by Java EE containers. It is located in the *$DOMAIN_HOME*`/config/fmwconfig` directory. This Appendix is comprised of the following sections:

- Section A.1, "Policy Distribution Configuration"
- Section A.2, "Security Module Configuration"
- Section A.3, "PDP Proxy Configuration"
- Section A.4, "Policy Store Service Configuration"

A.1 Policy Distribution Configuration

The Policy Distribution Component is responsible for distributing policy objects and policies from the policy store to one or more Security Modules. It can distribute in a controlled-push mode, a controlled-pull mode and a non-controlled mode. Each mode entails different configurations.

- Section A.1.1, "Policy Distribution Component Server Configuration"
- Section A.1.2, "Policy Distribution Component Client Configuration"

A.1.1 Policy Distribution Component Server Configuration

Typically, configuration for the Policy Distribution Component (in a scenario when it runs within Oracle Entitlements Server) is associated with the Policy Store configuration in the `jps-config.xml` file to fetch policies and policy objects for distribution. Only in cases when data is pulled in a controlled manner (*controlled-pull mode*) is the Policy Distribution Component associated with the PDP Service configuration on the Security Module side. Table A–1 contains the configuration parameters.

Table A–1 Policy Distribution Server Configuration

Name	Information
oracle.security.jps.pd.server.transactionalScope	**Description:** Defines the scope of the policy distribution as either to one Security Module or to all Security Modules. If distribution fails when it involves only one Security Module, it does not affect distributions to other Security Modules. **Optional** **Accepted Values:** All (default), One

A.1.2 Policy Distribution Component Client Configuration

The Policy Distribution Component client is responsible for making policies available to the Security Module. Thus, the Policy Distribution Client configuration is always associated with the PDP Service configuration portion of the `jps-config.xml` file on the Security Module side. Configuration is different depending on the mode of distribution and the environment in which the Security Module is running. The following sections contain descriptions of the applicable configuration parameters.

- Section A.1.2.1, "Policy Distribution Component Client Java Standard Edition Configuration (Controlled Push Mode)"
- Section A.1.2.2, "Policy Distribution Component Client Java Enterprise Edition Container Configuration (Controlled Push Mode)"
- Section A.1.2.3, "Policy Distribution Client Configuration (Controlled Pull Mode)"
- Section A.1.2.4, "Policy Distribution Client Configuration (Non-controlled Mode)"

A.1.2.1 Policy Distribution Component Client Java Standard Edition Configuration (Controlled Push Mode)

Table A–2 compiles the parameters for the Policy Distribution Component client configuration when the Oracle Entitlements Server is running in a Java Standard Edition (JSE) environment and is configured to distribute data in the controlled-push mode.

Table A–2 Policy Distribution Client Configuration, JSE, Controlled Push Mode

Name	Information
oracle.security.jps.runtime.pd.client.policyDistributionMode	**Description:** Specifies the mode of policy distribution. *Controlled distribution* is initiated by the Policy Distribution Component, ensuring that the Security Module receives policy data that has been created or modified since the last distribution. **Mandatory** **Accepted Value:** controlled-push
oracle.security.jps.runtime.pd.client.sm_name	**Description:** Defines the name of the Security Module. **Mandatory** **Accepted Value:** Name of the Security Module
oracle.security.jps.runtime.pd.client.localpolicy.work_folder	**Description:** Defines the name of any directory in which local cache files are stored. This directory must have read and write privileges. **Optional** **Accepted Value:** The name of any directory in which local cache files will be stored. This directory must have read and write privileges.

Table A–2 (Cont.) Policy Distribution Client Configuration, JSE, Controlled Push Mode

Name	Information
oracle.security.jps.runtime.pd.client.incrementalDistribution	**Description:** Defines whether the distribution is incremental or flush. *Incremental distribution* is when only new and modified data is distributed. *Flush distribution* is when the Policy Distribution Component notifies the Security Module to cleanup locally stored policies in preparation for a complete re-distribution of all policy objects in the policy store. **Optional** **Accepted Values:** ■ false (policy distribution is flush for this Security Module) ■ true (default value; policy distribution is incremental for this Security Module if the required change logs are kept in the policy store)
oracle.security.jps.runtime.pd.client.registrationRetryInterval	**Description:** When a Security Module starts, it registers itself with the Policy Distribution Component to ensure the local policy cache is up to date. If registration fails, it will retry each time this interval of time passes until successful. **Optional** **Accepted Value:** time in seconds (default value is 5)
oracle.security.jps.runtime.pd.client.waitDistributionTime	**Description:** If this value is defined and not equal to zero, it specifies the amount of time that a Security Module will wait for initial policy distribution to happen. During this wait period, authorization requests are blocked until either the initial policy distribution completes or the configured period expires. **Optional** **Accepted Value:** time in seconds (default value is 60)
oracle.security.jps.runtime.pd.client.RegistrationServerURL	**Description:** Defines the URL of the Oracle Entitlements Server Administration Server. Used by the Security Module to register itself with Oracle Entitlements Server when it starts. **Mandatory** **Accepted Value:** URL
oracle.security.jps.runtime.pd.client.backupRegistrationServerURL	**Description:** Defines a backup URL for the Oracle Entitlements Server Administration Server. Used by the Security Module to register itself with Oracle Entitlements Server when it starts if the primary URL (parameter above) is unavailable. **Optional** (although if not configured Oracle Entitlements Server failover will not work) **Accepted Value:** URL
oracle.security.jps.runtime.pd.client.DistributionServicePort	**Description:** Defines the port to which a remote Policy Distributor will push policy updates. **Mandatory** **Accepted Value:** port number
oracle.security.jps.pd.client.sslMode	**Description:** Defines whether communication between the Policy Distribution Component server and client will use the Secure Sockets Layer (SSL) protocol or not. **Mandatory** **Accepted Values:** none, two-way (default value)

Table A–2 (Cont.) Policy Distribution Client Configuration, JSE, Controlled Push Mode

Name	Information
oracle.security.jps.pd.client.ssl.identityKeyStoreFileName	**Description:** Defines the name of the Identity Key Store file in which client certificates are stored. Used for SSL communication between the Security Module and the Policy Distribution Component. **Mandatory** **Accepted Value:** the name of the keystore file
oracle.security.jps.pd.client.ssl.trustKeyStoreFileName	**Description:** Defines the name of the Trust Key Store file where Certificate Authority (CA) certificates are stored. Used for SSL communication between the Security Module and the Policy Distribution Component. **Mandatory** **Accepted Value:** the name of the identity key store file
oracle.security.jps.pd.client.ssl.identityKeyStoreKeyAlias	**Description:** Defines an Identity Key alias to identify the client certificate used for SSL communication between the Security Module and the Policy Distribution Component. **Optional** (if only one alias exists in the identity keystore there is no need to specify this value) **Accepted Value:** the identity key alias
oracle.security.jps.runtime.pd.client.SMinstanceType	**Description:** Defines the type of Security Module to which the Policy Distribution Component client is connecting. **Mandatory** **Accepted Value:** java (Other accepted values include wls, RMI and ws. Because this table covers the Java Security Module only, the value must be java.)

A.1.2.2 Policy Distribution Component Client Java Enterprise Edition Container Configuration (Controlled Push Mode)

Table A–3 compiles the parameters for the Policy Distribution Component client configuration when the Oracle Entitlements Server is running in a Java Enterprise Edition (JEE) environment and is configured to distribute data in the controlled-push mode.

Table A–3 Policy Distribution Client Configuration, JEE, Controlled Push Mode

Name	Information
oracle.security.jps.runtime.pd.client.policyDistributionMode	**Description:** Specifies the mode of policy distribution. *Controlled distribution* is initiated by the Policy Distribution Component, ensuring that the Security Module receives policy data that has been created or modified since the last distribution. **Mandatory** **Accepted Value:** controlled-push
oracle.security.jps.runtime.pd.client.sm_name	**Description:** Defines the name of the Security Module. **Mandatory** **Accepted Value:** Name of the Security Module

Table A–3 (Cont.) Policy Distribution Client Configuration, JEE, Controlled Push Mode

Name	Information
oracle.security.jps.runtime.pd.client.localpolicy.work_folder	**Description:** Defines the name of any directory in which local cache files are stored. This directory must have read and write privileges. **Optional** **Accepted Value:** The name of any directory in which local cache files will be stored. This directory must have read and write privileges.
oracle.security.jps.runtime.pd.client.incrementalDistribution	**Description:** Defines whether the distribution is incremental or flush. *Incremental distribution* is when new and modified data is distributed. *Flush distribution* is when the Policy Distribution Component notifies the Security Module to cleanup locally stored policies in preparation for a complete re-distribution of all policy objects in the policy store. **Optional** **Accepted Values:** ■ false (policy distribution is flush for this Security Module) ■ true (default value; policy distribution is incremental for this Security Module if the required change logs are kept in the policy store)
oracle.security.jps.runtime.pd.client.registrationRetryInterval	**Description:** When a Security Module starts, it registers itself with the Policy Distribution Component to ensure the local policy cache is up to date. If registration fails, it will retry each time this interval of time passes until successful. **Optional** **Accepted Value:** time in seconds (default value is 5)
oracle.security.jps.runtime.pd.client.waitDistributionTime	**Description:** If this value is defined and not equal to zero, it specifies the amount of time that a Security Module will wait for initial policy distribution to happen. During this wait period, authorization requests are blocked until either the initial policy distribution completes or the configured period expires. **Optional** **Accepted Value:** time in seconds (default value is 60)
oracle.security.jps.runtime.pd.client.RegistrationServerURL	**Description:** Defines the URL of the Oracle Entitlements Server Administration Server. Used by the Security Module to register itself with Oracle Entitlements Server when it starts. **Mandatory** **Accepted Value:** URL
oracle.security.jps.runtime.pd.client.backupRegistrationServerURL	**Description:** Defines a backup URL for the Oracle Entitlements Server Administration Server. Used by the Security Module to register itself with Oracle Entitlements Server when it starts if the primary URL (parameter above) is unavailable. **Optional** (although if not configured Oracle Entitlements Server failover will not work) **Accepted Value:** URL

Table A–3 (Cont.) Policy Distribution Client Configuration, JEE, Controlled Push Mode

Name	Information
oracle.security.jps.runtime.pd.client.SMinstanceType	**Description:** Defines the type of Security Module to which the Policy Distribution Component client is connecting. **Mandatory** **Accepted Values:** ▪ was ▪ wls
oracle.security.jps.runtime.pd.client.DistributionServiceURL	**Description:** Defines the URL to which the remote Policy Distributor will push policy updates. **Mandatory** **Accepted Values:** URL

A.1.2.3 Policy Distribution Client Configuration (Controlled Pull Mode)

Table A–4 compiles the parameters for the Policy Distribution Component client configuration when the Oracle Entitlements Server is running in either a JEE or a JSE environment and is configured to distribute data in the controlled-pull mode.

Table A–4 Policy Distribution Client Configuration, Controlled Pull Mode

Name	Information
oracle.security.jps.runtime.pd.client.policyDistributionMode	Specifies the mode of policy distribution. *Controlled distribution* is initiated by the Policy Distribution Component, ensuring that the Security Module receives policy data that has been created or modified since the last distribution. **Mandatory** **Accepted Value:** controlled-pull
oracle.security.jps.runtime.pd.client.sm_name	**Description:** Defines the name of the Security Module. **Mandatory** **Accepted Value:** the name of the Security Module
oracle.security.jps.runtime.pd.client.localpolicy.work_folder	**Description:** Defines the name of any directory in which local cache files are stored. This directory must have read and write privileges. **Optional** **Accepted Value:** The name of any directory in which local cache files will be stored. This directory must have read and write privileges.
oracle.security.jps.runtime.pd.client.incrementalDistribution	**Description:** Defines whether the distribution is incremental or flush. *Incremental distribution* is when new and modified data is distributed. *Flush distribution* is when the Policy Distribution Component notifies the Security Module to cleanup locally stored policies in preparation for a complete re-distribution of all policy objects in the policy store. **Optional** **Accepted Values:** ▪ false (policy distribution is flush for the Security Module) ▪ true (default value; policy distribution is incremental for this Security Module if the required change logs are kept in the policy store)

Table A–4 (Cont.) Policy Distribution Client Configuration, Controlled Pull Mode

Name	Information
oracle.security.jps.runtime.pd.client.waitDistributionTime	**Description:** If this value is defined and not equal to zero, it specifies the amount of time that a Security Module will wait for initial policy distribution to happen. During this wait period, authorization requests are blocked until either the initial policy distribution completes or the configured period expires. **Optional** **Accepted Value:** time in seconds (default value is 60)
oracle.security.jps.runtime.pd.client.PollingTimerEnabled	**Description:** Enables a periodic check for policy updates in the Policy Store. Can be set to false to disable polling for environment when policies are not expected to be modified. **Optional** **Accepted Values:** ■ false ■ true (default value)
oracle.security.jps.runtime.pd.client.PollingTimerInterval	**Description:** Defines the interval of time in which the Policy Distribution Component will check for policy data changes. **Optional** **Accepted Value:** time in seconds (default value of 600)
oracle.security.jps.ldap.root.name	**Description:** Defines the top (root) entry of the LDAP policy store directory information tree (DIT). **Mandatory** **Accepted Value:** the top (root) entry of the LDAP policy store directory information tree (DIT)
oracle.security.jps.farm.name	**Description:** Defines the RDN format of the domain node in the LDAP policy store. **Mandatory** **Accepted Value:** name of the domain
jdbc.url	**Description:** Takes a URL that points to the database. **Mandatory** (if using Java Database Connectivity API to connect to policy store) **Accepted Value:** URL
jdbc.driver	**Description:** Location of the driver if using Java Database Connectivity API to connect to an Apache Derby database. **Mandatory** **Accepted Value:** driver
datasource.jndi.name	**Description:** The JNDI name of the JDBC data source instance. The instance may correspond to a single source or multi-source datasource. Valid in only JEE applications. Applies only to database stores. **Mandatory** **Accepted Value:** name of JNDI data source; for example, `jdbc/APMDBDS`.
bootstrap.security.principal.key	**Description:** The key for the password credentials to access the policy store. Credentials are stored in the Credential Store Framework (CSF) store. **Mandatory** **Accepted Value:** CSF credential key

Table A–4 (Cont.) Policy Distribution Client Configuration, Controlled Pull Mode

Name	Information
bootstrap.security.principal.map	**Description:** The map for the password credentials to access the policy store. Credentials are stored in the CSF store. **Mandatory** **Accepted Value:** name of the CSF credential map

A.1.2.4 Policy Distribution Client Configuration (Non-controlled Mode)

Table A–5 compiles the parameters for Policy Distribution Component client configuration when the Oracle Entitlements Server is running in either a JEE or a JSE environment and is configured to distribute data in the non-controlled mode.

Table A–5 Policy Distribution Client COnfiguration, Non-controlled Mode

Name	Information
oracle.security.jps.runtime.pd.client.policyDistributionMode	**Description:** Specifies the mode of policy distribution. *Non-controlled distribution* is when the Security Module periodically retrieves policy data from a policy store (or from a component that serves as an intermediary between the two). **Optional** **Accepted Value:** non-controlled (default value)

A.2 Security Module Configuration

This section covers the configurations for the various types of Security Modules and their proxy clients.

- Section A.2.1, "Java Security Module"
- Section A.2.2, "Web Services Security Module"
- Section A.2.3, "RMI Security Module"
- Section A.2.4, "WebLogic Server Security Module"

A.2.1 Java Security Module

Table A–6 compiles the parameters to configure the Java Security Module embedded in either a JSE or a JEE container.

Table A–6 Java Security Module Configuration Parameters

Name	Information
oracle.security.jps.policystore.rolemember.cache.type	**Description:** Defines the role member cache type. Valid in J2EE and J2SE applications. Applies to LDAP and database stores. **Optional** **Accepted Values** ■ SOFTHASH (cleaning of a cache of this type relies on the garbage collector when there is a memory crunch) ■ WEAK (behavior of a cache of this type is similar to a cache of type SOFT but the garbage collector cleans it more frequently) ■ STATIC (default value; cache objects are statically cached and can be cleaned explicitly only according to the applied cache strategy, such as FIFO; the garbage collector does not clean a cache of this type)
oracle.security.jps.policystore.rolemember.cache.strategy	**Description:** Defines the type of strategy used in the role member cache. Valid in J2EE and J2SE applications. Applies to LDAP and database stores. **Optional** **Accepted Values** ■ NONE (all entries in the cache grow until a refresh or reboot occurs; there is no control over the size of the cache; not recommended but typically efficient when the policy footprint is very small) ■ FIFO (default value; the cache implements the first-in-first-out strategy)
oracle.security.jps.policystore.rolemember.cache.size	**Description:** Defines the number of roles kept in the role member cache. Valid in J2EE and J2SE application. Applies to LDAP and database stores. **Optional** **Accepted Value:** number (default value is 1000)
oracle.security.jps.policystore.rolemember.cache.warmup.enable	**Description:** Controls the way the Application Role membership cache is created. Valid in J2EE and J2SE applications. Applies to LDAP and database stores. **Optional** **Accepted Values** ■ true (the cache is created at server startup; use when the number of users and groups is significantly higher than the number of Application Roles) ■ false (default value; the cache is created on demand - lazy loading; use when the number of Application Roles is very high)
oracle.security.jps.policystore.policy.lazy.load.enable	**Description:** Enables or disables the policy lazy load. Valid in J2EE and J2SE applications. Applies to LDAP and database stores. **Optional** **Accepted Values** ■ false ■ true (default value)

Table A–6 (Cont.) Java Security Module Configuration Parameters

Name	Information
oracle.security.jps.policystore.policy.cache.strategy	**Description:** Defines the type of strategy used in the permission cache. Valid in J2EE and J2SE applications. Applies to LDAP and database stores. **Optional** **Accepted Values** ■ NONE (all entries in the cache grow until a refresh or reboot occurs; there is no control over the size of the cache; not recommended but typically efficient when the policy footprint is very small.) ■ PERMISSION_FIFO (default value; the cache implements the first-in-first-out strategy)
oracle.security.jps.policystore.policy.cache.size	**Description:** Defines the number of permissions kept in the permission cache. Valid in J2EE and J2SE applications. Applies to LDAP and database stores. **Optional** **Accepted Value:** number (default value is 1000)
oracle.security.jps.policystore.policy.cache.updateable	**Description:** Defines whether the policy cache is incrementally updated for management operations on policy data. **Optional** **Accepted Values** ■ false ■ true (default value)
oracle.security.jps.policystore.refresh.enable	**Description:** Enables or disables the policy store refresh. If this property is set, `oracle.security.jps.ldap.cache.enable` cannot be set. Valid in J2EE and J2SE applications. Applies to LDAP and database stores. **Optional** **Accepted Values:** ■ false ■ true (default value)
oracle.security.jps.policystore.refresh.purge.timeout	**Description:** Defines the time in milliseconds after which the policy store cache is purged. Valid in J2EE and J2SE applications. Applies to LDAP and database stores. **Optional** **Accepted Value:** time in milliseconds; default value is 43200000 which equals 12 hours
oracle.security.jps.ldap.policystore.refresh.purge.interval	**Description:** Defines the interval of time in which the policy store is polled for changes. Valid in J2EE and J2SE applications. Applies to LDAP and database stores. **Optional** **Accepted Value:** time in milliseconds; default value is 600000 which equals 10 minutes

Table A–6 (Cont.) Java Security Module Configuration Parameters

Name	Information
oracle.security.jps.pdp.missingAppPolicyQueryTTL	**Description:** Defines the interval of time to avoid frequently querying a non-exist Application (`ApplicationPolicy`) object. **Optional** **Accepted Value:** time to live in milliseconds (default value is 60000)
oracle.security.jps.pdp.AuthorizationDecisionCacheEnabled	**Description:** Specifies whether the authorization cache should be enabled. Valid in J2EE and J2SE applications. Applies to XML, LDAP, and database stores. **Optional** **Accepted Values** ■ false ■ true (default value)
oracle.security.jps.pdp.AuthorizationDecisionCacheEvictionCapacity	**Description:** Defines the maximum number of authorization and role mapping sessions to maintain. When the maximum is reached, old sessions are dropped and reestablished when needed. Valid in J2EE and J2SE applications. Applies to XML, LDAP, and database stores. **Optional** **Accepted Value:** number (default value is 500)
oracle.security.jps.pdp.AuthorizationDecisionCacheEvictionPercentage	**Description:** Defines the percentage of sessions to drop when the eviction capacity is reached. Valid in J2EE and J2SE applications. Applies to XML, LDAP, and database stores. **Optional** **Accepted Value:** number (default value is 10)
oracle.security.jps.pdp.AuthorizationDecisionCacheTTL	**Description:** Defines the number of seconds during which session data is cached. Valid in J2EE and J2SE applications. Applies to XML, LDAP, and database stores. **Optional** **Accepted Value:** time in seconds (default value is 60)
oracle.security.jps.pdp.anonymousrole.enable	**Description:** Specifies whether anonymous role has to be added to anonymous subject for policy matching. **Optional** **Accepted Values** ■ false ■ true (default value)
oracle.security.jps.pdp.authenticatedrole.enable	**Description:** Specifies whether authenticated role has to be added to authenticated subject for policy matching. **Optional** **Accepted Values** ■ false ■ true (default value)

A.2.2 Web Services Security Module

Table A–7 compiles the parameters to configure the Web Services Security Module embedded in either a JSE or a JEE container.

Table A–7 Web Services Security Module Configuration Parameters

Name	Information
oracle.security.jps.pdp.wssm.WSServiceRegistryPortNumber	**Description:** Defines the port on which the Web Services Security Module listens. **Mandatory** **Accepted Value:** port number
oracle.security.jps.pdp.wssm.WSServiceRegistryHost	**Description:** Defines the name of the server on which the Web Services Security Module is running. **Optional** **Accepted Value:** server name (default value is localhost)
oracle.security.jps.pdp.wssm.Protocol	**Description:** Defines the transport protocol used between the Policy Distribution Component client and server. **Optional** **Accepted Values** ■ https ■ http (default value)
oracle.security.jps.pdp.sm.IdentityMaxCacheSize	**Description:** Specifies the maximum number of users for which information is cached. When the maximum is reached, old records are dropped and reestablished when needed. **Optional** **Accepted Value:** number
oracle.security.jps.pdp.sm.IdentityCacheEvictionPercentage	**Description:** Specifies percentage of identities that must be evicted when cache has reached the maximum size. **Optional** **Accepted Value:** number indicating percentage
oracle.security.jps.pdp.sm.IdentityCachedEntryTTL	**Description:** Specifies time-to-live of an identity cache record. **Optional** **Accepted Value:** time in seconds
oracle.security.jps.pdp.wssm.responseContext	**Description:** Specifies whether to merge data from many AppContext responses into a single AppContext response. **Optional** **Accepted Values** ■ Merged ■ Unmerged (default value)
oracle.security.jps.pdp.wssm.ssl.identityKeyStoreFileName	**Description:** Defines the name of the Identity Key Store file where client certificates are stored for the Web Services Security Module. Used for SSL communications between the remote client and the Web Services Security Module. **Optional** **Accepted Value:** name of the Identity Key Store file
oracle.security.jps.pdp.wssm.ssl.trustKeyStoreFileName	**Description:** Defines the name of the Trust Key Store file in which CA certificates are stored. Used for SSL communications between the remote client and the Web Services Security Module. **Optional** **Accepted Value:** name of the Trust Key Store file

Table A–7 (Cont.) Web Services Security Module Configuration Parameters

Name	Information
oracle.security.jps.pdp.wssm.ssl.identityKeyStoreKeyAlias	**Description:** Specifies the Identity Key alias used to identify the Web Services Security Module client certificate used for SSL communication between the Web Services Security Module and the remote client. Acepted value: Idenity key alias **Optional** **Accepted Value:** Identity Key alias

A.2.3 RMI Security Module

Table A–8 compiles the parameters to configure the RMI Security Module embedded in either a JSE or a JEE container.

Note: Currently this configuration is for a standalone deployment. We need to add the Container based configuration later.

Table A–8 RMI Security Module Configuration Parameters

Name	Information
oracle.security.jps.pdp.rmism.RMIRegistryPortNumber	**Description:** Defines the port on which the RMI Security Module listens to the RMI server. **Mandatory** **Accepted Value:** port number.
oracle.security.jps.pdp.rmism.UseSSL	**Description:** Defines whether the SSL protocol is used for secure communication between the RMI Security Module and RMI server. **Optional** **Accepted Values** ■ true ■ false (default)
oracle.security.jps.pdp.sm.IdentityMaxCacheSize	**Description:** Specifies the maximum number of users for which information is cached. When the maximum is reached, old records are dropped and reestablished when needed. **Optional** **Accepted Value:** number
oracle.security.jps.pdp.sm.IdentityCacheEvictionPercentage	**Description:** Specifies percentage of identities that must be evicted when cache has reached the maximum size. **Optional** **Accepted Value:** number representing percentage
oracle.security.jps.pdp.sm.IdentityCachedEntryTTL	**Description:** Specifies the time-to-live of an identity cache record. **Optional** **Accepted Value:** time in seconds

A.2.4 WebLogic Server Security Module

Table A–9 compiles the parameters to configure the WebLogic Server (WLS) Security Module embedded in a JEE container. These parameters are used only when the WLS Security Module is configured to be used as a PEP.

- See Section 1.3.2, "The Policy Decision Point and the Policy Enforcement Point" for contextual information.
- See Section 11.1, "Integrating with WebLogic Server" to enable the WebLogic Server Security Module.

Table A–9 WebLogic Server Security Module Configuration Parameters

Name	Information
oracle.security.jps.pdp.wlssm.UndefinedApplicationEffect	**Description:** Specifies the effect that the provider has to return if an application is not defined in the policy store. **Optional** **Accepted Values** ▪ permit ▪ abstain ▪ deny
oracle.security.jps.pdp.wlssm.NoApplicablePolicyEffect	**Description:** Specifies the effect that the provider has to return if no applicable policies have been found. **Optional** **Accepted Values** ▪ permit (represents an open system) ▪ abstain ▪ deny (represents a closed system)

A.3 PDP Proxy Configuration

This section contains information regarding configuration for the Security Module proxies.

- Section A.3.1, "Web Services Security Module Proxy Client"
- Section A.3.2, "RMI Security Module Proxy Client"

A.3.1 Web Services Security Module Proxy Client

Table A–10 compiles the parameters to configure the Web Services Security Module proxy client.

Table A–10 Web Services Proxy Client Configuration Parameters

Name	Information
oracle.security.jps.pdp.PDP Transport	**Description:** Specifies the underlying protocol to be used by Multi-protocol Security Module to communicate with Oracle Entitlements Server. **Mandatory** **Accepted Values**: no default value; XACML is always available in the Web Services Security Module. ■ WS ■ RMI
oracle.security.jps.pdp.proxy.PDPAddress	**Description:** Specifies the host and port number of either the Web Services Security Module. For example, `http://dadvml0134:9015` **Optional** **Accepted Value:** a comma separated list of URIs (if more then one address is specified the first is conidered the primary, and the rest as backups)
oracle.security.jps.pdp.proxy.RequestTimeoutMilliSecs	**Description:** Defines the interval of time in which an authorization request times out when the remote PDP (RMI or Web Services Security Module) is not responding. **Optional** **Accepted Value:** time in milliseconds (default value is 10000)
oracle.security.jps.pdp.proxyFailureRetryCount	**Description:** Specifies the number of attempts to make before attempting the alternate failover server. **Optional** **Accepted Value:** number (default value is 3)
oracle.security.jps.pdp.proxy.FailbackTimeoutMilliSecs	**Description:** Specifies the interval of time after which a failed primary server is tried again for failover. **Optional** **Accepted Value:** time in milliseconds (default value is 180000)
oracle.security.jps.pdp.proxy.SynchronizationIntervalMilliSecs	**Description:** Defines how often the PDP Proxy polls the PDP server in order to synchronize its state. For example, the interval is used to periodically check whether the authorization cache has to be flushed. **Optional** **Accepted Value:** time in seconds (default value is 60)
oracle.security.jps.pdp.proxy.wssm.ssl.identityKeyStoreFileName	**Description:** Defines the name of the Identity Key Store file where client certificates for the Web Services Security Module are stored. Used for SSL communication between a client and the Web Services Security Module. **Optional** **Accepted Value:** name of the Identity Key Store file
oracle.security.jps.pdp.proxy.wssm.ssl.trustKeyStoreFileName	**Description:** Defines the name of the Trust Key Store file where CA certificates for Web Services Security Module are stored. Used for SSL communication between a client and the Web Services Security Module. **Optional** **Accepted Value:** the name of the Trust Key Store file.

Table A–10 (Cont.) Web Services Proxy Client Configuration Parameters

Name	Information
oracle.security.jps.pdp.proxy.wssm.ssl.identityKeyStoreKeyAlias	**Description:** Specifies the alias name of the Web Services client certificate. Used for SSL communication between a client and the Web Services Security Module. **Optional** **Accepted Value:** alias of the identity key store (if only one alias exists in the identity key store, no need to specify this value)
oracle.security.jps.pdp.proxy.wssm.protocol	**Description:** Defines the transport protocol used between the Policy Distribution Component client and server. **Optional** **Accepted Values** ▪ https ▪ http (default value)

A.3.2 RMI Security Module Proxy Client

Table A–11 compiles the parameters to configure the RMI Security Module Proxy Client.

Table A–11 PDP RMI Proxy Client Configuration Parameters

Name	Information
oracle.security.jps.pdp.PDPTransport	**Description:** Specifies the underlying protocol to be used by Multi-protocol Security Module to communicate with Oracle Entitlements Server. **Mandatory** **Accepted Values:** no default value; XACML is always available in the RMI Security Module. ▪ WS ▪ RMI
oracle.security.jps.pdp.proxy.PDPAddress	**Description:** Specifies the host and port number of the RMI Security Module. For example, `rmi://localhost:9400` **Mandatory** **Accepted Value:** a comma separated list of URIs (if more then one address is specified the first is conidered the primary, and the rest as backups)
oracle.security.jps.pdp.proxy.RequestTimeoutMilliSecs	**Description:** Defines the interval of time in which an authorization request times out when the remote PDP (RMI or Web Services Security Module) is not responding. **Optional** **Accepted Value:** time in milliseconds (default value is 10000)
oracle.security.jps.pdp.proxyFailureRetryCount	**Description:** Specifies the number of attempts to make before attempting the alternate failover server. **Optional** **Accepted Value:** number (default value is 3)
oracle.security.jps.pdp.proxy.FailbackTimeoutMilliSecs	**Description:** Specifies the interval of time after which a failed primary server is tried again for failover. **Optional** **Accepted Value:** time in milliseconds (default value is 180000)

Table A–11 (Cont.) PDP RMI Proxy Client Configuration Parameters

Name	Information
oracle.security.jps.pdp.proxy.SynchronizationIntervalMilliSecs	**Description:** Defines how often the PDP Proxy polls the PDP server in order to synchronize its state. For example, the interval is used to periodically check whether the authorization cache has to be flushed. **Optional** **Accepted Value:** time in seconds (default value is 60)

A.4 Policy Store Service Configuration

Table A–12 compiles the configuration parameters for the Policy Store Service.

Table A–12 Policy Store Service Configuration Parameters

Name	Information
ldap.url	**Description:** Defines the URL of the LDAP policy store. Valid in JEE and JSE applications and only applies to LDAP stores. **Mandatory** **Accepted Value:** URI of the LDAP policy store in the format `ldap://host:port`.
max.search.filter.length	**Description:** Defines the maximum length of a search filter. **Mandatory** **Accepted Value:** integer defining the maximum length of a search filter; for example, 1024
oracle.security.jps.ldap.root.name	**Description:** Defines the RDN format of the root node in the LDAP policy store. Valid in JEE and JSE applications. Applies to LDAP and database stores. **Mandatory** **Accepted Value:** root name of jps context; for example, `cn=jpsroot`.
oracle.security.jps.farm.name	**Description:** Defines the RDN format of the root node in the LDAP policy store. Valid in JEE and JSE applications. Applies to LDAP and database stores. **Mandatory** **Accepted Value:** farm name of the domain; for example, `cn=base_domain`.

Table A–12 (Cont.) Policy Store Service Configuration Parameters

Name	Information
oracle.security.jps.policystore.resourcetypeenforcement mode	**Description:** Controls the throwing of exceptions if any of the following checks fail: ▪ Verify that if two resource types share the same permission class, that permission must be either ResourcePermission or extend AbstractTypedPermission, and this last resource type cannot be created. ▪ Verify that all permissions have resource types defined, and that the resource matcher permission class and the permission being granted match. Valid in JEE and JSE applications. Applies to LDAP and database stores. **Optional** **Accepted Values** ▪ strict (when any of the above checks fail, the system throws an exception and the operation is aborted) ▪ lenient (default value; when any of the above checks fail, the system does not throw any exceptions, the operation continues without disruption, and any discrepancies encountered are logged)
bootstrap.security.principal.key	**Description:** Defines the key for the password credentials to access the LDAP policy store, stored in the CSF store. Valid in JEE and JSE applications. Applies to LDAP and database stores. **Mandatory** **Accepted Value:** the key name of the credential; for example, `oes_sm_key`. The out-of-the-box value is `bootstrap`.
bootstrap.security.principal.map	**Description:** Defines the map for the password credentials to access the LDAP policy store, stored in the CSF store. Valid in JEE and JSE applications. Applies to LDAP and database stores. **Mandatory** **Accepted Value:** map name of the credential; for example, `oes_sm_map`. The default value is `BOOTSTRAP_JPS`.
jdbc.driver	**Description:** Defines the name of the JDBC driver. **Mandatory** **Accepted Value:** name of the JDBC driver.
datasource.jndi.name	**Description:** The JNDI name of the JDBC data source instance. The instance may correspond to a single source or multi-source datasource. Valid in only JEE applications. Applies only to database stores. **Mandatory** **Accepted Value:** name of JNDI data source; for example, `jdbc/APMDBDS`.
jdbc.url	**Description:** Defines the JDBC driver connection URL. **Mandatory** **Accepted Value:** the JDBC driver connection URL.

Table A–12 (Cont.) Policy Store Service Configuration Parameters

Name	Information
oracle.security.jps.pd.localMode	**Description:** Defines whether the policy store is running in local mode. **Mandatory** **Accepted Values** ■ true ■ false

Index

A

C

D

Database to XML, 11-10
LDAP to XML, 11-6

R

S

U

W

X