



BEA WebLogic Enterprise Security™

Administration Application Guide

Copyright

Copyright © 2005 BEA Systems, Inc. All Rights Reserved.

Restricted Rights Legend

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

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

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

Trademarks or Service Marks

BEA, BEA JRockit, BEA Liquid Data for WebLogic, BEA WebLogic Server, Built on BEA, Jolt, JoltBeans, SteelThread, Top End, Tuxedo, and WebLogic are registered trademarks of BEA Systems, Inc. BEA AquaLogic, BEA AquaLogic Data Services Platform, BEA AquaLogic Enterprise Security, BEA AquaLogic Service Bus, BEA AquaLogic Service Registry, BEA Builder, BEA Campaign Manager for WebLogic, BEA eLink, BEA Manager, BEA MessageQ, BEA WebLogic Commerce Server, BEA WebLogic Enterprise, BEA WebLogic Enterprise Platform, BEA WebLogic Enterprise Security, BEA WebLogic Express, BEA WebLogic Integration, BEA WebLogic Java Adapter for Mainframe, BEA WebLogic JDriver, BEA WebLogic JRockit, BEA WebLogic Log Central, BEA WebLogic Personalization Server, BEA WebLogic Platform, BEA WebLogic Portal, BEA WebLogic Server Process Edition, BEA WebLogic WorkGroup Edition, BEA WebLogic Workshop, and Liquid Computing are trademarks of BEA Systems, Inc. BEA Mission Critical Support is a service mark of BEA Systems, Inc. All other company and product names may be the subject of intellectual property rights reserved by third parties.

All other trademarks are the property of their respective companies.

Contents

1. Introduction to System Administration

About This Document	1-1
Audience	1-2
Product Documentation on the dev2dev Web Site	1-3
Overview of System Administration	1-4
Distributed Computing Security Infrastructure	1-5
Attributes	1-10
Security Protections for System Administration Tools	1-10
Administration Console	1-10

2. Failover and System Reliability

Understanding Failover	2-1
WebLogic Enterprise Security Failover Considerations	2-2
Failover Considerations for the Administration Server	2-3
Failover Considerations for the Database Server	2-4
Failover Considerations for a Security Service Module	2-5
Failover Considerations for a Service Control Manager	2-6
Understanding Database Replication	2-6
Database Replication in an Oracle Environment	2-7
Preparing for Oracle Database Replication	2-7
Master and Materialized View Site Requirements	2-8
Master Site Requirements	2-10

Materialized View Site Requirements	2-10
Requirements for the Machine Running the Replication Setup Scripts	2-11
Setting the Required Replication Setup Parameters	2-11
Setting Up Oracle Database Replication	2-12
Using Scripts to Set Up Oracle Database Replication	2-13
Setting Up Oracle Database Replication Manually	2-14
Using Scripts to Clean Up Oracle Database Replication	2-14
Cleaning Up the Oracle Database Replication Manually	2-15
Miscellaneous Oracle Database Replication Tasks	2-15
Database Replication in a Sybase Environment	2-16
Preparing for Sybase Database Replication	2-16
Privileges for the Primary and the Replicate ASE Servers	2-17
Primary ASE Server and Primary Database Requirements	2-17
Replicate ASE Server and Replicate Database Requirements	2-18
Requirements for the Machine Used to Run Sybase Database Replication Setup Scripts	2-19
Parameters Needed for Sybase Database Replication Setup	2-19
Setting Up Sybase Database Replication	2-21
Using Scripts to Set Up Sybase Database Replication	2-21
Setting Up Sybase Database Replication Manually	2-22
Setting up the Primary ASE Server and the Primary Database	2-23
Starting the ASE Replicator	2-23
Adding a Remote Server in Primary ASE Server for the Replicate ASE Server	2-24
Setting Up the Replicate ASE Server and the Primary Database	2-24
Setting up the Sybase Database Replication Process	2-25
Cleaning Up Sybase Database Replication	2-25
Using Scripts to Clean Up Sybase Database Replication	2-26
Cleaning Up the Sybase Database Replication Manually	2-27

Cleaning Up the Sybase Database Replication Process	2-27
Cleaning Up the Replicate ASE Server and Primary Database	2-28
Removing the Remote Server	2-28
Stopping ASE Replicator	2-28
Cleaning Up the Primary ASE Server and the Primary Sybase Database	2-29
Completing Sybase Database Replication Cleanup	2-29

3. Administration Policy

Security Roles	3-1
Dynamic Role Mapping.	3-2
Understanding the Administration Policy	3-2
Admin Role	3-3
Deployer Role.	3-5
Operator Role	3-5
Monitor Role	3-6
Everyone Role	3-6
Anonymous Role	3-6
Resources	3-6
Privileges	3-8
Context Attributes	3-10
Evaluation Functions	3-14
Authorization Queries	3-14
Enumerated Types	3-23
Default Admin Policy	3-23
Example Policy Customizations	3-26

4. Security Administration

Managing Security.	4-1
----------------------------	-----

Security Configuration	4-2
Resources	4-4
Resource Attribute	4-6
Privilege	4-7
Privilege Group	4-8
Identity	4-9
User	4-10
Group	4-11
Identity Attribute	4-12
Role	4-12
Role Policy	4-12
Policy	4-13
Policy Rule	4-13
Policy Inquiry	4-14
Policy Verification	4-14
Declarations	4-15
Deployment	4-15
What's Next?	4-16

5. Using the Console

Overview	5-1
Checking the Console Version Number	5-2
Setting Console Preferences	5-2
Starting the Administration Console	5-3
Logging out of the Administration Console	5-4
Using the Administration Console	5-5
Getting Help	5-8
Configuring the Administration Server for Failover	5-8

Additional BEA Documentation Available on the Internet	5-9
--	-----

6. Starting and Stopping Services

Starting and Stopping Administration Server Processes On Windows	6-1
Starting and Stopping Administration Server Processes on Unix	6-3
Starting and Stopping Security Service Module Processes	6-5
Starting and Stopping Processes on Windows	6-5
Starting and Stopping Processes on UNIX	6-7
Start-Up Option on Linux Platforms	6-8

7. Configuring Secure Sockets Layer for a Production Environment

Some SSL Basics	7-1
Private Keys, Digital Certificates, and Trusted Certificate Authorities	7-2
One-Way SSL Versus Two-Way SSL	7-3
How WebLogic Enterprise Security Locates Trust	7-3
Configuring SSL	7-3
Obtaining Private Keys, Digital Certificates, and Trusted Certificate Authorities . .	7-4
Creating a Keystore and Loading Private Keys and Trusted Certificate Authorities .	7-4
Common Keytool Commands	7-6
Using the ImportPrivateKey Utility	7-7
Configuring Keystores	7-8
Configuring One-Way SSL	7-10
Configuring Two-Way SSL	7-11
SSL Certificate Validation	7-12
Setting the Level of Certificate Validation	7-13
Checking Certificate Chains	7-15
Troubleshooting Problems with Certificates	7-16

Specifying the Version of the SSL Protocol	7-16
--	------

8. Enabling Single Sign On

Configuring Single Sign On with Microsoft Clients.	8-1
Requirements	8-1
Enabling a Web Service or Web Application.	8-2
Configuring the SPNEGO Provider	8-3
Editing the Descriptor File	8-3
Configuring the Active Directory Authentication.	8-4
Configure the Active Directory Authentication Provider	8-6
Configure the Client .NET Web Service	8-6
Configure the Internet Explorer Client Browser	8-7
Configure the Sites	8-7
Configure Intranet Authentication	8-8
Verify the Proxy Settings	8-8
Set the Internet Explorer 6.0 Configuration Settings	8-8

9. Security Configuration

Overview	9-1
Security Configuration	9-1
Understanding the Service Control Manager.	9-2
Configuring a Service Control Manager	9-2
Understanding the Security Service Module	9-3
Configuring a Security Service Module.	9-4
Binding a Security Service Module to a Service Control Manager.	9-8
Unbinding a Security Service Module from a Service Control Manager	9-8
Configuring Security Providers	9-9
Configuring an Authentication Provider	9-10

Changing the Order of Authentication Providers	9-11
Setting the JAAS Control Flag	9-12
Configuring an Open LDAP Authentication Provider	9-13
Configuring a Windows NT Authentication Provider	9-14
Configuring an Active Directory Authentication Provider	9-15
Configuring an iPlanet LDAP Authentication Provider	9-16
Configuring Failover for LDAP Authentication Providers	9-17
Configuring a Novell LDAP Authentication Provider	9-20
Configuring Failover for the Database Authentication Provider	9-21
Configuring a Database Authentication Provider	9-23
Oracle Database Configuration	9-24
Sybase Database Configuration	9-25
Specifying SQL Query Strings and Provider Extensions	9-25
Configuring an ALES Identity Assertion Provider	9-28
Configuring a SAML Identity Assertion Provider	9-29
Configuring a Single Pass Negotiate Identity Asserter	9-30
Configuring an X.509 Identity Assertion Provider	9-31
Configuring an ALES Credential Mapping Provider	9-32
Configuring a Database Credential Mapper	9-33
Configuring Failover for the Database Credential Mapper Provider	9-36
Configuring a SAML Credential Mapping Provider	9-37
Configuring an ASI Authorization Provider	9-38
Using the asipasswd Utility to Configure the Metadirectory Password	9-39
Configuring an ASI Adjudication Provider	9-40
Configuring an ASI Role Mapping Provider	9-41
Configuring a Resource Deployment Audit Provider	9-43
Configuring a Log4j Audit Channel Provider	9-44
Configuring a Custom Security Provider	9-45

Deleting a Security Provider	9-46
Configuring a WebLogic Server Security Service Module	9-47
Configuring the WebLogic Security Providers	9-47
Configuring the WebLogic Authentication Provider	9-49
Configuring the WebLogic Authorization Provider	9-50
Configuring a WebLogic Role Mapping Provider	9-50
Configuring the WebLogic Credential Mapping Provider	9-51

10.Performance and Caching

Authorization Caching	10-1
Configuring Authorization Caching.	10-1
Authorization Caching Expiration Functions	10-3

11.Deployment

Deployment	11-1
Understanding Deployment	11-1
Managing Deployment	11-2
Distributing Policy	11-2
Distributing Configuration	11-4
Distributing Structural Changes	11-5
Viewing Distribution Results	11-5
Viewing Deployment Status	11-6

12.Provider Extensions

What is a Provider Extension?	12-1
Authorization and Role Mapping Extensions	12-2
Using Java-Based Plug-ins	12-2
Using the Java-based Plug-in Interfaces	12-3
Resource Converter	12-4

Attribute Retriever	12-5
Attribute Converter.	12-6
Using Language Extensions	12-7
Building an Extension.	12-8
Deploying the Extension.	12-8
Using the Extension	12-9
Custom Audit Plug-ins	12-9
Using the Custom Audit Plug-in	12-9
Audit Plug-in Renderer Class.	12-10
Database Authentication Plug-in.	12-10

13.Audit Events

What is an AuditEvent?	13-1
What Events are Audited?	13-3
Custom Audit Context Extensions	13-4
Audit Event Interfaces and Audit Events	13-5
AuditAtnEvent	13-5
AuditAtzEvent	13-6
AuditCredentialMappingEvent.	13-7
AuditMgmtEvent	13-7
AuditPolicyEvent	13-7
AuditRoleDeploymentEvent	13-8
AuditRoleEvent	13-9
Admin Policy Audit Events	13-9
Additional Audit Event Interfaces.	13-16
Authentication - AuditAtnEvent	13-17
Policy Deployment - AuditPolicyDeployEvent	13-18
Policy Undeployment - AuditPolicyUndeployEvent	13-18

Policy Events - AuditPolicyEvent	13-19
Role Mapping - AuditRoleEvent	13-19
Role Deployment - AuditRoleDeployEvent.	13-20
Role Undeployment - AuditRoleUndeployEvent.	13-20
Predicate Events - AuditPredicateEvent.	13-20
ContextHandler Object	13-20
PolicyAdministrationEvent.	13-21
Using Custom Audit Providers	13-21

14.Function Reference

Function Pointers.	14-1
*CredFunc() - Custom Credential Function Pointer	14-2
Description	14-2
Syntax	14-2
Parameters.	14-2
Returns	14-2
Example	14-2
See Also	14-2
*EvalFunc() - Custom Evaluation Function Pointer	14-3
Syntax	14-3
Parameters.	14-3
Returns	14-3
Example	14-3
See Also	14-4
*ShutdownFunc () - Custom Shutdown Function Pointer.	14-4
Syntax	14-4
Parameters.	14-4
Returns	14-4

Example	14-4
See Also	14-4
*PluginInitFunc() - Plug-in Initialization Function Pointer	14-5
Syntax	14-5
Parameters	14-5
Returns	14-5
Example	14-5
registerCustomCredentialFunction() - Register Credential Function	14-6
Syntax	14-6
Parameters	14-6
Returns	14-6
Example	14-6
See Also	14-6
registerCustomEvaluationFunction() - Register Evaluation Function	14-7
Syntax	14-7
Parameters	14-7
Returns	14-7
Example	14-7
See Also	14-7
registerShutdownFunction() - Register Shutdown Function	14-8
Syntax	14-8
Parameters	14-8
Returns	14-8
Example	14-8
See Also	14-8
Session Class	14-9
Session::SetAttribute() - Append AttributeValue Object	14-10
Syntax	14-10

Parameters	14-10
Returns	14-10
Example	14-10
See Also	14-11
Session::getAttribute() - Get AttributeValue Object from Attribute	14-12
Syntax	14-12
Parameters	14-12
Returns	14-12
Example	14-12
See Also	14-12
Session::getEvalResult() - Get Evaluation Result	14-13
Syntax	14-13
Parameters	14-13
Returns	14-13
Example	14-13
See Also	14-13
Session::appendReturnData() - Return Evaluation Results	14-14
Syntax	14-14
Parameters	14-14
Returns	14-15
Example	14-15
See Also	14-15
Session::getDomainName() - Get Domain Name for the Session	14-16
Syntax	14-16
Parameters	14-16
Returns	14-16
Example	14-16
See Also	14-16

Session::getLocationName() - Get Location Name for Session	14-17
Syntax	14-17
Parameters	14-17
Returns	14-17
Example	14-17
See Also	14-17
Session::getApplicationName() - Get Application Name for Session	14-18
Syntax	14-18
Parameters	14-18
Returns	14-18
Example	14-18
See Also	14-18
Session::getUserID() - Get User Name for Session	14-19
Syntax	14-19
Parameters	14-19
Returns	14-19
Example	14-19
See Also	14-19
AttributeValue Class	14-20
Single Value	14-20
Lists of Values	14-21
Methods Common to Both Types	14-21
Internal Methods	14-21
AttributeValue::addValue() - Add and Set a String List Attribute Value	14-22
Syntax	14-22
Parameters	14-22
Returns	14-22
Example	14-22

See Also	14-22
AttributeValue::AttributeValue() - Constructor	14-23
Syntax	14-23
Parameters	14-23
Returns	14-23
Example	14-23
See Also	14-24
AttributeValue::entries() - Count Number of List Elements	14-25
Syntax	14-25
Parameters	14-25
Returns	14-25
Example	14-25
See Also	14-25
AttributeValue::getValue() - Get Single Attribute Value	14-26
Syntax	14-26
Parameters	14-26
Returns	14-26
Example	14-26
See Also	14-26
AttributeValue::has() - Check If Value is Already Present in a List	14-27
Syntax	14-27
Parameters	14-27
Returns	14-27
Example	14-27
See Also	14-27
AttributeValue::IsList() - Is Attribute Value an Indexed List?	14-28
Syntax	14-28
Parameters	14-28

Returns	14-28
Example.....	14-28
See Also.....	14-28
AttributeValue::IsSingle() - Is Attribute Value a Single Value?	14-29
Syntax	14-29
Parameters.....	14-29
Returns	14-29
Example.....	14-29
See Also.....	14-29
AttributeValue::isUndefined() - Is Attribute Value an undefined object?	14-30
Syntax	14-30
Parameters.....	14-30
Returns	14-30
Example.....	14-30
See Also.....	14-30
AttributeValue::setValue() - Set Single Attribute Value.....	14-31
Syntax	14-31
Parameters.....	14-31
Returns	14-31
Example.....	14-31
See Also.....	14-31
AttributeValue::removeAt() - Remove Indexed List Attribute Value	14-32
Syntax	14-32
Parameters.....	14-32
Returns	14-32
Example.....	14-32
See Also.....	14-32
AttributeValue::removeValue() - Remove Named List Attribute Value.....	14-33

Syntax	14-33
Parameters	14-33
Returns	14-33
Example	14-33
See Also	14-33
AttributeValue::size() - Count Number of List Elements	14-34
Syntax	14-34
Parameters	14-34
Returns	14-34
Example	14-34
See Also	14-34
AttributeValue [] Operator - Returns the Value of an Indexed String List Element	14-35

Introduction to System Administration

This section provides an introduction to system administration tasks and discusses the various tools available for configuring and managing your enterprise using WebLogic® Enterprise Security. If you are not familiar with the architecture and services provided, please see the [Introduction to WebLogic Enterprise Security](#). This document provides a starting point for System Administrators who are using the WebLogic Enterprise Security Administration Application.

This section covers the following topics:

- [“About This Document”](#) on page 1-1
- [“Overview of System Administration”](#) on page 1-4
- [“Distributed Computing Security Infrastructure”](#) on page 1-5
- [“Administration Console”](#) on page 1-10

About This Document

This document describes how to use the Administration Application to configure and deploy security service modules. It is organized as follows:

- [Chapter 1, “Introduction to System Administration,”](#) provides an overview, conceptual, and architectural information for the WebLogic Enterprise Security products.
- [Chapter 2, “Failover and System Reliability,”](#) describes WebLogic Enterprise Security features that support recovery from failure.

- [Chapter 3, “Administration Policy,”](#) describes the default policy supplied with the product.
- [Chapter 4, “Security Administration,”](#) describes basic concepts used in WebLogic Enterprise Security Console and how to design and implement your security policy.
- [Chapter 5, “Using the Console,”](#) describes how to use the Administration Console to the Administration Console, how to log on and off, and how to configure failover.
- [Chapter 6, “Starting and Stopping Services,”](#) describes the fundamentals of the BEA WebLogic Enterprise Security services and how to start and stop those services.
- [Chapter 7, “Configuring Secure Sockets Layer for a Production Environment,”](#) describes how to configure SSL to use digital certificates suitable for a production environment rather than the demonstration certificates configured by default in the WebLogic Enterprise Security Administration Application.
- [Chapter 8, “Enabling Single Sign On,”](#) describes the setup steps necessary to achieve Single Sign-On (SSO) integration with .NET based web services clients, as well as Internet Explorer browser clients.
- [Chapter 9, “Security Configuration,”](#) describes how to use the providers to implement your security policies and how to configure custom providers using the Administration Console.
- [Chapter 10, “Performance and Caching,”](#) discusses issues of performance when implementing authorization policies and how to configure your providers for the improved performance.
- [Chapter 11, “Deployment,”](#) describes how to distribute configuration and policy data to Security Service Modules and Service Control Managers.
- [Chapter 12, “Provider Extensions,”](#) describes advanced concepts of how to use plug-in function that you write to extend the capabilities of the existing providers.
- [Chapter 13, “Audit Events,”](#) describes the audit events you can use through the Audit plug-in and Log4j Audit Channel provider.
- [Chapter 14, “Function Reference,”](#) describes the C++ API that you can use to write an Authorization plug-in.

Audience

This administration guide is written for Administrators who are implementing and maintaining security configurations, authentication and authorization schemes, and setting up and maintaining access to deployed application resources. Application Administrators have a general knowledge of security concepts and the Java security architecture. They understand Java, XML, deployment descriptors, and can identify security events in server and audit logs.

The Administrator configures all security providers, implementing authentication, authorization, role mapping, auditing and failover policies. The Administrator may or may not be responsible for installation and configuration of the database component, but must be familiar with the enrollment process.

Product Documentation on the dev2dev Web Site

BEA product documentation, along with other information about BEA software, is available from the BEA dev2dev web site:

<http://dev2dev.bea.com>

To view the documentation for a particular product, select that product from the Product Centers menu on the left side of the screen on the dev2dev page. Select More Product Centers. From the BEA Products list, choose WebLogic Enterprise Security 4.2. The home page for this product is displayed. From the Resources menu, choose Documentation 4.2. The home page for the complete documentation set for the product and release you have selected is displayed.

Related Information

The BEA corporate web site provides all documentation for BEA WebLogic Enterprise Security. Other BEA WebLogic Enterprise Security documents that may be of interest to the reader include:

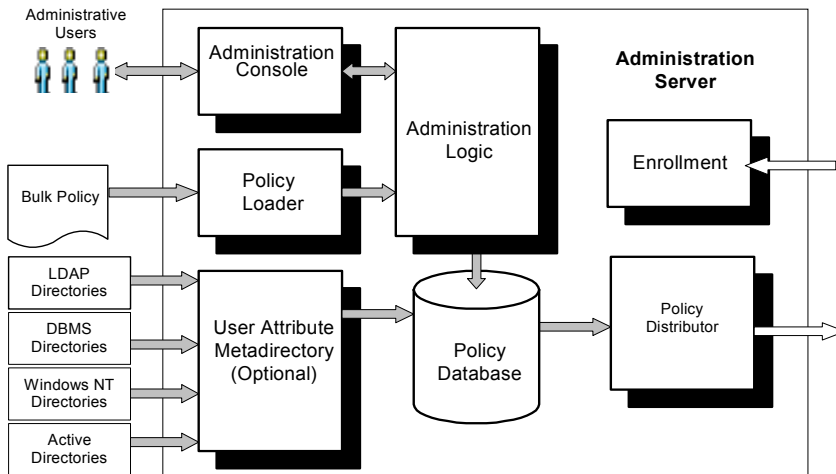
- [*Introduction to WebLogic Enterprise Security*](#)—This document summarizes the features of the BEA WebLogic® Enterprise Security products and presents an overview of the architecture and capabilities of the security services. It provides a starting point for understanding the family of BEA WebLogic Enterprise Security products.
- [*Programming Security for Java Applications*](#)—This document describes how to implement security in Java applications. It includes descriptions of the Security Service Application Programming Interfaces and programming instructions.
- [*Developing Security Providers for BEA WebLogic Enterprise Security*](#)—This document provides security vendors and security and application developers with the information needed to develop custom security providers.
- [*BEA WebLogic Enterprise Security Policy Managers Guide*](#)—This document defines the policy model used by BEA WebLogic Enterprise Security, and describes how to import and export policy data.
- [*Javadocs for Java API*](#)—This document provides reference documentation for the Java Application Programming Interfaces that are provided with and supported by this release of BEA WebLogic Enterprise Security.
- [*Javadocs for Security Service Provider Interfaces*](#)—This document provides reference documentation for the Security Service Provider Interfaces that are provided with and supported by this release of BEA WebLogic Enterprise Security.

Overview of System Administration

You manage a WebLogic Enterprise Security environment by using any of several system administration tools provided with the product. A WebLogic Enterprise Security environment can consist of a single Administration Application instance or multiple instances, each hosted on one or more physical machines; one or more Service Control Managers hosted on individual machines, with any number of Security Service Modules associated with each one. The system administration tools include the Administration Console, the Policy Import and Export tools, a Policy Distributor, Policy Database, and an API, with which you manage security, database connections, messaging, transaction processing, and the runtime configuration of your applications. You may also want to configure a meta-directory to manage users.

The basic administrative unit for a WebLogic Enterprise Security installation is called an enterprise domain. An enterprise domain is a logically related group of Security Service Modules from which the Administration Server manages resources as a unit. An enterprise domain always includes at least one Administration Server instance, one Service Control Manager, and one Security Service Module. The Administration Server serves as a central point of contact for instances and system administration tools.

Figure 1-1 Administration Server Architecture



You can configure multiple servers to be part of a cluster to support failover. A cluster is a group of server instances that work together to provide scalability and high-availability for applications. For additional information on configuring WebLogic Enterprise Security for failover, see [“Failover and System Reliability” on page 2-1](#).

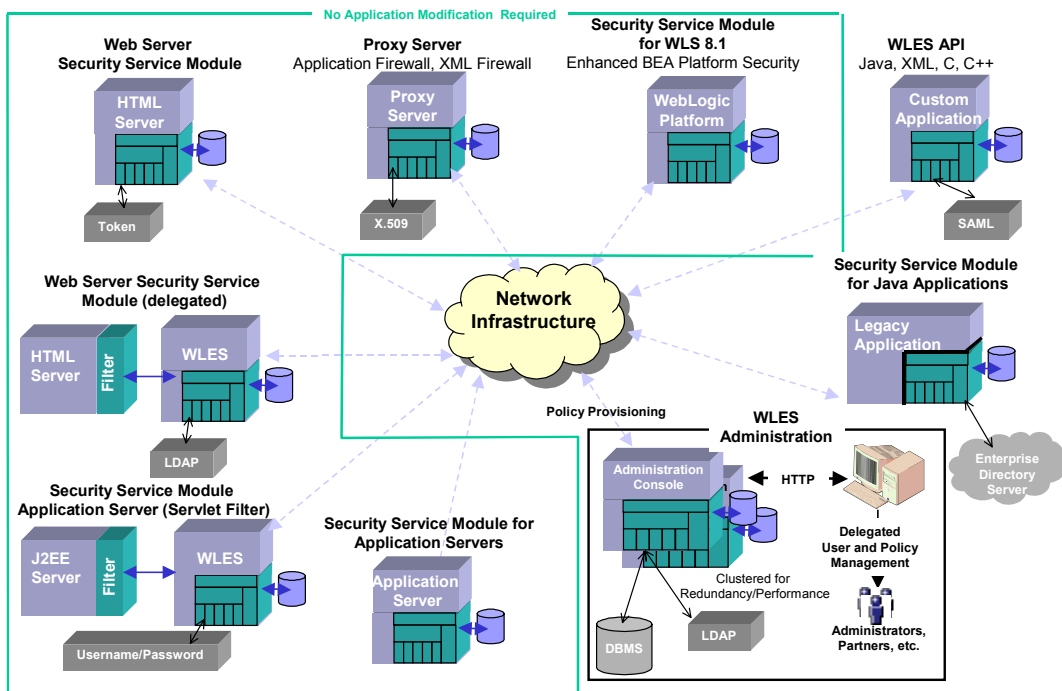
Your enterprise domain is divided into smaller domains, based on the number of Security Service Modules you have installed. Each Security Service Module may share or use different configuration or policy data, based on the business needs of an organization.

Distributed Computing Security Infrastructure

Applications across the enterprise are built on a heterogeneous infrastructure with diverse resources. With an application security infrastructure as shown in [Figure 1-2](#), the Security

Service Modules support a fully distributed architecture; all applications across the network are integrated.

Figure 1-2 Distributed Computing Security Infrastructure



The BEA WebLogic Enterprise Security products provide a variety of services that use its Security Framework, including enhanced policy-based authorization with role mapping, authentication with support for single sign-on and credential mapping, and customizable auditing features. A services-oriented strategy to application security infrastructure improves efficiency and strengthens security by providing a unified and consistent approach across the enterprise. BEA delivers security services that allow third-party security technologies to be exposed as reusable services, to further reduce integration time and costs, promote choice, and ensure investment protection.

The type of security services you implement depends on the type of the application component itself, and enforcement solutions are implemented as a set of providers delivered with each Security Service Module. The BEA WebLogic Enterprise Security services seek to provide ease

of use, manageability for end users and administrators, and customizability for application developers and security developers. Administrators who configure and deploy applications can use the providers included with the product that support most standard security functions.

- **WebLogic Server 8.1 Security Service Module**

Supports BEA WebLogic Server, Version 8.1 and enhances the existing security services in the application server, providing customizable auditing, multi-domain standards-based single sign-on, database and Microsoft Window NT authentication, database credential mapping, and expanded policy expression capabilities for authorization and role assignments.

- **IIS Web Server Security Service Module**

Supports the IIS Web Server. After installation, the security service module (SSM) binds with the web server through the web server application programming interface (ISAPI) so that the SSM can be used to protect web server application resources.

- **Apache Web Server Security Service Module**

Supports the Apache Web Server. After installation, the SSM binds the web server through the web server filter so that the SSM can be used to protect web server application resources.

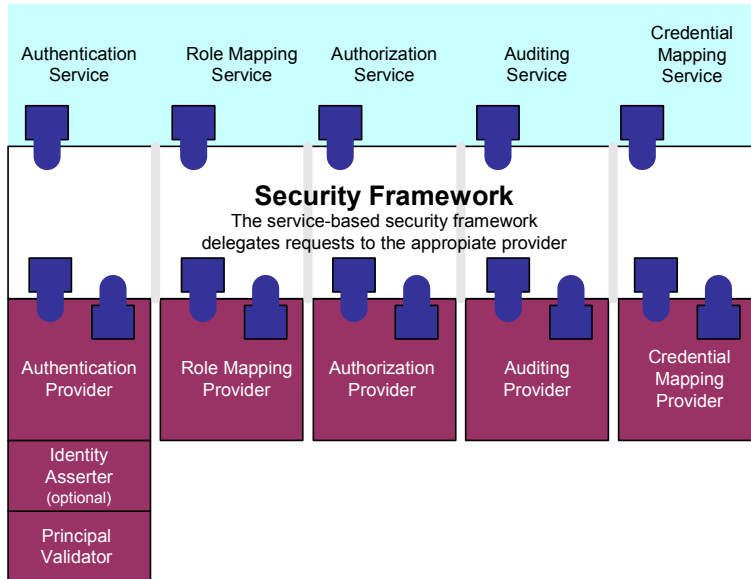
- **Web Services Security Service Module**

Supports web servers. After installation, the SSM security services can be accessed by a web server through the Web Services application programming interface and used to protect web server application resources.

- **Java Security Service Module**

An application programming interface (API) that allows security developers to develop environment interfaces or even integrate an application security infrastructure into an application. These interfaces support the most commonly required security functions and are organized into services that are logically grouped by functionality.

Figure 1-3 Security Framework



Each Security Service Module is delivered with a full set of security providers. [Table 1-1](#) lists the types of providers that are available for configuration. For information on configuring providers, see [“Security Configuration” on page 9-1](#).

Table 1-1 Security Providers

Provider	Description
Authentication Provider	<p>Supports open-standard support for SAML and X.509 identity assertion, and authentication support for Microsoft Windows NT, Microsoft Active Directory, Netscape LDAP, Novell LDAP, relational database, and OpenLDAP login modules.</p> <p>Identity Assertion Provider—enables perimeter-based authentication to support single sign-on, and allows users or system processes to assert their identity using certificate-based authentication, SAML, or other token formats.</p> <p>Principal Validation Provider—enhances security by signing and verifying the authenticity of the principals.</p>
Credential Mapping Provider	<p>Maps credentials used by a legacy or any remote system. The application then uses the appropriate credentials to log in to a remote system on behalf of a subject that already authenticated to support single sign-on.</p>
Authorization Provider	<p>Controls access to resources based on the role and policy assigned to the requested resource. An access decision is the part of the authorization provider that actually determines whether a user has permission to perform an operation on a resource.</p> <p>Secures access to resources and transactions, enabling an organization with increasingly complex user communities to provide secure finely-grained access to resources. Access decisions provided through a role-based authorization provider incorporate relevant environmental, contextual, and transaction-specific information, allowing security policies to support business processes throughout the organization.</p> <p>Adjudication Provider—resolves authorization conflicts when you configure multiple authorization providers.</p>
Role Mapping Provider	<p>Supports dynamic role associations by obtaining a computed set of roles granted to a requestor for a resource.</p>
Auditing Provider	<p>Provides an electronic trail of all transaction activity and can include changes to system configuration parameters, policy changes, and transactions. For each audit item, the information can include who, what, when, where, and sometimes why.</p>

Attributes

System administration infrastructure in WebLogic Enterprise Security is implemented using attributes that can be configured through the Administration Console; thus, it is necessary that you understand how they are used.

Providers contain a set of attributes that define configuration parameters for various Security Service Modules. Many attributes for administration have pre-set or default values. When the Administration Server starts, it reads the configuration from the database and overrides the default attribute values of the attributes with any values found. Every time you change an attribute using the Administration Console or administration tools, its value is stored in the database. Each instance of a Security Service Module has its own configuration, although it may share its configuration with other like modules.

Attributes may be associated with users or groups (subject attributes), resources (resource attributes) or policy requests (dynamic attributes). Characteristics that define users, groups directories are called identity attributes. Attributes may be descriptive, configure policy behavior, manage delegated administration, or be used in forming policy as part of the policy condition. Attributes must have a defined type, which denotes the range of legal values that an attribute may have. A number of predefined types exist, such as string, date, time, ip address, or you can supply custom attribute types. The value of the attribute may be assigned to only one instance of an attribute. For a more complete description of how to use attributes in rules, see “[Securing Resources and Defining Policy Rules](#),” in the *BEA WebLogic Enterprise Security Policy Managers Guide*.

Security Protections for System Administration Tools

All system administration operations are protected based on the user name employed to access a system administration tool. A user (or the group a user belongs to) must be a member of one of four security roles. These roles grant or deny a user access to various sets of system administration operations. The roles are Admin, Operator, Deployer, and Monitor. For additional information on Administration roles, see “[Administration Policy](#)” on page 3-1.

Administration Console

The Administration Console is a web application hosted by the Administration Server. You access the Administration Console from any machine on the local network that can communicate with the Administration Server through a web browser (including a browser running on the same machine as the Administration Server). The Administration Console allows you to manage your

enterprise domain containing multiple instances of Security Service Modules. For information on general use of the Administration Console, see [“Using the Console” on page 5-1](#).

Through the Administration Application, system administrators can perform all management tasks without having to learn about the underlying management architecture. These management capabilities include:

- [Security Configuration](#)
- [Failover and System Reliability](#)
- [Securing Resources and Defining Policy Rules](#)
- [Configuring a Metadirectory](#)
- [Deployment](#)
- [Import policy data](#)
- [Export Policy data](#)
- [Starting and Stopping Security Service Module Processes](#)
- [Configuring Secure Sockets Layer for a Production Environment](#)
- To learn more about any of these topics, simply click the associated link. For information on starting the Administration Console, see [“Using the Administration Console” on page 5-5](#). Online help is available from the Administration Console by clicking on the "?" icon located in the upper right portion of the console.

Failover and System Reliability

This section describes WebLogic Enterprise Security features that support recovery from failure. It covers the following topics:

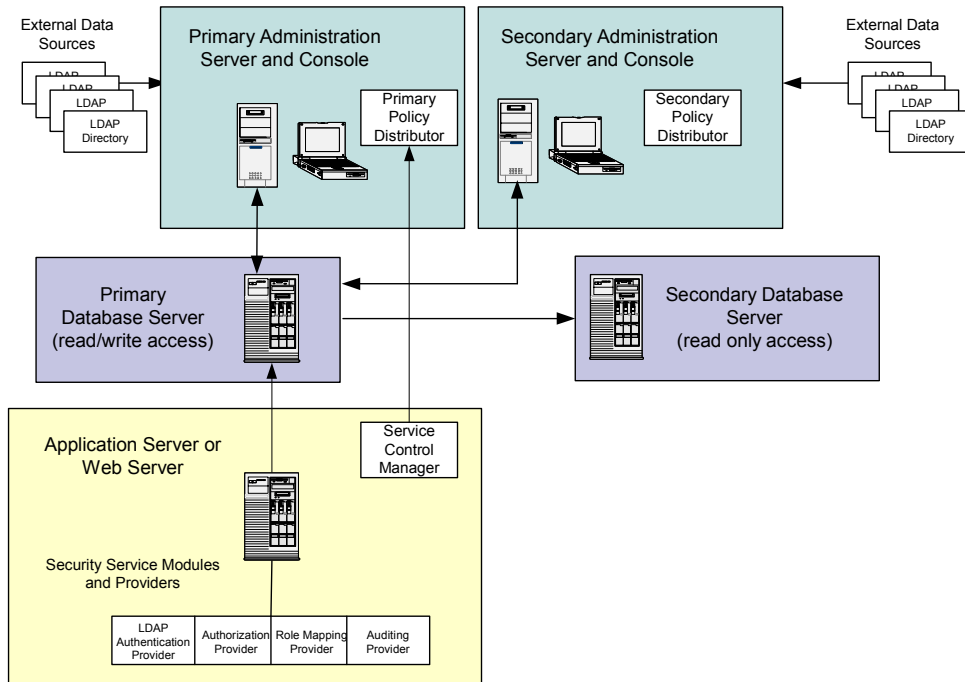
- [“Understanding Failover” on page 2-1](#)
- [“WebLogic Enterprise Security Failover Considerations” on page 2-2](#)
- [“Understanding Database Replication” on page 2-6](#)
- [“Database Replication in an Oracle Environment” on page 2-7](#)
- [“Database Replication in a Sybase Environment” on page 2-16](#)

Understanding Failover

[Figure 2-1](#) shows how WebLogic Enterprise Security System supports failover, through the use of a redundant component architecture. The Administration Console, external data sources (used by the LDAP providers), and the database server, all support failover through configuration in the Administration Console. To ensure reliable operations, you must consider the ramifications of the failure of each component.

To accommodate your basic needs, you can install two Administration Servers: a primary and a secondary. The number of redundant database servers you configure is totally up to you, although a minimum of two is recommended to maintain reliable services. The secondary Administration Server is only used when the primary becomes unavailable; likewise with the database server.

Figure 2-1 WebLogic Enterprise Security System Failover



WebLogic Enterprise Security Failover Considerations

The following sections discuss the failover considerations for the various WebLogic Enterprise Security components:

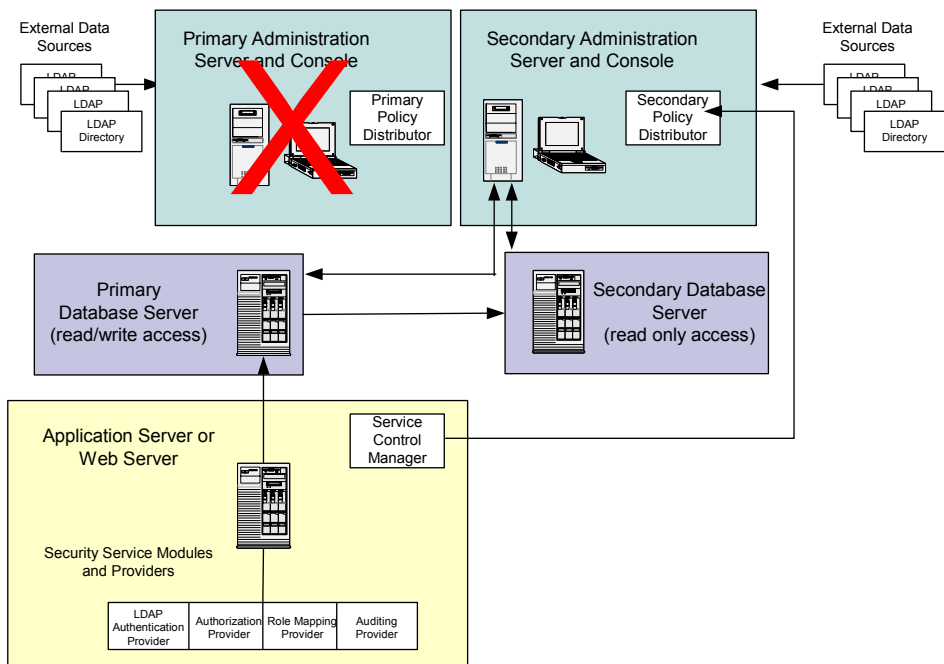
- [“Failover Considerations for the Administration Server”](#) on page 2-3
- [“Failover Considerations for the Database Server”](#) on page 2-4
- [“Failover Considerations for a Security Service Module”](#) on page 2-5
- [“Failover Considerations for a Service Control Manager”](#) on page 2-6

Failover Considerations for the Administration Server

Figure 2-2 shows how failover works when the primary Administration Server fails. One benefit of the WebLogic Enterprise Security architecture is that even if all the Administration Servers go down (either for maintenance or due to failure), including the secondary Administration Servers, there is no impact on the applications in production or on the security services provided by those Security Service Modules and providers that you have configured. As long as you have back up instances of your policy database, external data sources, and back ups of application files, you can safely restart the Administration Server on another machine without interrupting services. However, you cannot install or enroll new Security Service Modules until the primary Administration Server is running or you have reconfigured the secondary server as the primary. You can only enroll Security Service Modules using a primary Administration Server.

For information on how to configure the Administration Server, see your Console Help: *Configuring the Administration Server for Failover*.

Figure 2-2 Administration Server Failover

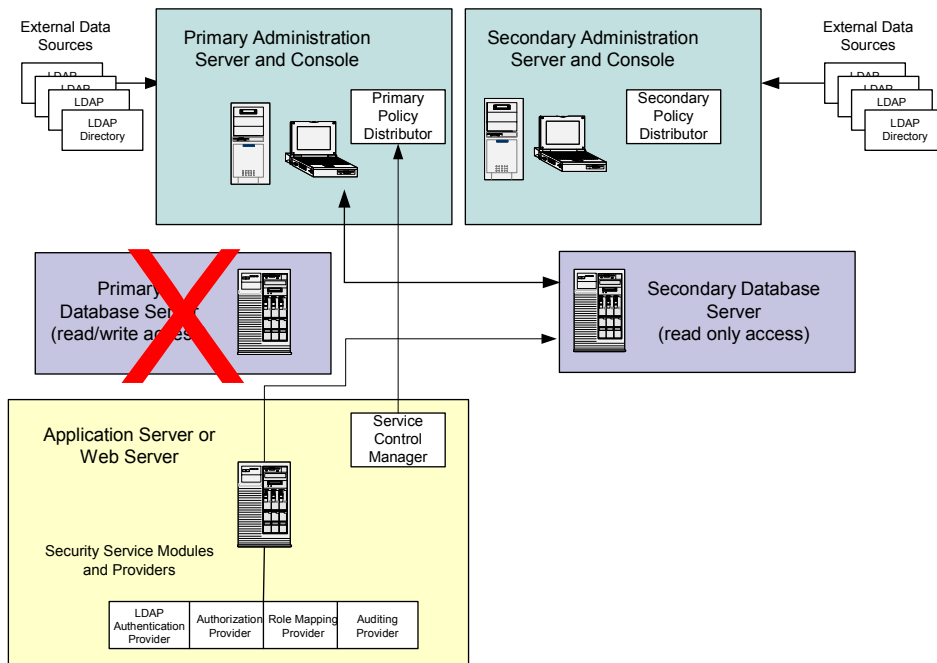


Failover Considerations for the Database Server

Figure 2-3 shows how failover works when the primary Database Server fails. The number of redundant database servers you configure is totally up to you. A minimum of two is recommended to maintain reliable services: a primary database is read/write, while the secondary provides read-only access. The secondary database, created by using the database replication procedures, contains only the data that is needed for the Security Service Modules, rather than a complete set of the configuration and security data. The secondary database is primarily used for failover in Security Service Modules and not the Administration Server.

Because the Database Server contains all of the configuration and security data used by the Administration Application, to protect your applications and resources, you want to make sure it is highly available and reliable. This can be accomplished by implementing recommendations from your database manufacturer (for example, through the use of clustering architecture; hot standby).

Figure 2-3 Database Failover



Common methods of archiving high availability include periodic back-ups, fault tolerant disks, and copying files manually whenever they are changed. This is also the case for any optional external data sources you have configured. Both Sybase and Oracle offer database backup methods. Refer to Sybase and Oracle documentation for details. The backup can be used for database recovery in the case of disk failure. To provide high availability for the Security Service Modules when using the security providers supplied with the product, you must perform certain database replication procedures as described in the [“Database Replication in an Oracle Environment” on page 2-7](#) and [“Database Replication in a Sybase Environment” on page 2-16](#).

Failover Considerations for a Security Service Module

Failover support through the Administration Console is provided for database-related providers and LDAP authentication providers. Configuration for database related providers includes the specification of the secondary (backup database) and support for LDAP authenticators includes the specification of the secondary LDAP server.

The following providers support configuration of a secondary database:

- Database Credential Mapping provider
- Database Authentication provider
- ASI Authorization provider
- ASI Role Mapper provider

Note: The ASI Authorization Provider contacts an external process to evaluate its authorization queries. If that process dies, the ASI Authorization provider denies access to all resources. The ASI Authorization provider may contact an external metadirectory database to retrieve subject attributes and group membership for use in authorization and delegation decisions. If the database connection fails, the provider connects to the configured failover replicated metadirectory. The provider tries to reconnect to the failed database after a configurable timeout. If all metadirectory connections fail and the policy evaluation is configured to require user attributes and group membership, all access is denied. For additional information on starting and stopping services, see [“Starting and Stopping Services” on page 6-1](#).

The following providers support configuration of a secondary LDAP server:

- Novell LDAP Authenticator
- Active Directory Authenticator
- iPlanet Authenticator

- Open LDAP Authenticator

For additional information on how to configure the provider for failover, see [“Security Configuration” on page 9-1](#) or the Console Help for the specific provider.

Note: The NT Authenticator already supports multiple domain controllers. The WebLogic Authenticator, WebLogic Authorizer and WebLogic Role Mapper use the internal LDAP server for WebLogic server as its data store. No support for a redundant source is required.

Failover Considerations for a Service Control Manager

The Service Control Manager has a mechanism for enrolling with multiple Administration Servers so that configuration can be administered from multiple Administration Consoles. A process manager is used to monitor the status of the Service Control Manager process and, if it stops responding or fails, the process manager restarts the Service Control Manager. The process manager runs as a Windows NT Service or UNIX daemon. This allows the Service Control Manager to be automatically restarted on machine reboots.

Note: If a Service Control Manager fails, all Security Service Modules controlled by the manager continue to run, but any new Security Service Module instances will not be able to start, because they cannot retrieve their configuration. However, a Security Service Module can still enroll with the Administration Server, even if the Service Control Manager fails.

Understanding Database Replication

To provide reliability and high availability of Security Service Modules, replication of part of the policy database is necessary. The replicated database objects needed for this replication process include tables and indices that are required by the Security Service Modules. They contain the data that are used by the following providers:

- Database Authentication provider
- Database Credential Mapper provider
- ASI Authorization and ASI Role Mapper providers

Note: The ASI Authorization and ASI Role Mapper providers are implemented using an out-of-process evaluation engine (the ARME process). If this process fails, access for all resources is denied. This process is monitored and restarted if an error occurs.

Most of the policy database used for security policy and for configuration administration and management is not replicated. The administration and management functions are performed against one and only one database. This means that the components such as the Administration Application, the Policy Distributor, and the Metadirectory Synchronization process are only connected to one database.

Warning: Your administrator needs to ensure that the primary database is backed up regularly using a database backup tool or using a hardware backup strategy to protect the database against fatal systematic errors, should one ever occur.

The steps included in following sections describe how to create a read-only database replica. While some scripts automate this process, to complete the replication, you need to perform certain steps manually. For instructions on how to replicate databases in the Oracle and Sybase environments, see the following sections:

- [“Database Replication in an Oracle Environment” on page 2-7](#)
- [“Database Replication in a Sybase Environment” on page 2-16](#)

Database Replication in an Oracle Environment

WebLogic Enterprise Security uses the Materialized View (MV) Replication method to replicate the database objects from a master database (also called the master site) to one or more materialized view databases (also called the materialized view sites). In an earlier version of Oracle, such as Oracle 8i, a materialized view replication was called a snapshot replication, and a materialized view site was called a snapshot site. The master database is the database that is administered through the Administration Application. The database schema of the master database is installed and set up during the Administration Application installation. The materialized view database provides failover and availability for the Security Service Modules. You can set up the Security Service Modules at a later time, according to your business needs.

The following sections describe the replication tasks:

- [“Preparing for Oracle Database Replication” on page 2-7](#)
- [“Setting Up Oracle Database Replication” on page 2-12](#)

Preparing for Oracle Database Replication

Use the following items as a checklist to prepare your replication environment.

- [“Master and Materialized View Site Requirements” on page 2-8](#)

- “Master Site Requirements” on page 2-10
- “Materialized View Site Requirements” on page 2-10
- “Requirements for the Machine Running the Replication Setup Scripts” on page 2-11
- “Setting the Required Replication Setup Parameters” on page 2-11

Warning: If you are concerned with the database password being used in the scripts, you can run the replication setup manually. All passwords are designated as clear text in the `set_repl_env_oracle.sh` and `set_repl_env_oracle.bat` scripts. When you are through, you can either delete the script or remove the password from the script.

Master and Materialized View Site Requirements

Requirements for the master site and the materialized view site include the following:

- You are able to login to the database as database administrator (DBA) and you have the privileges to restart the database.
- Ensure that the database name is in the form of `DB_NAME.DB_DOMAIN`. This ensures that the database links can be set up correctly. You can check this by running this SQL statement after logging into the database:

```
select * from global_name;
```

If the `DB_DOMAIN` is missing, correct it. This task requires DBA privileges.

- Change the initialization parameters for both databases (the master database and the materialized view database) according to [Table 2-1](#). The value for each initialization parameter depends on your use of the database. Refer to [Table 2-1](#), and set the parameters as follows:
 - If the default values for the parameters that are defined as “Required” in the Comment field do not suit your needs, change them.
 - Set the parameters that are defined as “Optional” in the Comment field according to the use of your database.

For detailed information on these parameters, refer to your Oracle Documentation. For Oracle 9i, see Chapter 6: “Planning Your Replication Environment” in *Advanced Replication* (for Oracle 9.2) or *Oracle9i Replication* (for Oracle 9.0). For Oracle 8i, see Chapter 7: “Planning Your Replication Environment” of *Oracle8i Replication*. Also check the Oracle documentation for details on how to set the initialization parameters.

Table 2-1 Initialization Parameters

Parameter	Recommended Value	Comment
GLOBAL_NAMES	true	Required
JOB_QUEUE_PROCESSES	at least 1	Required
OPEN_LINKS	at least 4	Required
PROCESSES	at least 40	Required
REPLICATION_DEPENDENCY_TRACKING	true	Required
COMPATIBLE		Optional
PARALLEL_MAX_SERVERS		Optional
PARALLEL_MIN_SERVERS		Optional
SHARED_POOL_SIZE		Optional
Oracle 9i only		
PARALLEL_AUTOMATIC_TUNING		Optional
UTL_FILE_DIR	A valid directory in the host machine of the database	Required: the directory must be created
Oracle 8i only		
ENQUEUE_RESOURCES		Optional
JAVA_POOL_SIZE		Optional
JOB_QUEUE_INTERVAL		Optional
MAX_ENABLED_ROLES		Optional

Table 2-1 Initialization Parameters (Continued)

Parameter	Recommended Value	Comment
MTS_DISPATCHERS	PROTOCOL=TCP)(PRE=oracle .aurora.server.SGiopServer)	Required
OPEN_CURSORS		Required

Master Site Requirements

Requirements for the master site include the following:

- Your master database must be either an Oracle Enterprise edition or an Oracle Standard edition.
- Ensure that the replication administrator `REPADMIN` and `PROXY_ADMIN` does not already exist in this database. If either one already exists, you cannot automate the replication setup using the scripts because the scripts will fail to create them. You need to set up the master site manually. You can use SQL `select * from all_users` to check for its existence.
- Ensure that the policy database schema is already installed. You need to know the schema owner (`WLES` user) and login password.
- Ensure that the size of the `TABLESPACE` used by the `WLES` user is large enough to accommodate the extra storage needed by replication.

Materialized View Site Requirements

Requirements for the materialized view site include the following:

- In this host machine, set up an Oracle Local Service Name for the master database using the same name as the master database name, in the form of `DB_NAME.DB_DOMAIN`.
- Ensure that the replication administrator `MVADMIN` does not already exist in this database. If it already exists, you cannot automate the replication setup using the scripts because the scripts will fail to create it. You need to set up the materialized view site manually.
- Ensure that the same `WLES` user as in master database does not exist in this database as this user is created. However, you need to locate a `TABLESPACE` for this `WLES` user before the setup using the scripts. By default, the temporary `TABLESPACE` for this user is set to `TEMP`.
- Decide how frequently you want the replicated objects refreshed in this database. Set the refresh time interval to a multiple of one minute.

Requirements for the Machine Running the Replication Setup Scripts

On this machine, set up Oracle Local Service Names for the master database and materialized view database, using the same names as their database names, in the form of:

`DB_NAME.DB_DOMAIN`

Setting the Required Replication Setup Parameters

[Table 2-2](#) lists and describes the parameters that you must set before you can set up the replication. You must set these parameters in the setup script:

`set_repl_env_oracle.bat` (Windows)

or

`set_repl_env_oracle.sh` (Unix/Linux)

In addition to the parameters listed in [Table 2-2](#), you need to set `ORACLE_HOME`, if it is not yet set in your environment, and the `__REPL_SETUP_TYPE__` parameter, depending on what kind of replication setup you choose.

Table 2-2 Oracle Replication Setup Parameters

Parameter Name	Description	Example
<code>__MASTER_SERVICE_NAME__</code>	Service name of master database (<code>DB_NAME.DB_DOMAIN</code>).	<code>mstrdb.bea.com</code>
<code>__MASTER_SYSTEM_PASSWD__</code>	Password for <code>SYSTEM</code> user (DBA) in master database.	<code>Pswd000</code>
<code>__REPADMIN_PASSWD__</code>	Password for <code>REPADMIN</code> (to be created/dropped, replication admin) in master database.	<code>Pswd111</code>
<code>__PROXY_MVADMIN_PASSWD__</code>	Password for <code>PROXY_MVADMIN</code> (to be created/dropped) in master database.	<code>Pswd222</code>
<code>__MASTER_PURGE_INTEVAL__</code>	Time interval in minutes to purge completed deferred transactions at master site.	<code>60</code>
<code>__MV_SERVICE_NAME__</code>	Service name of materialized view (MV) database (<code>DB_NAME.DB_DOMAIN</code>).	<code>mvdb.bea.com</code>

Table 2-2 Oracle Replication Setup Parameters (Continued)

Parameter Name	Description	Example
__MV_SYSTEM_PASSWD__	Password for SYSTEM user (DBA) in materialized view database.	Pswd333
__MVADMIN_PASSWD__	Password for MVADMIN (to be created/dropped, replication admin) in MV database.	Pswd444
__TBLSPACE_DBUSER_WLES__	Tablespace for WLES database user (__DBUSER_WLES_UPPERCASE__) in MV database.	USERS
__MV_PURGE_INTEVAL__	Time interval in minutes to purge completed deferred transactions at MV site.	60
__MV_PUSH_INTEVAL__	Time interval in minutes to push deferred transactions to master at MV site.	60
__REFRESH_INTEVAL__	Time interval in minutes to refresh the refresh group at MV site.	2
__DBUSER_WLES_UPPERCASE__	The policy database user name in both the master and materialized view databases; must be in UPPER CASE. It cannot exist in the materialized view database before setting up replication. This user is created to use tablespace: __TBLSPACE_DBUSER_WLES__	WLES_USER
__DBUSER_PASSWD_WLES__	The database user password in both master and materialized view databases.	Pswd555

Setting Up Oracle Database Replication

Use the following procedures to set up the Oracle database replication using scripts or manually.

- [“Using Scripts to Set Up Oracle Database Replication” on page 2-13](#)
- [“Setting Up Oracle Database Replication Manually” on page 2-14](#)
- [“Using Scripts to Clean Up Oracle Database Replication” on page 2-14](#)

- [“Cleaning Up the Oracle Database Replication Manually” on page 2-15](#)
- [“Miscellaneous Oracle Database Replication Tasks” on page 2-15](#)

Warning: You must set up the master site before replication setup can be performed for a materialized view site.

Using Scripts to Set Up Oracle Database Replication

To use scripts to set up replication, perform the following steps:

1. Change to the directory:

```
WLES_HOME/bin
```

2. To set up the environment and input parameters, edit the following file:

```
set_repl_env_oracle.bat (Windows)
```

or

```
set_repl_env_oracle.sh (Unix/Linux)
```

Save a copy of this file before editing. The parameter `__REPL_SETUP_TYPE__` gives you the option to set up either a master site, a materialized site or both.

3. Run the following script:

```
replicate_oracle.bat (Windows)
```

or

```
replicate_oracle.sh (Unix/Linux)
```

You are prompted to continue. A message is displayed when the script completes, indicating whether the execution was successful or if it failed. The following log file shows the status of the execution:

```
WLES_HOME/log/replicate_oracle.log
```

4. After the setup completes, edit or delete the following file and close the window:

```
set_repl_env_oracle.bat (Windows)
```

or

```
set_repl_env_oracle.sh (Unix/Linux)
```

Setting Up Oracle Database Replication Manually

To set up replication manually, perform the following steps:

1. In the `WLES_HOME/data/oracle/` directory, locate these SQL files:
`rep_mastersite.sql`
`rep_mvsite.sql`
2. Save a copy of each one.
3. Edit them by globally replacing the variables with your settings, as instructed at the beginning of each file.
4. Manually execute the step-by-step SQL statements using the SQLPLUS tool. The comments in these SQL files describe what each step does.

Using Scripts to Clean Up Oracle Database Replication

Warning: Replication clean up must be performed for all materialized view sites before the master site can be cleaned up.

To use scripts to clean up the replication, perform the following steps:

1. Change to the directory:
`WLES_HOME/bin`
2. Set the environment and input parameters in the following file:

`set_repl_env_oracle.bat` (Windows)

or

`set_repl_env_oracle.sh` (Unix/Linux)

Save a copy of this file before editing it. The parameter `__REPL_SETUP_TYPE__` gives you the option to clean up either a master site, a materialized site or both.

3. Run the following script:
`clean_repl_oracle.bat` (Windows)
or
`clean_repl_oracle.sh` (Unix/Linux)

You are prompted to continue. A message is displayed when the script completes, indicating whether the script was successful or if it failed. The following log file shows the status of the execution:

```
WLES_HOME/log/clean_repl_oracle.log
```

4. After the cleanup completes, edit or delete following file and close the window:

```
set_repl_env_oracle.bat (Windows)
```

or

```
set_repl_env_oracle.sh (Unix/Linux)
```

Cleaning Up the Oracle Database Replication Manually

To clean up the replication manually, perform the following steps:

1. In the `WLES_HOME/data/oracle/` directory, locate the following SQL files:

```
rep_clean_master.sql
```

```
rep_clean_mv.sql
```

2. Save a copy of each one.
3. Edit the files by globally replacing the variables with your settings, as instructed at the beginning of each file.
4. Manually execute the SQL statements step-by-step using the SQLPLUS tool. The comments in these SQL files describe what each step does.

Miscellaneous Oracle Database Replication Tasks

To complete the replication of the Oracle database, perform the following steps:

1. The database name must be in the form of `DB_NAME.DB_DOMAIN`. If the database name is not in this form, login to the database as DBA and run the following SQL command to change it:

```
alter database rename global_name to DB_NAME.DB_DOMAIN;
```

2. Change the initialization parameters (refer to Oracle documentation on how change the initialization parameters):

```
db_name: to DB_NAME
```

```
db_domain: to DB_DOMAIN
```

```
service_name: to DB_NAME.DB_DOMAIN
```

3. Change the listener for the new `global_name`.
4. If you change the `global_name`, update the Oracle listener setting, and then restart the listener. The `GLOBAL_NAME` is recorded in the Oracle configuration file called `listener.ora` in the `ORACLE_HOME/network/admin/` directory.
5. Change the setting for the Local Service Name in all Oracle clients.

If you change the local service name that points to this database, you must update it on all applicable client machines. Change the `SERVICE_NAME` of the `CONNECT_DATA` to the new `service_name` in client configuration file `tnsnames.ora`, usually in the `ORACLE_HOME/network/admin/` directory. If your system uses Directory Naming or Oracle Naming methods, update them instead of `tnsnames.ora`. Refer to your Oracle documentation for details.

Database Replication in a Sybase Environment

WebLogic Enterprise Security uses the Sybase ASE Replicator to replicate the database objects from a primary database in primary Adaptive (ASE) server to one or more replicate databases in the replicate ASE servers. The ASE Replicator process is an external application that connects to and interacts with the ASE server, and coordinates all replication processing. The policy in the primary database is administered through the Administration Server. The database schema in the primary database is installed and set up during the Administration Application installation. The replicate databases provide failover and availability for the WebLogic Enterprise Security Security Service Modules. They can be set up according to your business needs at a later time. For more information, see the *ASE Replicator User's Guide* from Sybase.

The following sections describe the replication tasks:

- [“Preparing for Sybase Database Replication” on page 2-16](#)
- [“Setting Up Sybase Database Replication” on page 2-21](#)
- [“Cleaning Up Sybase Database Replication” on page 2-25](#)

Preparing for Sybase Database Replication

The following sections provide checklists so you can prepare your replication environment:

- [“Privileges for the Primary and the Replicate ASE Servers” on page 2-17](#)
- [“Primary ASE Server and Primary Database Requirements” on page 2-17](#)
- [“Replicate ASE Server and Replicate Database Requirements” on page 2-18](#)

- “Requirements for the Machine Used to Run Sybase Database Replication Setup Scripts” on page 2-19
- “Parameters Needed for Sybase Database Replication Setup” on page 2-19

Warning: If you are concerned with the database password being used in the scripts, BEA recommends that you run the replication setup manually. All passwords are designated in clear text in the `set_repl_env_sybase.sh` and `set_repl_env_sybase.bat` scripts. When you are through, you can either delete the script or remove the password from the script.

Privileges for the Primary and the Replicate ASE Servers

Make sure you are able to login to the servers as DBA (sa) and you have the privilege to restart the servers.

Primary ASE Server and Primary Database Requirements

The primary ASE server and primary database requirements are as follows:

- Your server must be of ASE version 12.5.0.3 or later. This version includes the ASE Replicator software. You need to patch it if the server is of earlier version of 12.5.
- The ASE Replicator is started and is running in this machine, even though it can be run from other machines that has ASE Replicator software installed.
- Ensure the ASE Replicator system user login does not already exist in this server. You can specify the name of this login in the setup scripts. If the system user login already exists, you cannot automate the replication setup using the scripts because the scripts will fail to create it, and you will need to set up the replication manually. You can use SQL select name from syslogins to check its existence.
- Ensure that the policy database schema is already installed. You need to know the schema owner (wles user) and login password.
- Ensure that you (sa) have created the database for the replication process to use as the distribution database. Before creating the database, you may need to create two database devices, one used for data and another used for transaction log. Allocate sufficient database storage. You can use the `disk init ...` and `create database ... isql` commands, or use another GUI tool to set these up. Also, while this database is dropped when running the replication cleanup scripts, the database devices are not. See the [Administration Application Installation Guide](#) for instructions on how to set up the database.

- In this host machine, set up the following server entries in Sybase configuration file using either the Sybase tool Dsedit or using a text editor. The configuration file is:

`sql.ini` (Windows)

or

`interfaces` (Unix/Linux).

- Primary ASE server—the same name as the primary database server.
- ASE Replicator—the name and port number you choose here is used later when you start the ASE Replicator.
- Replicate ASE server—the same name as the replicate server.

Replicate ASE Server and Replicate Database Requirements

The replicate ASE server and replicate database requirements are as follows:

- Ensure that this ASE server is installed and configured with logical page size equal to or larger than the logical page size of the primary ASE server. Use SQL statement:
`select @@maxpagesize` to check them out.
- Ensure the replication Maintenance user login does not already exist in this server. The same ASE Replicator system user login is used. If it already exists, you cannot automate the replication setup using the scripts because the scripts will fail to create it, and you will need to set up the replicate database manually.
- Ensure the same WLES user login as in primary ASE server does not exist in this server, as it is created. If it already exists, you cannot automate the replication setup using the scripts because the scripts will fail to create it, and you will need to set up the replication manually.
- Ensure that you (sa) have created a new database to use as the replicate database. This database is used to store replicated policy data. Before creating the database, you may need to create two database devices, one used for data and another used for transaction log. Allocate sufficient database storage. Transaction log size can be much smaller than that in policy database in the primary ASE server. Also, while this database and the database devices are not be dropped when running the replication cleanup scripts, the schema for WLES user is dropped.

Requirements for the Machine Used to Run Sybase Database Replication Setup Scripts

The requirements for the machine used to run the Sybase database replication setup Scripts are as follows:

Note: If you use the primary ASE server machine to run the Sybase database replication setup scripts, you do not have to do anything and you can skip this section.

- In this host machine, set up the following server entries in the Sybase configuration file using Sybase tool Dsedit or using text editor. The configuration file is:

`sql.ini` (Windows)

or

`interfaces` (Unix/Linux)

You can copy this file from the primary ASE server machine.

- Primary ASE server—the same name as the primary database server.
- ASE Replicator—the server name of the ASE Replicator.
- Replicate ASE server—the same name as the replicate server.

Parameters Needed for Sybase Database Replication Setup

[Table 2-3](#) lists the initialization parameters you need to set before you set up the replication. You set these parameters in:

`set_repl_env_sybase.bat` (Windows)

or

`set_repl_env_sybase.sh` (Unix/Linux)

You also need to set SYBASE if it is not yet set in your environment. In addition, you need to set `__REPL_SETUP_TYPE__` depending on what kind of replication setup you choose.

Table 2-3 Sybase Initialization Parameters

Parameter Name	Description	Example
__PRIMARY_DBSERVER__	Name of the primary ASE server.	PRISVR
__PRIMARY_DBNAME__	Name of the primary policy database in the primary ASE server.	policy
__SA_PASSWD_PRIMARY__	Password for sa (DBA) in the primary ASE server.	Pswd000
__DISTRIBUTION_DBNAME__	Distribution database name in the primary ASE server. This database is used by the replication process (or Replicator). The database contains the database schema (tables, etc.) used by the replicator.	distrDB
__ASE_REPLICATOR_NAME__	Name of the ASE Replicator. This name should be the same as that in the Sybase configuration file.	PRISVR_RPL
__REPLICATE_DBSERVER__	Name of the replicate ASE server.	RPLSVR
__REPLICATE_DBNAME__	Name of the secondary policy database in the replicate ASE server.	policy
__SA_PASSWD_REPLICATE__	Password for sa (DBA) in the replicate ASE server.	Pswd111
__USER_REPADMIN__	ASE Replicator system username/login and Maintenance username/login.	repadmin
__PASSWD_REPADMIN__	Password for user __USER_REPADMIN__.	Pswd222
__DBUSER_WLES__	Policy database username/login in both primary and replicate ASE servers.	wlesuser
__DBUSER_PASSWD_WLES__	Policy database password in both primary and replicate ASE servers.	Pwd333
__REPL_SETUP_TYPE__	Indicates whether to set or clean up both primary and replicate databases. The value is one of these: both, primary, or replicate.	both

You also need to know the values for __SIZE_IDXCOL__ and __SIZE_COL__ when you set up a replication manually. The value for these two can be found in the policy database in the primary ASE server using SQL statement:

```
select __SIZE_IDXCOL__ = idx_column_size, __SIZE_COL__ =
reg_column_size from __DBUSER_WLES__ column_sizes
```

Note: `__DBUSER_WLES__` is substituted with the value described.

Setting Up Sybase Database Replication

Use the following procedures to set up the Sybase database replication using scripts or manually.

- [“Using Scripts to Set Up Sybase Database Replication” on page 2-21](#)
- [“Setting Up Sybase Database Replication Manually” on page 2-22](#)
- [Setting up the Primary ASE Server and the Primary Database](#)
- [“Starting the ASE Replicator” on page 2-23](#)
- [“Adding a Remote Server in Primary ASE Server for the Replicate ASE Server” on page 2-24](#)
- [“Setting Up the Replicate ASE Server and the Primary Database” on page 2-24](#)
- [“Setting up the Sybase Database Replication Process” on page 2-25](#)

Warning: The Primary ASE server and database must be set up and ASE Replicator started before you can perform replication setup on a replicate database.

Using Scripts to Set Up Sybase Database Replication

To use scripts to set up Sybase database replication, perform these steps:

1. Change to the following directory:

```
WLES_HOME/bin
```

2. Edit the following file to set the environment and input parameters:

```
set_repl_env_sybase.bat (Windows)
```

or

```
set_repl_env_sybase.sh (Unix/Linux)
```

Save a copy of this file before editing it. The parameter `__REPL_SETUP_TYPE__` allows you to set up either a primary, replicate database, or both.

3. Run the following script:

```
replicate_sybase.bat (Windows)
```

or

```
replicate_sybase.sh (Unix/Linux)
```

4. You are prompted to continue.
5. You are then prompted to restart the primary ASE server. Go to the primary ASE server machine, and stop and start the ASE server. After the server is started, press <Enter> to continue.
6. You are prompted to start the ASE Replicator. Go to the primary ASE server machine and start the ASE Replicator process. After the server is started, press <Enter> to continue.

You are prompted to continue. A message is displayed when the script completes, indicating whether the script was successful or if it failed. The following log file shows the status of the execution:

```
WLES_HOME/log/replicate_sybase.log
```

7. Edit or delete the file:

```
set_repl_env_sybase.bat (Windows)
```

or

```
set_repl_env_sybase.sh (Unix/Linux)
```

8. Close the window after the setup completes.

Setting Up Sybase Database Replication Manually

To set up Sybase database replication manually, perform these steps:

1. In the `WLES_HOME/data/sybase/` directory, locate and copy the following SQL files:

```
rep_pdb_conf.sql  
rep_rsvr_rdb.sql  
rep_rdb_conf.sql  
rep_replication.sql
```
2. Edit them by globally replacing the variables with your settings, as instructed in each file.
3. Manually execute the SQL statements step-by-step using the `isql` tool. The order is described in the following sections. When you are setting up a second replicate database, you need to perform the following tasks:
 - [Setting up the Primary ASE Server and the Primary Database](#)

– [Starting the ASE Replicator](#)

Setting up the Primary ASE Server and the Primary Database

Login to the primary ASE server as sa (DBA) and execute: `rep_pdb_conf.sql`. You can do this statement-by-statement. You also need to restart the primary ASE server manually, after running this script. This SQL file performs the following tasks.

1. Enables and configures CIS in the primary ASE server.
2. Sets up ASE Replicator system user login and assigning it to replication role.
3. Adds the ASE Replicator system user to the policy database (primary database) and grants its permissions.
4. Defines remote servers in the primary ASE server.
5. Defines the local server name for the primary ASE server.
6. Defines a server named `local` as a remote alias for the primary ASE server.
7. Defines a remote server for the ASE Replicator.
8. Configures the `tempdb` database.
9. Sets up ASE Replicator system user and changes the database option in the distribution database.
10. Adds the ASE Replicator system user to the distribution database.
11. Grants permissions to ASE Replicator system user in the distribution database.
12. Changes the database option in the distribution database.
13. Sets up the ASE Replicator system user in database `sybsystemprocs`.
14. Creates the `sp_helpddb` stored procedure in database `sybsystemprocs`, and grants execution permission to ASE Replicator system user.

Starting the ASE Replicator

To start the ASE replication, perform these steps:

1. Go to the primary ASE server machine and change to the directory: `RPL-12_5/bin` in the Sybase installation directory.
2. If ASE Replicator is started the first time:

- a. Make sure the `RPL-12_5` directory has write permission.
- b. Using the proper command options, execute the following command:

`aserep.bat` (Windows)

or

`aserep.sh` (Unix/Linux)

Check the *ASE Replicator User's Guide* for details.

Note: You can restart the ASE Replicator by running either: `aserep.bat` or `aserep.sh` using the same command options, or by running the `RUN` script that is located under `SYBASE/RPL-12_5/replicatorName` and providing the `-u` and `-p` options.

Adding a Remote Server in Primary ASE Server for the Replicate ASE Server

To add a remote server in the primary ASE server for the replicate ASE server, perform these steps:

1. Login to primary ASE server as sa (DBA).
2. Execute the following script:

`rep_rsvr_rdb.sql`

Setting Up the Replicate ASE Server and the Primary Database

Login to the replicate ASE server as sa (DBA) and execute: `rep_rdb_conf.sql`. You can do this statement-by-statement. Before starting, make sure the replicate policy database already exists. This SQL file performs the following tasks.

1. Adds the `WLES` user login.
2. Adds the `WLES` user login to the replicate database.
3. Creates the replicate tables and indices in replicate database, according to the logical page size setting of primary database.
4. Adds a Maintenance user login (for example, `repadmin`). This is the same as ASE Replicator system user login.
5. Adds `repadmin` as a user in replicate database.
6. Grants permission to `repadmin`.

Setting up the Sybase Database Replication Process

Login to primary ASE server as ASE Replicator system user login and execute: `rep_replication.sql`. You can do this statement-by-statement. Before starting, make sure the ASE Replicator is running. This SQL file performs the following tasks:

1. Creates the primary database connection if it does not yet exist.
2. Creates a replicate database connection.
3. Suspends both the replicate and primary database connections.
4. Creates a publication.
5. Adds the articles (replicated objects) to publication.
6. Creates a subscription.
7. Adds the articles (replicated objects) to subscription.
8. Materializes the replication articles (replicated objects).
9. Resumes both replicate and primary database connections to start replication.

Cleaning Up Sybase Database Replication

Use the following procedures to automate replication cleanup or to do it manually. Replication cleanup also stops the ASE Replicator and drops the distribution database in the primary ASE server. If you experience a failure while using the scripts to clean up replication, you are advised to finish the cleanup manually.

Before attempting cleanup, make sure that there is no database connection for `WLES` user login to the replicate ASE server and for ASE Replicator system user login to both replicate and primary ASE servers. If there are connections, cleanup will fail. You can use `sp_help loginName` to check whether the user still has connections.

- [“Using Scripts to Clean Up Sybase Database Replication” on page 2-26](#)
- [“Cleaning Up the Sybase Database Replication Manually” on page 2-27](#)
- [“Cleaning Up the Sybase Database Replication Process” on page 2-27](#)
- [“Cleaning Up the Replicate ASE Server and Primary Database” on page 2-28](#)
- [“Removing the Remote Server” on page 2-28](#)

- [“Stopping ASE Replicator” on page 2-28](#)
- [“Cleaning Up the Primary ASE Server and the Primary Sybase Database” on page 2-29](#)
- [“Completing Sybase Database Replication Cleanup” on page 2-29](#)

Warning: You must perform the replication clean up for all replicate databases before the primary database can be cleaned up and the ASE Replicator removed.

Using Scripts to Clean Up Sybase Database Replication

To use scripts to clean up Sybase database replication, perform these steps:

1. Make sure that there is no database connection for WLES user login to the replicate ASE server, and for ASE Replicator system user login to both the replicate and primary ASE servers.
2. Change to the following directory:

```
WLES_HOME/bin
```

Set the environment and input parameters in the following file:

```
set_repl_env_sybase.bat (Windows)
```

or

```
set_repl_env_sybase.sh (Unix/Linux)
```

Save a copy of this file before editing it. The parameter `__REPL_SETUP_TYPE__` gives you the option to clean up either a primary database and ASE server, a replicate database and ASE server, or both.

3. Run the following script:

```
clean_repl_sybase.bat (Windows)
```

or

```
clean_repl_sybase.sh (Unix/Linux)
```

You are prompted to continue. A message is displayed when the script completes, indicating whether the execution was successful or if it failed. The following log file shows the status of the execution:

```
WLES_HOME/log/clean_repl_sybase.log
```

4. Edit or delete the following file:

```
set_repl_env_sybase.bat (Windows)
```


or

```
set_repl_env_sybase.sh (Unix/Linux)
```

5. Close the window after the cleanup completes.
6. To complete replication cleanup, remove the ASE Replicator process by deleting all files from the `RPL-12_5/replicatorName` Sybase installation directory.

Cleaning Up the Sybase Database Replication Manually

To clean up Sybase database replication manually, perform these steps:

1. In the `WLES_HOME/data/sybase/` directory, locate the following SQL files and save a copy of each one:

```
rep_clean_replication.sql
```

```
rep_clean_rdb.sql
```

```
rep_clean_rsvr_rdb.sql
```

```
rep_clean_stop_rpl.sql
```

```
rep_clean_pdb.sql
```

2. Edit each file globally, replacing the variables with your settings, as instructed in each file.
3. Manually execute the SQL statements step-by-step using `isql` tool. The order is described below. When you are cleaning up a replicate database alone, you need to perform the following tasks:
 - [“Cleaning Up the Replicate ASE Server and Primary Database” on page 2-28](#)
 - [“Removing the Remote Server” on page 2-28](#)

Cleaning Up the Sybase Database Replication Process

Login to the primary ASE server as the ASE Replicator system user login and execute: `rep_clean_replication.sql`. You can do this statement-by-statement. This SQL file performs the following tasks:

1. Suspends both the replicate and primary database connections.
2. Drops the articles (replicated object) from subscription.
3. Drops the subscription.
4. Drops the articles (replicated object) from publication.

5. Drops the publication.
6. Drops the replicate database connection.
7. Drops the primary database connection if it is not being used for any other publications; resumes the connection if it is used.

Cleaning Up the Replicate ASE Server and Primary Database

Login to replicate ASE server as sa (DBA) and execute: `rep_clean_rdb.sql`. You can do this statement-by-statement. This SQL file performs the following tasks:

1. Drops the replicate tables. Indices are dropped automatically.
2. Drops the `WLES` user in replicate database.
3. Deletes the `WLES` user login in replicate ASE server.
4. Drops the Maintenance user in replicate database.
5. Deletes the Maintenance user login in the replicate ASE server.

Removing the Remote Server

In the Primary ASE Server for the Replicate ASE Server, follow these steps to remove the remote server:

1. Login to the primary ASE server as sa (DBA).
2. Execute the following script:
`rep_clean_rsrv_rdb.sql`
3. In the primary ASE server, remove the remote server for the replicate ASE server.

Stopping ASE Replicator

To stop the ASE Replicator, perform these steps:

1. Login to the ASE Replicator, using the ASE Replicator system user login.
2. Execute the following script to shut down the ASI Replicator:

```
rep_clean_stop_rpl.sql
```

Note: Alternatively, shutdown ASE Replicator manually.

Cleaning Up the Primary ASE Server and the Primary Sybase Database

Login to primary ASE server as sa (DBA) and execute: `rep_clean_pdb.sql`. You can do this statement-by-statement. This SQL file performs the following tasks.

1. Drops the procedure `sp_helpddb` in database `sybsystemprocs`.
2. Drops the ASE Replicator system user in database `sybsystemprocs`.
3. Drops the ASE Replicator system user in primary policy database.
4. Removes remote servers in the primary ASE server.
5. Removes the local server name for the primary ASE server.
6. Removes the server named `local` as a remote alias for the primary ASE server.
7. Removes the remote server for the ASE Replicator.
8. Drops the distribution database.
9. Deletes the ASE Replicator system user login.

Completing Sybase Database Replication Cleanup

To complete replication cleanup, remove the ASE Replicator process by deleting all of the files under `RPL-12_5/replicatorName` in your Sybase installation directory.

Administration Policy

This section provides information on the following topics:

- [“Security Roles” on page 3-1](#)
- [“Dynamic Role Mapping” on page 3-2](#)
- [“Understanding the Administration Policy” on page 3-2](#)
- [“Default Admin Policy” on page 3-23](#)

Security Roles

The security role defines a set of capabilities granted to users and groups based on specific conditions. Like groups, security roles allow you to restrict access to resources for several users at once. By using security roles, you limit what the user can do. Security roles differ from groups as follows:

- Security roles are computed and granted to users or groups dynamically, based on conditions such as user name, group membership, or the time of day. Groups are static.
- Security roles can be scoped to specific resources within a single application in a Security Service Module, unlike groups, which are always scoped to an entire domain.

Granting a security role to a user or a group confers the defined access privileges to that user or group, as long as the user or group is in the security role. For example, an administrator may define a security role called `AppAdmin`, which has write access to the resources of a particular web application. Any user or group granted the `AppAdmin` security role then has write access to that URL (Web) resource. Multiple users or groups can be granted a single security role and a user or group can belong to multiple roles.

Dynamic Role Mapping

At runtime, the Security Service compares users or groups against a role condition to determine whether they should be dynamically granted a security role. This process is referred to as **role mapping**, and occurs just prior to when the security service renders an access decision for a protected resource. An access decision is the component of an Authorization provider that determines whether a subject has permission to perform a given operation on a resource.

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

Understanding the Administration Policy

An administration policy is provided with the product to protect the Administration Application and its resources. This policy defines what tasks each role is allowed to perform within the console, which resources are protected, and how. While one user may be able to perform any and all tasks, you may want to allow other users to perform only a subset of those tasks. You can customize the administration policy by changing or replacing the roles and rules to suit your business needs. You may add users to each role or you may create your own roles with a new set of capabilities. You can customize the Admin policy and modify it to suit your needs.

Because you may have several different levels of administration (for example one administrator might manage resources and another might manage identities), you probably want to assign various tasks to different individuals.

The following security roles are defined by the administration policy:

- [“Admin Role” on page 3-3](#)
- [“Deployer Role” on page 3-5](#)
- [“Monitor Role” on page 3-6](#)
- [“Operator Role” on page 3-5](#)

Two additional roles, [“Anonymous Role” on page 3-6](#) and [“Everyone Role” on page 3-6](#) are also included, however, these have minimal rules written against them. The sections that follow describe each role and its capabilities. The capabilities of each role are not hard coded; they are determined by the policy rules. The rule set may be modified or completely replaced to change the capabilities as needed. Additionally, new roles may be created to provide other combinations of capabilities.

A user of the Administration Application is assigned to one or more of these roles to determine what capabilities they have. The only role assignment provided in the default policy is the assignment of the user `system` to the `Admin` role.

Policy data is represented in the console using a specific proprietary format that requires the use of qualifiers. For additional information on qualifiers and naming of policy elements, see [Securing Resources and Defining Policy Rules](#), in the *BEA WebLogic Enterprise Security Policy Managers Guide*.

Note: After you understand the Admin policy design, you can begin modifying it to suit your own needs, as described in [“Default Admin Policy” on page 3-23](#).

Admin Role

The Admin role has complete control over a set of policy elements:

- Create users and groups, privileges and privilege groups, resources, etc.
- View the server configuration, including the encrypted value of encrypted attributes
- Modify the entire server configuration
- Import and export policy data
- Install and deploy Security Service Modules
- Start, resume, and stop Administration Servers
- Run queries

Anyone belonging to this role has all capabilities, except the ability to write deny rules. Although BEA discourages the use of deny rules, you can change the policy to allow this capability.

To view a list users and groups who belong to the Admin role, click Role, and then click Admin. The Roles pane displays the following information.

Figure 3-1 Roles Pane for Admin

Name	User or Group	Resource	Role Condition
Admin	user/wles/system policy»WLES		none
Anonymous			
Deployer			
Everyone			
Monitor			
none			
Operator	user/wles/system policy»WLESRecovery»console		none

The Roles pane lists the rules assigning users and groups to roles. In this example, the user system is granted the Admin role for the root nodes of the Administration Application, WLES and WLESRecovery»»console. The privileges you define in your administration policy typically represent actions that you grant or deny. You grant or deny a privileges through security rules. Typically sets of privileges are granted to Roles, then Roles are granted to users and groups.

To view the policy rules associated with the Admin role, click Role, click Admin, and then click Policy Inquiry.

Figure 3-2 Policy Rules for Admin Role

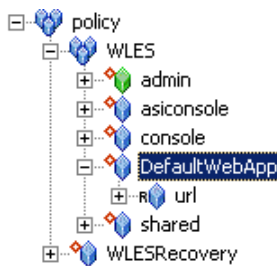
Privilege	Resource	Policy Conditions
GET	policy»WLES»DefaultWebApp	none
GET	policy»WLES»asiconsole	none
GET	policy»WLES»asiconsole»url»asi»audit	none
GET	policy»WLES»asiconsole»url»asi»enroll»authorizationcheck	none
GET	policy»WLES»console	none
POST	policy»WLES»DefaultWebApp	none
POST	policy»WLES»asiconsole	none
POST	policy»WLES»asiconsole»url»asi»audit	none
POST	policy»WLES»asiconsole»url»asi»enroll»authorizationcheck	none
POST	policy»WLES»console	none
addMember	policy»WLES»admin	none
bind	policy»WLES»admin»Infrastructure»Engines	none
cascadeDelete	policy»WLES»admin	none
copy	policy»WLES»admin»Identity»Subject»User	none

The Policy Inquiry page lists each resource that the user is allowed to access and the privilege (for example, GET, POST, addMember, bind), that applies to that resource. Each entry in the resulting table shows the name of a Privilege (What action can the Admin take?), a Resource (What resource can the Admin access?), and any Policy Conditions that apply to the policy rule (What conditions apply to the rule?).

Notice the use of Policy Conditions; `sys_user_q` is a built-in system attribute defined as the current user in the system group: `//user/wles/system/`. For additional information on built-in system attributes, see “[Securing Resources and Defining Policy Rules](#),” in the *BEA WebLogic Enterprise Security Policy Managers Guide*.

A Resource is defined as it appears in the resource tree. For example, the resource referred to as `policy>>WLES>>DefaultWebApp` is represented in the console as follows:

Figure 3-3 Representation of the WLES Resource



Here, the `DefaultWebApp` is a child node bound to its parent: `WLES`. In turn, this node has one child resource node defined as `url`, which represents the URL of the default web application being protected.

Deployer Role

The Deployer security role has more limited control over a set of policy elements:

- View the server configuration (except for encrypted attributes)
- View, modify and deploy configuration and policy data
- Run queries

No one is assigned to the Deployer role (as shown here). You may choose to add users to this role and modify the rules associated with it or you may create a new role that better suits your needs. To view the policy rules associated with this role, click Role, click Deployer, and then click Policy Inquiry.

Operator Role

The Operator security role has the following rights:

- View the server configuration, except for encrypted attributes

- Start the Administration Server
- Run queries

There are no users assigned to this role. You may choose to add users to this role and modify the rules associated with it or you may create a new role that better suits your needs.

Monitor Role

The Monitor security role has the following rights:

- Monitor the activities performed in the Administration Console
- View the server configuration, except for encrypted attributes

This security role effectively provides read-only access to the Administration Console. There are no users assigned to this role. You may choose to add users to this role and modify the rules associated with it or you may create a new role that better suits your needs.

Everyone Role

The Everyone security role is assigned to all authenticated and unauthenticated users (allusers). The Everyone role is limited to the following rights:

- Change their password
- View the login page
- Access unprotected resources and operations

You can assign privileges to allow additional capabilities. You may choose to add users to this role and modify the rules associated with it or you may create a new role that better suits your needs.

Anonymous Role

The Anonymous role has no rights and no users are assigned to this role. This role typically contains all unauthenticated users and allows access to only unprotected resources.

Resources

[Table 3-1](#) lists the resources protected by the Admin Policy with a description of each one. You can write rules based on these resources to control what privileges users have within the Administration Application. By default, the admin resources are all nested below the

//app/policy/WLES organization node. These resources are organized into a hierarchy to make writing policy simpler.

Table 3-1 Resources

Resource Name	Description
admin/Declaration/Attribute	Used to protect operations on attribute declarations.
admin/Declaration/Constant	Used to protect operations on constant declarations.
admin/Declaration/Enumeration	Used to protect operations on enumeration declarations.
admin/Declaration/EvaluationFunction	Used to protect operations on evaluation function declarations.
admin/Identity/Directory/Instance	Used to protect operations on identity directory instances.
admin/Identity/Directory/AttributeMapping/Single	Used to protect operations on what scalar attributes may be assigned to users within a directory.
admin/Identity/Directory/AttributeMapping/List	Used to protect operations on what vector attributes may be assigned to users within a directory.
admin/Identity/Subject/User	Used to protect operations on users.
admin/Identity/Subject/Group	Used to protect operations on groups.
admin/Identity/Subject/AttributeAssignment/Single	Used to protect operations on scalar subject attribute values.
admin/Identity/Subject/AttributeAssignment/List	Used to protect operations on vector subject attribute values.
admin/Identity/Subject/Password	Used to protect operations on user passwords.
admin/Resource/Instance	Used to protect operations on resources.
admin/Resource/AttributeAssignment/Single	Used to protect operations on scalar resource attribute values.
admin/Resource/AttributeAssignment/List	Used to protect operations on vector resource attribute values.

Table 3-1 Resources

admin/Resource/MetaData/LogicalName	Used to protect operations on setting the "logical name" resource metadata.
admin/Resource/MetaData/IsApplication	Used to protect operations on setting the "is application" resource metadata.
admin/Resource/MetaData/IsDistributionPoint	Used to protect operations on setting the "is distribution point" metadata.
admin/Policy/Rule/Grant	Used to protect operations on grant rules.
admin/Policy/Rule/Deny	Used to protect operations on deny rules.
admin/Policy/Rule/Delegate	Used to protect operations on delegate rules.
admin/Policy/Action/Role/Instance	Used to protect operations on roles (when used as actions).
admin/Policy/Action/Privilege/Instance	Used to protect operations on privileges.
admin/Policy/Action/Privilege/Group	Used to protect operations on privilege groups.
admin/Policy/Analysis/InquiryQuery	Used to protect operations on policy inquiries.
admin/Policy/Analysis/VerificationQuery	Used to protect operations on policy verification.
admin/Infrastructure/Engines/ARME	Used to protect operations on definitions of the Authorization and Role Mapping Engine (ARME).
admin/Infrastructure/Engines/SCM	Used to protect operations on definitions of the Service Control Manager (SCM).
admin/Infrastructure/Management/Loader	Used to protect operations on the policy loader.
admin/Policy/Repository	Used to protect operations on the policy repository.

Privileges

[Table 3-2](#) lists the privileges that apply to the security roles and describes each one. The `view` privilege is required to see the contents of an instance of a policy element, for example, to see the value of an attribute. The `listAll` privilege is needed to list all the instances of a particular type of policy element, for example, to see all the users in a directory. You may want to think of this as the difference between being able to open a file and to list a file in a directory. You can use

these privileges in rules to control what operations administrators of the Administration Application may perform.

Table 3-2 Privileges

Privilege	Explanation
create	Create a policy element, including identities, directories, users, groups, attributes), resources and their attributes, configuration data and their bindings, and privileges and privilege groups.
view	View the contents of a policy element, including identities (identity directories, users, groups, identity attributes), resources and their attributes, configuration data and their bindings, privileges and privilege groups.
delete	Delete a policy element, including identities (identity directories, users, groups, identity attributes), resources and their attributes, configuration data and their bindings, and privileges and privilege groups.
cascadeDelete	Delete an element and its sub-elements (no permission check is made on sub-elements), including identities (identity directories, users, groups, identity attributes), resources and their attributes, configuration data and their bindings, and privileges and privilege groups.
rename	Rename a policy element, including identities (identity directories, users, groups, identity attributes), resources and their attributes, configuration data and their bindings, and privileges and privilege groups.
modify	Modify the contents of a policy element, including identities (identity directories, users, groups, identity attributes), resources and their attributes, configuration data and their bindings, and privileges and privilege groups.
listAll	Filter lists of instances based on a pattern specification.
addMember	Add a member to a group.
removeMember	Remove a member from a group.
execute	Execute a policy analysis query.
deployUpdate	Deploy a policy update.

Table 3-2 Privileges (Continued)

Privilege	Explanation
deployStructuralChange	Deploy a structural change.
bind	Bind a resource to an ASI Authorization and ASI Role Mapping provider.
unbind	Unbind a resource from an ASI Authorization and ASI Role Mapping provider.
login	Log on to the Administration Application, including the Administration Console, and the Policy Import and Export tools.
copy	Copy a policy element, including identities (identity directories, users, groups, identity attributes), resources and their attributes, configuration data and their bindings, and privileges and privilege groups.

Context Attributes

When the Administration Application performs an authorization check, contextual data is provided to allow for fine-grained protection of policy operations. For example, when creating a privilege, the name of the privilege is supplied as an attribute, enabling you to control access to a single unique privilege and to all privileges.

The following attributes are used by the Administration Application; additionally, you may use any of the standard built-in attributes which are always available during authorization.

Table 3-3 Context Attributes

Attribute Name	Data Type	Description
declaration	string	Name of a declaration.
data_type	string	The name of a data type, for example, a string, integer, date.
attribute_usage_type	Enumeration (resource_attribute, subject_attribute, dynamic_attribute)	Specifies the type of policy element with which an attribute declaration is associated.

Table 3-3 Context Attributes

new_name	string	Generic attribute used when renaming elements.
new_attribute_usage_type	Enumeration (resource_attribute, subject_attribute, dynamic_attribute)	The new value for this item used to modify operations.
value	string	Generic attribute used to represent the value of an element.
values	list of strings	Generic attribute used to represent the value of an element as a list.
directory	string	The name of a directory.
attribute	string	The name of an attribute.
default_value	string	The default value of an attribute.
default_values	list of strings	The default value of a list attribute.
new_default_value	string	Used in modification operations to represent the new default value of an attribute value.
new_default_values	list of strings	Used in modification operations to represent the new default value of a list attribute.
subject_name	string	The name of a subject.
subjects	list of strings	A list of subjects.
groups	list of strings	The group membership of the subject.
subject_type	Enumeration (user_subject, group_subject, role_subject)	The type of subject.

Table 3-3 Context Attributes

member_subject_type	Enumeration (user_subject, group_subject, role_subject)	The type of the subject group member.
member_subject	string	Name of subject group member.
action	string	Name of the action.
action_type	Enumeration (privilege_action, role_action)	Type of the action.
resource	string	The name of the resource.
resources	list of strings	A list of resources.
constraint	string	The constraint of a rule; this is the portion between the 'if' and ';' exclusive.
new_action	string	Name of new action in a modified rule.
new_action_type	string	New action type in a modified rule.
new_resource	string	New resource in a modified rule.
new_subject_name	string	New subject name.
new_constraint	string	New constraint in a modified rule.
delegator	string	The name of the delegator in a rule.
new_delegator	string	New delegator in a modified rule.
actions	list of strings	A set of actions.
action_groups	list of strings	A list of privilege group names.
action_group	string	The name of a privilege group.

Table 3-3 Context Attributes

parent_resource	string	The parent of the resource.
meta_data	string	The name of the metadata item.
logical_name	string	The logical name of a resource.
deleted_directories	list of strings	A list of deleted directories.
deleted_engines	list of strings	A list of deleted engines. ¹
deployed_engines	list of strings	A list of deployed engines.
deleted_bindings	list of strings	A list of deleted engine binding node pairs.
deleted_applications	list of strings	A list of deleted applications.
engine	string	The name of an ARME or SCM cluster.
engine_bindings	list of strings	A list of bindable resources bound to the ARME or SCM.
owner	string	The owner of analysis query.
effect_type	Enumeration (grant_effect, deny_effect, delegate_effect)	The type of rule effect.
title	string	The title of a analysis query.

1. The term engine refers to an ASI Authorization and ASI Role Mapper provider that are configured to operate in conjunction with one another. This combination of providers are configured to manage your authorization policy.

Evaluation Functions

The following evaluation functions are provided to help you write custom administration policies. They may be used in the constraint portion of rules to limit the applicability of the rule based on contextual information.

Table 3-4 Evaluation Functions

Function Name	Description
resource_is_child(c,p,[d])	Check if c a child of p. d is a Boolean standing for direct. By default, d is true, meaning check if c is directly a child of p. If false, then c may be a descendant of p at any depth.
subject_in_directory(s,d)	Check if subject s is in directory d. This does not guarantee that either s or d exists, only that based on the name one would be in the other.
subject_is_group(s) subject_is_user(s) subject_is_role(s)	Check if the subject of a user group or role.
action_is_privilege(a) action_is_role(a)	Check if the action is a privilege or role

Authorization Queries

Now that you have an understanding of the elements that make up the administration policy, it is important to understand when the administration system performs authorization queries and what contextual attribute data is supplied with that query. This is the data that you may reference when writing rules to protect the Administration Application.

Table 3-5 Authorization Queries

Admin Resource	Privilege	Context attributes	Description
Declaration/Attribute	create	declaration	Queried when user attempts to create a new attribute declaration.
	delete	declaration	Queried when user attempts to delete an attribute declaration.
	rename	declaration, new_name	Queried when user attempts to rename an attribute declaration.

Table 3-5 Authorization Queries

	modify	declaration	Queried when user attempts to modify an attribute declaration.
Declaration/ Constant	create	declaration, value	Queried when user attempts to create a new constant.
	delete	declaration, value	Queried when user attempts to delete a constant.
	rename	declaration, value, new_name	Queried when user attempts to rename a constant.
	modify	declaration, value, new_value	Queried when user attempts to modify a constant.
Declaration/ Enumeration	create	declaration, value	Queried when user attempts to create a new enumeration.
	delete	declaration, value	Queried when user attempts to delete an enumeration.
	rename	declaration, value, new_name	Queried when user attempts to rename an enumeration.
	modify	declaration, value, new_value	Queried when user attempts to modify an enumeration.
Declaration/Evaluation Function	create	declaration	Queried when user attempts to create an evaluation function.
	delete	declaration	Queried when user attempts to delete an evaluation function.
	rename	declaration, new_name	Queried when user attempts to rename an evaluation function.
Identity/Directory/ Instance	create	directory	Queried when user attempts to create a directory.
	delete	directory	Queried when user attempts to delete a directory.
	cascade Delete	directory	Queried when user attempts to delete a directory and all its users.

Table 3-5 Authorization Queries

	rename	directory, new_name	Queried when user attempts to rename a directory.
Identity/Directory/ AttributeMapping/ Single	create	attribute, default_value, directory	Queried when user attempts to add a scalar attribute to an attribute schema of a directory.
	delete	attribute, default_value, directory	Queried when user attempts to delete a scalar attribute from an attribute schema of a directory.
	modify	attribute, default_value, directory, new_default_value	Queried when user attempts to modify a scalar attribute in an attribute schema for a directory.
Identity/Directory/ AttributeMapping/ List	create	attribute, default_value, directory	Queried when user attempts to add a vector attribute to an attribute schema of a directory.
	delete	attribute, default_value directory	Queried when user attempts to delete a vector attribute from an attribute schema of a directory.
	modify	attribute, default_value, directory, new_default_value	Queried when user attempts to modify a vector attribute in an attribute schema of a directory.
Identity/Subject/User	create	subject_name	Queried when user attempts to create a new user.
	copy	subject_name, new_subject_name	Queried when user attempts to copy a user.
	delete	subject_name	Queried when user attempts to delete a user.
	cascade Delete	subject_name	Queried when user attempts to cascade a user and all rules associated with the user.
	rename	subject_name, new_subject_name	Queried when user attempts to rename a user.
Identity/Subject/Group	create	subject_name	Queried when user attempts to create a new group.

Table 3-5 Authorization Queries

	delete	subject_name	Queried when user attempts to delete a group.
	rename	subject_name, new_subject_name	Queried when user attempts to rename a group.
	addMember	subject_name, member_subject	Queried when user attempts to add a member to a group.
	remove Member	subject_name, member_subject	Queried when user attempts to remove a member from a group.
Identity/Subject/ Attribute Assignment/ Single	create	attribute, value, subject_name	Queried when user attempts to set a value to a currently unset scalar subject attribute.
	delete	attribute, value, subject_name	Queried when user attempts to unset a currently set scalar subject attribute.
	modify	attribute, value, subject_name, new_value	Queried when user attempts to modify the value of a currently set scalar subject attribute.
Identity/Subject/ AttributeAssignment/ List	create	attribute, value, subject_name	Queried when user attempts to set a value to a currently unset vector subject attribute.
	delete	attribute, value, subject_name	Queried when user attempts to unset a currently set vector subject attribute.
	modify	attribute, value, subject_name, new_value	Queried when user attempts to modify the value of a currently set vector subject attribute.
Identity/Subject/ Password	modify	subject_name	Queried when user attempts to modify the password for a user. The subject_name attribute contains the name of the user for which the password is associated.
Resource/Instance	create	resource, resource_type	Queried when user attempts to create a new resource.
	delete	resource	Queried when user attempts to delete a resource.

Table 3-5 Authorization Queries

	cascade Delete	resource	Queried when user attempts to cascade delete a resource. This includes deletion of all child resources and associated rules.
	rename	resource, new_name	Queried when user attempts to rename a resource.
Resource/Attribute Assignment/ Single	create	attribute, resource, value	Queried when user attempts to set a value to a currently unset scalar resource attribute.
	delete	attribute, resource, value	Queried when user attempts to unset a currently set scalar resource attribute.
	modify	attribute, resource, value, new_value	Queried when user attempts to modify the value of a currently set scalar resource attribute.
Resource/Attribute Assignment/ List	create	attribute, resource, value	Queried when user attempts to set a value to a currently unset vector resource attribute.
	delete	attribute, resource, value	Queried when user attempts to unset a currently set vector resource attribute.
	modify	attribute, resource, value, new_value	Queried when user attempts to modify the value of a currently set vector resource attribute.
Resource/MetaData/ IsApplication	modify	resource, value, new_value	Queried when user attempts to toggle the “is application” resource metadata.
Resource/MetaData/ IsDistributionPoint	modify	resource, value, new_value	Queried when user attempts to toggle the “is distribution point” resource metadata.
Resource/MetaData/ Logical Name	create	logical_name, resource	Queried when user attempts to create a logical name for a resource.
	delete	logical_name, resource	Queried when user attempts to delete a logical name for a resource.

Table 3-5 Authorization Queries

	rename	logical_name, resource, new_name	Queried when user attempts to rename a logical name for a resource.
Policy/Rule/Grant	create	action, resource, subject_name, constraint	Queried when user attempts to create a new grant rule. “action”, “resource”, and “subject_name” attributes are lists.
	delete	action, resource, subject_name, constraint	Queried when user attempts to delete a grant rule. The “action”, “resource”, and “subject_name” attributes are lists.
	modify	action, resource, subject_name, constraint, new_action, new_resource, new_subject_name, new_constraint	Queried when user attempts to modify a grant rule. The “action”, “resource”, and “subject_name” attributes are lists.
Policy/Rule/Deny	create	action, resource, subject_name, constraint	Queried when user attempts to create a new deny rule. “action”, “resource”, and “subject_name” attributes are lists.
	delete	action, resource, subject_name, constraint	Queried when user attempts to delete a deny rule. The “action”, “resource”, and “subject_name” attributes are lists.
	modify	action, action_type, resource, subject_name, subject_type, constraint, new_effect, new_action, new_action_type, new_resource, new_subject_name, new_subject_type, new_constraint	Queried when user attempts to modify a deny rule. The “action”, “resource”, and “subject_name” attributes are lists.
Policy/Rule/Delegate	create	action, resource, subject_name, delegator, constraint	Queried when user attempts to create a new delegate rule. “action”, “resource”, and “subject_name” attributes are lists.

Table 3-5 Authorization Queries

	delete	action, resource, subject_name, delegator, constraint	Queried when user attempts to delete a delegate rule. The “action”, “resource”, and “subject_name” attributes are lists.
	modify	action, resource, subject_name, delegator, constraint, new_action, new_resource, new_subject_name, new_delegator, new_constraint	Queried when user attempts to modify a delegate rule. The “action”, “resource”, and “subject_name” attributes are lists.
Policy/Action/Role/Instance	create	action	Queried when user attempts to create a new role.
	delete	action	Queried when user attempts to delete a role.
	rename	action, new_name	Queried when user attempts to rename a role.
Policy/Action/Privilege/Instance	create	action	Queried when user attempts to create a privilege.
	delete	action	Queried when user attempts to delete a privilege.
	rename	action, new_name	Queried when user attempts to rename a privilege.
Policy/Action/Privilege/Group	create	action_group	Queried when user attempts to create a privilege group.
	delete	action_group	Queried when user attempts to delete a privilege group.
	rename	action_group, new_name	Queried when user attempts to rename a privilege group.
	addMember	action_group, action	Queried when user attempts to add a privilege to a privilege group.

Table 3-5 Authorization Queries

	remove Member	action_group, action	Queried when user attempts to remove a privilege from a privilege group.
Policy/Analysis/ Inquiry Query	create	title, owner, effect_type, subjects, actions, resources, delegator	Queried when user attempts to create a new policy query.
	delete	title, owner	Queried when user attempts to delete a policy query.
	modify	title, owner, effect_type, subjects, actions, resources, delegator	Queried when user attempts to modify a policy query.
	execute	title, owner, effect_type, subjects, actions, resources, delegator	Queried when user attempts to execute a policy query. If this is an unsaved query “title” and “owner” will be set to an emptystring.
Policy/Analysis/ Verification Query	create	title, owner, actions, resources	Queried when user attempts to create a new policy verification query.
	delete	title, owner	Queried when user attempts to delete a policy verification query.
	modify	title, owner, actions, resources	Queried when user attempts to modify a policy verification query.
	execute	title, owner, actions, resources	Queried when user attempts to execute a policy verification query. If this is an unsaved query “title” and “owner” will be set to an emptystring.
Policy/Repository	deploy Update	resource, directory	Queried when user attempts to deploy a policy update. “resource” is the distribution node and all nodes below it may be effected. This check is made for each chosen distribution point.

Table 3-5 Authorization Queries

	deploy Structural Change	deleted_directories, deployed_engines, deleted_engines, deleted_bindings, deleted_applications	Queried when user attempts to deploy a structural change.
Infrastructure/Engines/A RME	create	engine	Queried when user attempts to create a new Security Service Module.
	delete	engine	Queried when user attempts to delete a Security Service Module.
	rename	engine, new_name	Queried when user attempts to rename a Security Service Module.
	bind	engine, resource	Queried when user attempts to bind a resource to a Security Service Module.
	unbind	engine, resource	Queried when user attempts to unbind a resource from a Security Service Module.
Infrastructure/Engines/S CM	create	engine	Queried when user attempts to create a Service Control Manager.
	delete	engine	Queried when user attempts to delete a Service Control Manager.
	rename	engine, new_name	Queried when user attempts to rename a Service Control Manager.
	bind	engine, resource	Queried when user attempts to bind a Security Service Module to a Service Control Manager. The “resource” contains the name of the Security Service Module.
	unbind	engine, resource	Queried when user attempts to unbind a Security Service Module from a Service Control Manager. The “resource” contains the name of the Security Service Module.

Table 3-5 Authorization Queries

Infrastructure/ Management/Console	login		Queried when user attempts to login to the Administration Console.
Infrastructure/ Management/Loader	login		Queried when user attempts to login to the Policy Import tool.

Enumerated Types

[Table 3-6](#) lists the name of each enumerated type used in the Admin policy.

Table 3-6 Enumerated Types

Name	Values	Description
attribute_usage_type_enum	(resource_attribute, subject_attribute, dynamic_attribute)	Specifies the valid usage for attributes.
subject_type_enum	(user_subject, group_subject, role_subject)	Specifies the valid subject types.
action_type_enum	(privilege_action, role_action)	Specifies the valid action types.
resource_type_enum	(organizational_resource, bindable_resource, component_resource)	Specifies the valid resource types. These names may need some work, they all seem to have multiple names. (open issue).
effect_type_enum	(grant_effect, deny_effect, delegate_effect)	Specifies the valid effect types.

Default Admin Policy

Rules define capabilities that ultimately control what operations a user is allowed to perform within the Administration Application. The admin policy provided with the product defines a default set of capabilities.

[Table 3-7](#) lists and describes the rules included in the default admin policy.

Table 3-7 Default Admin Policy Rules

Default Rule	Description
<code>grant(//priv/delete, //app/policy/WLES/admin, //role/Admin) if true;</code>	Grants members of the Admin role the ability to delete any policy element:
<code>grant(//priv/cascadeDelete, //app/policy/WLES/admin, //role/Admin) if true;</code>	Grants members of the Admin role the ability to perform the cascadeDelete operation on any policy element. Specifically, cascadeDelete applies to deletion of directories and resource hierarchies.
<code>grant(//priv/rename, //app/policy/WLES/admin, //role/Admin) if true;</code>	Grants members of the Admin role the ability to rename any policy element:
<code>grant(//priv/deployStructuralChange, //app/policy/WLES/admin/Policy/Repository, //role/Admin) if true;</code>	Grants members of the Admin role the ability to deploy structural changes:
<code>grant(//priv/login, //app/policy/WLES/admin/Infrastructure/ Management/Loader, //role/Admin) if true;</code>	Grants members of the Admin role the ability to use the policy loader tool:
<code>grant(//priv/copy, //app/policy/WLES/admin/Identity/ Subject/User, //role/Admin) if true;</code>	Grants members of the Admin role the ability to copy any policy element:
<code>grant([//priv/bind, //priv/unbind], //app/policy/WLES/admin/Infrastructure/ Engines, //role/Admin) if true;</code>	Grants members of the Admin role the ability to bind and unbind resources and configuration to ARMEs and SCMs respectively:
<code>grant(//priv/deployUpdate, //app/policy/WLES/admin/Policy/Repository, [//role/Admin, //role/Deployer]) if true;</code>	Grants members of the Admin and Deployer roles the ability to deploy policy updates:
<code>grant(//priv/modify, //app/policy/WLES/admin, [//role/Admin, //role/Deployer]) if true;</code>	Grants members of the Admin and Deployer roles the ability to modify any policy element:

Table 3-7 Default Admin Policy Rules (Continued)

Default Rule	Description
<code>grant(//priv/view, //app/policy/WLES/admin, [//role/Admin,//role/Monitor, //role/Operator,//role/Deployer]) if true;</code>	Grants members of the Admin, Monitor, Operator, and Deployer roles the ability to view any policy element:
<code>grant(//priv/listAll, //app/policy/WLES/admin, [//role/Admin,//role/Monitor, //role/Operator,//role/Deployer]) if true;</code>	Grants members of the Admin, Monitor, Operator, and Deployer roles the ability to perform the listAll operation on any policy element:
<code>grant(//priv/modify, //app/policy/WLES/admin/Identity/Subject/ Password, //role/Everyone) if subject_name = sys_user_q;</code>	Grants members of the Everyone role the ability to modify their own password. Notice that members of Admin and Deployer roles can modify the password for any user.
<code>grant(//priv/create, [//app/policy/WLES/admin/Declaration, //app/policy/WLES/admin/Identity, //app/policy/WLES/admin/Infrastructure, //app/policy/WLES/admin/Resource], //role/Admin) if true;</code> <code>grant(//priv/create, [//app/policy/WLES/admin/Policy/Action, //app/policy/WLES/admin/Policy/Analysis, //app/policy/WLES/admin/Policy/Rule/ Delegate, //app/policy/WLES/admin/Policy/Rule/Grant], //role/Admin) if true;</code>	Grants members of the Admin role the ability to create most policy elements. The only type of policy element not included in this list is the deny: //app/policy/WLES/admin/Policy/Rule/Deny which denies the Admin role the ability to create deny rules.
<code>grant([//priv/create,//priv/modify, //priv/view], //app/policy/WLES/admin/Policy/Analysis, [//role/Admin,//role/Monitor, //role/Operator,//role/Deployer]) if owner = sys_user_q;</code>	Grant members of the Admin, Monitor, Operator and Deployer roles the ability to create, modify, and view policy analysis queries of which they are the owner:
<code>grant(//priv/execute, //app/policy/WLES/admin/Policy/Analysis, [//role/Admin,//role/Monitor, //role/Operator,//role/Deployer]) if owner = sys_user_q or owner = "";</code>	Grants members of the Admin, Monitor, Operator and Deployer roles the ability to execute policy analysis queries of which they are the owner of, or of which have no owner:

Table 3-7 Default Admin Policy Rules (Continued)

Default Rule	Description
<code>grant([//priv/addMember,//priv/removeMember], //app/policy/WLES/admin, [//role/Admin,//role/Deployer]) if true;</code>	Grants members of the Admin and Deployer roles the ability to add and remove members to subject groups and privilege groups.
<code>grant(//role/Everyone, //app/policy/WLES, //sgrp/wles/allusers/) if true;</code>	Assigns all members in the WLES directory into the Everyone role.
<code>grant(//role/Admin, //app/policy/WLES, //user/wles/system/) if true;</code>	Assigns the system user into the Admin role.
<code>grant(//role/Anonymous, //app/policy/WLES, //user/wles/anonymous/);</code>	Assigns anonymous users the Anonymous role. Note: All unauthenticated users are members of the Anonymous role.

Example Policy Customizations

The default admin policy is intended to be customized to meet an individual customers needs. Because the dynamic roles may be granted contextually, you may make a user a member of a role within a limited scope. For example, you can make user Joe a member of the Admin role, but only over resources.

```
grant(//role/Admin, //app/policy/WLES/admin/Resource, //user/wles/Joe/) if true;
```

This allows Joe to act as an Administrator over resources, but does not give him rights to write policy.

To make your rule more explicit, you can let user Bob be an Administrator when modifying any policy element for a certain application. With an application with a resource rooted at `//app/policy/PetStore`, you can define the following rule.

```
grant(//role/Admin, //app/policy/WLES/admin, //user/wles/Bob/) if sys_defined(resource) and resource_is_child(resource, //app/policy/PetStore, no);
```

This allows Bob to act in the Admin role for any admin policy query involving the `PetStore`.

Now that you feel you understand the basic rules involved in constructing the Admin Policy, see [“Securing Resources and Defining Policy Rules,”](#) in the *BEA WebLogic Enterprise Security Policy Managers Guide* for additional information on how to implement rules.

Security Administration

Managing Security

You manage security by using the Administration Console to configure Service Control Managers, Security Service Modules and their providers, and to design, edit, and distribute security policies. A policy is composed of a set of rules, modeled on your organizational business requirements that state who has access to which resources.

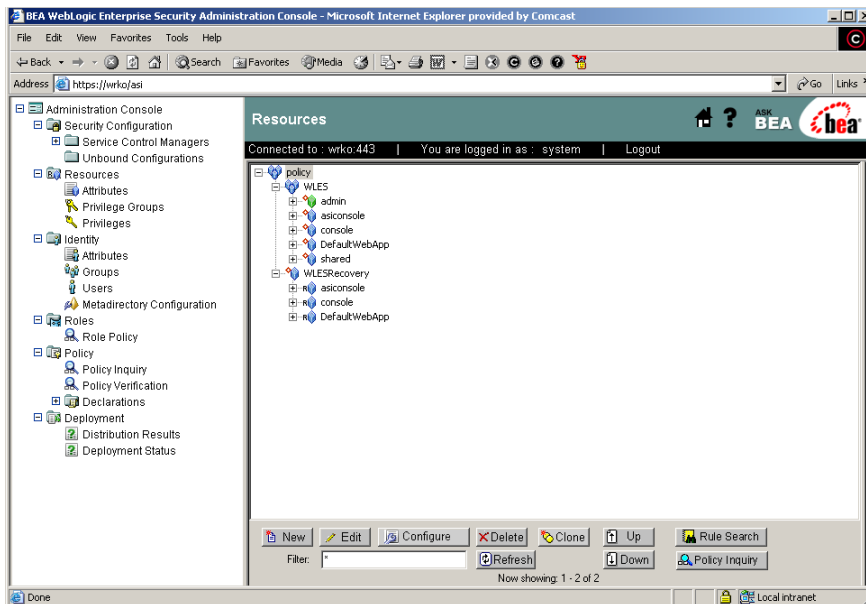
The rules that define your security policy are built from policy elements: identity (users and groups with their associated attributes), privileges, roles, and the resources you want to protect. A policy can be applied to either a single resource or to many resources across the enterprise. An enterprise-wide policy represents the superset of all of your security policies. The following sections describe security management concepts and how you implement security using the Administration Console:

- [Security Configuration](#)—Service Control Managers, Security Service Modules, and provider configuration
- [Resources](#)—Privileges, privilege groups and resource attributes
- [Identity](#)—Users and groups, and identity attributes
- [Role](#)—Rule-based role assignment
- [Policy](#)—Rule-based access control
- [Deployment](#)—Distribution of policy and security configuration

<~runChNum>

Figure 4-1 shows how the Administration Console represents the various policy elements and resources. The resources shown here represent the initial policy settings used to protect the console. The two resources that are protected are `WLES` and `WLESRecovery` and their various components.

Figure 4-1 Administration Console Resource Representation



Security Configuration



A Security Configuration is represented by the  icon in the Administration Console, and consists of one or more Service Control Managers, one or more Security Service Modules represented by the lock  icon, and the providers associated with each module. You can use the security providers that are provided, purchase custom security providers from third-party security vendors, or develop your own custom security providers. Table 4-1 shows how each provider type is represented in the Administration Console.

Table 4-1 Security Providers and their Representations

	Adjudication		Authorization
	Auditing		Credential Mapping
	Authentication		Role Mapping

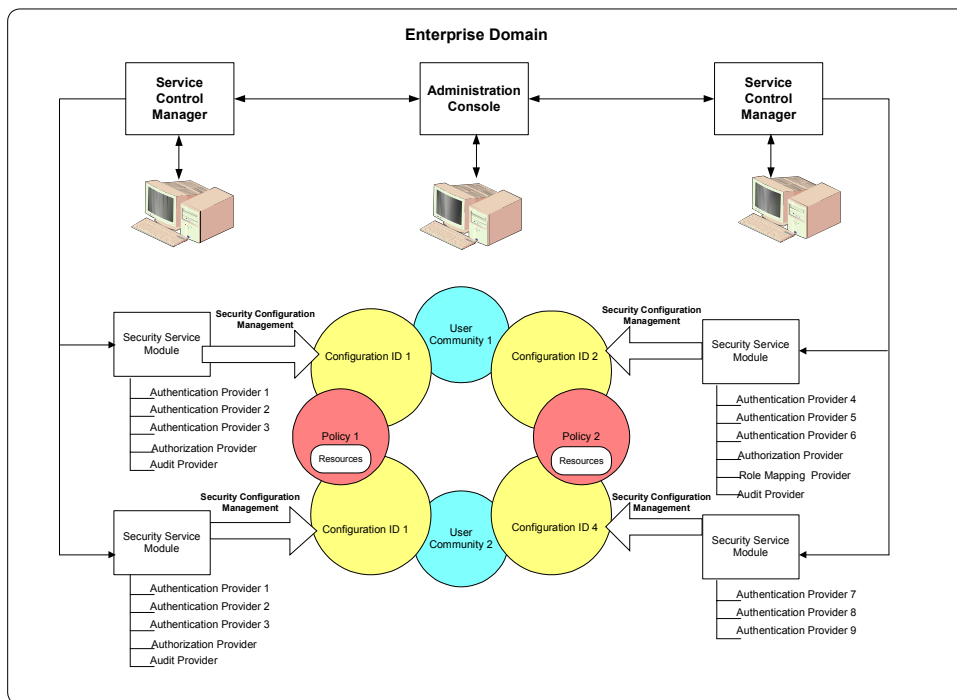
To learn more about security providers and their services, see [Security Services](#) in the *Introduction to WebLogic Enterprise Security*. For information on how to develop custom security providers, see [Developing Security Providers for WebLogic Enterprise Security](#).

A configuration is a named collection of elements and security services over which a common and consistent set of policies (or rules) are administered in a coordinated fashion. Each configuration has a unique Configuration ID and is bound to a Service Control Manager. Applications that use the same Configuration ID reference the same user communities, policy, and security configuration.

A configuration also defines the method of authentication and authorization used to provide access to a resource for one or more users and groups. The authorization policy protects the resource against unauthorized access. A resource has no protection until you create and apply a policy. Each Security Service Module has a Configuration ID associated with it that defines the security providers, policy, and settings for that Security Service Module.

In [Figure 4-2](#), two Security Service Modules are installed on each of two machines, with all modules and configurations managed centrally by one Administration Application. The Service Control Manager provides configuration to each Security Service Module as needed, any time a change in configuration is required. Notice that Security Service Modules on the left use the same set of providers and share a set of policies, resources, and policy domains. The Security Service Modules on the right use a similar set of policies, resources, and policy domains, but use a different set of providers.


Figure 4-2 Policies and Security Configuration





Resources


A resource is a general term that refers to an entity or group of entities that you can protect. Resources can include applications, data, or system components. Resources may also include background services with which the user has no direct interaction. For example, a bank might offer banking services on a web page that simulate the actions of a teller. The components (buttons, tables, data fields) displayed on that web page are all resources. The Administration Console displays resources in a tree structure called a resource hierarchy as shown in [Figure 4-3](#). The resources shown here represent the administration services and policy data elements that are accessible to the Administration Application on startup.

A page generated from a JSP is also a resource. The page can call EJBs or COM resources to execute a transaction. The back office services that transfer money between accounts, issue a payment, or run a report are also resources, although they may not appear on the web page or execute on the application server.

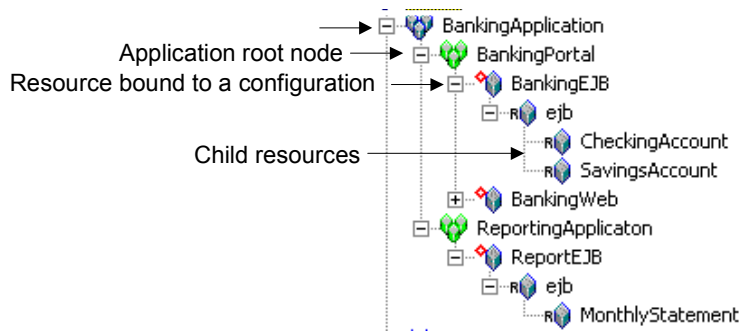
Individual resources in the hierarchy are also called nodes and the type of node can convey additional information about the resource. Some nodes are organizational because they represent a logical grouping of resources. For example, an organizational node called accounting may represent all resources in the accounting department. A node representing a group of resources is represented by a blue resource icon .

Another type of organizational node is an application node represented by a green resource icon . This icon depicts a collection of resources that provide a specific set of services. Applications are considered organizational because they often consist of multiple, distributed components, such as an n -tier web application that may have resources on a web server, an application server, and a data server. However, an application node can also be used to represent an application with a single component, such as a desktop spreadsheet application.

Individual resource nodes are represented by the  icon, and typically represent an individual resource that can be protected. An administrator may define as many resources and levels in the hierarchy as needed to represent data, services and system components within an application.

A special node, called a binding node, is depicted by a resource or application node with a small red diamond . A binding node is used to associate (or bind) a subset of the resource tree to a particular configuration.


Any resource at or above a binding node can be marked as a policy distribution point. When a policy distribution is initiated, you can choose to distribute either all updates (by selecting the root node) or you can limit which updates are distributed by selecting resources from the existing distribution points. Only updates that were made at or below the selected distribution points are distributed. [Figure 4-3](#) shows how organizations, applications and resources are represented and bound to Security Service Modules in the Administration Console.

Figure 4-3 Organizations, Applications, and Resources form your Security Configuration

Some typical resources that you might want to secure, include:


- An application, an application window, or a dialog box
- Specific business transactions, such as a money transfer or security trade
- Application controls, such as buttons and menu selections
- Database or directory server structures
- Web pages (URLs), servlets, and Enterprise Java Beans (EJB)
- Products or services through BEA WebLogic Portal

Resource Attribute

A resource attribute is represented by the  icon in the Administration Console. All resources can have attributes, which store information about the resources to which they belong. Common resource attributes might be a file type, resource owner, or the creation date.






















Attributes are inherited by child resources from their parent. If a resource explicitly sets the value for an attribute, the inherited value overrides the resource value.

Privilege

A privilege is represented by the key  icon in the Administration Console and is an action that you can perform on a resource. For instance, execute is a typical application privilege; and read and write are typical file-system privileges.

You can use the privileges provided or you can create your own. [Figure 4-4](#) shows how privileges appear in the Administration Console. Notice that each privilege refers to an action. A related collection of privileges may be organized into a privilege group for management purposes.

Figure 4-4 Privilege Representation in the Administration Console

Name	Last Modified Date	Modified By
 any	10/07/03 19:33:13	root
 GET	10/07/03 19:37:04	//user/wles/system/
 POST	10/07/03 19:37:04	//user/wles/system/
 addMember	10/07/03 19:36:24	//user/wles/system/
 bind	10/07/03 19:36:24	//user/wles/system/
 cascadeDelete	10/07/03 19:36:24	//user/wles/system/
 copy	10/07/03 19:36:24	//user/wles/system/
 create	10/07/03 19:36:24	//user/wles/system/
 delete	10/07/03 19:36:24	//user/wles/system/
 deployStructuralChange	10/07/03 19:36:24	//user/wles/system/
 deployUpdate	10/07/03 19:36:24	//user/wles/system/
 execute	10/07/03 19:36:24	//user/wles/system/
 listAll	10/07/03 19:36:24	//user/wles/system/
 login	10/07/03 19:36:24	//user/wles/system/
 modify	10/07/03 19:36:24	//user/wles/system/
 removeMember	10/07/03 19:36:24	//user/wles/system/
 rename	10/07/03 19:36:24	//user/wles/system/
 unbind	10/07/03 19:36:24	//user/wles/system/
 view	10/07/03 19:36:24	//user/wles/system/
 writeEvent	10/07/03 19:37:04	//user/wles/system/
 writeEvents	10/07/03 19:37:04	//user/wles/system/

Privilege Group

























A privilege group is represented by the keys  icon in the Administration Console and allows you to organize privileges into logical groups for ease of management. For example, it is common to define a privilege group that applies to a particular application or set of transactions. Privilege groups can be used as filters when constructing rules, although they cannot appear directly in the rule. [Figure 4-5](#) shows an example of how privilege groups and their associated privileges appear in the Administration Console.

Figure 4-5 Privilege Group Representation in the Administration Console

Name	Last Modified Date	Modified By
 ALL	10/07/03 19:33:13	root

Group's Privileges
 any
 GET
 POST
 addMember
 bind
 cascadeDelete
 copy
 create
 delete
 deployStructuralChange
 deployUpdate
 execute
 listAll
 login
 modify
 removeMember
 rename
 unbind
 view
 writeEvent
 writeEvents



Identity

The Identity  icon represents your directories and user communities in the Administration Console. Although BEA WebLogic Enterprise Security provides tools to manage users and groups locally, they are typically managed through an external repository, such as a Lightweight Directory Access Protocol (LDAP) directory server or a network database. A virtual view of identity data can be created through the replication and synchronization of the data using the metadirectory tools. User and group information, along with any attributes, is then stored as metadata in the policy database and is then available for viewing directly through the Administration Console.

Note: Identity data are used to calculate roles used in your authorization policy and is not used for authentication purposes. Authentication is supported through your external repositories by configuring an authentication provider.


A directory typically represents groups of users of a particular application or resource, or users in a specific location. Each directory has an associated attribute schema. The schema defines the attributes applied to members of the directory. [Figure 4-6](#) shows how directories are represented in the Administration Console. In this example, there are two directories: wles and Employees. The Employees directory shows the attributes that are stored for each member of the directory: zipcode, state, name, city and address. The state attribute has a default value set to MA (Massachusetts).

Figure 4-6 Directory Representation in the Administration Console

Name				
	wles			
	Employees			
		Attribute	Type	Default Value
		zipcode	integer	
		state	string	"MA"
		name	string	
		city	string	
		address	string	





































The number of directories you have depends on the level of granularity needed to separate your user community. You may want to have one global directory containing all users. In this case, you can populate a single directory using multiple external repositories. Or, you might want separate directories—one directory for customers, one for employees, and one for partners, where the method of authentication is different for each group. Having one directory requires a unique name for each user and group. If you are not able to guarantee this when you integrate your repositories, you should maintain separate directories. For information on integrating with external identity repositories, see the [Administration Application Installation Guide](#).

User


A user is represented by the  icon in the Administration Console and corresponds to an individual who makes a request to access a resource, although a user can be an automated process that accesses the system. Users are included in an authorization policy by assigning users to groups, and then assigning that group to a role. Each user within a directory must have a unique identity or user name.

Users can be associated with certain characteristics, referred to as identity attributes; these attributes store information about the user. The list of attributes that can be set for a user is dictated by the attribute schema of the directory to which the user belongs. [Figure 4-7](#) shows an example of a user representation with identity attributes. In this example, `userid1000` belongs to four groups and there are nine attributes associated with the user.

Figure 4-7 User Representation in the Administration Console

Name	Group Membership
 <code>userid1000</code>	 <code>myrole</code>
 <code>userid1001</code>	 <code>subjectgroup19</code>
 <code>userid1002</code>	 <code>subjectgroup28</code>
 <code>userid1003</code>	 <code>subjectgroup5</code>
 <code>userid1004</code>	
 <code>userid1005</code>	
 <code>userid1006</code>	
 <code>userid1007</code>	
 <code>userid1008</code>	
 <code>userid1009</code>	
 <code>userid101</code>	
 <code>userid1010</code>	
 <code>userid1011</code>	
 <code>userid1012</code>	
 <code>userid1013</code>	
 <code>userid1015</code>	
 <code>userid1016</code>	
 <code>userid1017</code>	
 <code>userid1018</code>	
 <code>userid1020</code>	
 <code>userid1021</code>	
 <code>userid1022</code>	
 <code>userid1023</code>	
	 <code>admins</code> ["//user/wles/system/"]
	 <code>city</code> not set
	 <code>empno</code> 101000
	 <code>firstname</code> "Caroline"
	 <code>jobcode</code> "T163"
	 <code>lastname</code> "McKinney"
	 <code>numbers</code> not set
	 <code>promo_flag</code> "N"
	 <code>subjectgroup_name</code> ["Back-end Project Manager", "Engineering Director", "Test Engineer"]

Group

A group is represented by the  icon in the Administration Console and is a logical collection of users that share some common characteristics, such as department, job function, or job title. For example, a company may separate its sales staff into two groups, Sales Representatives and Sales Managers, because they want their sales personnel to have different levels of access to resources depending on their job functions.

A group can contain either users or other groups; users who are assigned to a group are called group members. Nested memberships of groups within a group form a hierarchy. Group membership can be assigned only from within the same directory. Groups have a static identity that an administrator assigns. All group names must be unique within a policy domain.


Managing groups is more efficient than managing large numbers of users individually. By using groups, you do not need to define policy for each and every user. Instead, each user in the group inherits the rules applied to the group; this rule also applies to nested groups. Granting a permission or role to a group is the same as giving that permission or role to each user who is a member of the group. For example, an administrator can specify roles for 50 users at one time by placing the users in a group, and then granting that group the role on a given resource.

Figure 4-8 shows how groups are represented in the Administration Console. In this illustration, the `junior_trader` group contains three group members, three user members, and one attribute (`my_favorite_color`).

Figure 4-8 Group Representation in the Administration Console




Identity Attribute

Identity attributes are represented by the  icon in the Administration Console (as shown in [Figure 4-8](#)). Each user and group can have different characteristics defined as identity attributes. The type of information or attributes collected—a method typically referred to as profiling—also varies and typically includes information such as name and address, phone, e-mail address, personal preferences, and so forth. Identity attributes can be extracted from the external data source.


An identity attribute is declared specifically to contain identity information. An attribute value can be used in rules to set limits for that user. Attributes provide a very powerful way to refer to users and groups indirectly in rules, which results in a more dynamic and versatile policy.

Role

A role is represented by the  icon in the Administration Console. A role is a dynamic alias used to associate users and groups to policy-based functional responsibilities. A role represents a collection of privileges on a resource. Roles are computed and granted to users or groups dynamically, based on conditions, such as user name, group membership, identity attributes, or dynamic data, such as the time of day. Roles membership can apply to only specific resources within a single application or can be applied globally across the enterprise, based on how you distribute the policy. A role can also be delegated from one user to another user.

Granting a role to a user or a group confers the defined access privileges to that user or group, as long as the user or group is a member of the role. Multiple users or groups can be granted a single security role.

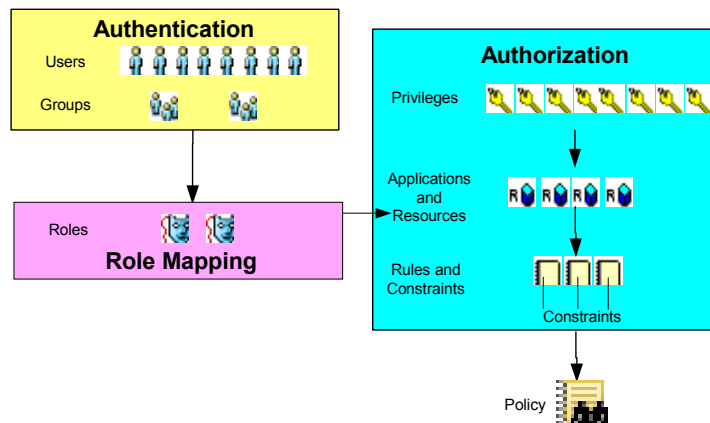
Role Policy

A role policy is represented by the  icon in the Administration Console and displays the rules that determine the membership for each role you have created, to which a user or group can be assigned. BEA recommends defining roles, and then assigning users and groups to those roles. A delegation rule assigns a role or privilege on a resource from one user or group (delegator) to another user or group (delegate). To delegate, the delegator must first have the role assigned to them. Role rules are used to grant users or groups membership into a given role. The membership can be limited based on a number of items including the resource hierarchy, identity, contextual, and resource attributes. Roles may also be delegated from one user to another using role delegation rules.


Policy

Authorization policy controls what actions a user can perform on a resource as defined by a set of rules. The authorization rules are typically written to grant specific privileges upon resources to users within a given role under a defined set of conditions. Figure 4-9 shows how the process of authorizing a user involves both computing the user role membership and then applying the authorization policy.

Figure 4-9 Role-based Authorization Policy




Policy Rule

A policy rule is represented by the blue checkmark  icon in the Administration Console. A policy rule defines the privileges, resources, subjects (users, groups, and roles), and conditions to use on a resource. BEA recommends that authorization rules be written to grant privileges to roles, rather than granting them directly to users or groups.


A policy rule may also be used to assign a privilege on a resource from one user (delegator) to another user or group (delegate); this is referred to as a delegate rule.

Policy Inquiry

A policy inquiry is represented by the  icon, in the Administration Console and allows you to analyze your authorization policy before distributing it. You can generate a query by providing a value for any policy element (privilege, user or group, role or resource). If you want more specific results, you may combine two or more policy elements.

A policy inquiry is a type of policy analysis that allows you to ask questions about how a policy responds to specific access requests. Policy inquiry performs a special type of rule search that takes into account role inheritance, resource hierarchies, and user to group mappings when determining the results. As with searches, there are two main classes of inquiries: grant and deny. Grant inquiries are more common. Grant inquiries ask what rules contain access rights that directly or indirectly match input values. Deny inquiries ask what rules deny access rights that directly or indirectly match input values.


Policy Verification

A policy verification is represented by the  icon in the Administration Console. You can cross check a policy using policy verification or you can search the list of policies to determine what privileges users and groups have over the selected resources. A policy verification allows you to find users whose rights were intended to be mutually exclusive.

At first, inquiries and verifications look very similar. However, an inquiry asks about specific users, actions, and applications. In verification, you are asking which users can perform mutually exclusive activities. With verifications you are looking for conflicting entitlements in individual user's security policies. With policy verifications, you are asking who is entitled to do one of two different actions. The results contain a list of those users who can do both, so you can verify whether your policy contains access rights for a single user that could pose a security risk.


Note: Policy inquiry and verification do not take role membership into account when analyzing the policy.

Declarations

A declaration is represented by the  icon. A declaration is a variable that represents either a predefined value (for example, days of the week) or a value that is dynamically defined at runtime (the date). To help you design efficient rules, various built-in declarations are provided for your use. There are four types of declarations:

- **enumerated type**—A type that consists of a predefined list of ordered values from which you create constants and attributes. The system comes with a number of predefined enumerated types and you can define your own. For example, you could define the enumerated type "color" with the values of "red", "green", or "blue".
- **constant**—A named, predefined, static value, or set of values that you can reference in a policy for a value that does not change at runtime.
- **attribute**—Represents characteristics that define dynamic values, users, groups, resources and configurations. An attribute has an associated type which may either be a built-in type (such as string, integer, date) or an enumerated type.
- **evaluation function**—A named function that you can use in a rule condition to perform more advanced operations. Each function may have a number of parameters and returns a Boolean result. There are a number of built-in evaluation functions and you can declare and use your own custom evaluation functions. Each custom evaluation function must be registered as a plugin with the authorization and role mapping engine (ARME) using it.

Deployment

Deployment is represented by the  icon in the Administration Console. Once you have designed and tested your security policy and configuration, you need to deploy so the policy and configuration take effect in your environment. When you distribute policy, you choose the distribution point for the policy before you actually distribute it. The distribution point identifies what portions of the policy updates are made active in your environment. After the distribution, you can view the results of the policy distribution. You also distribute security configuration data. After a configuration update, you must restart the effected modules to make use of the new configuration; this means you must restart the Security Service Module service.

The Deployment Status page allows you to gather information regarding policy and configuration deployments. The page shows if any components are out of sync with the central Administration Application.

What's Next?

Now that you understand the basic building blocks involved in security configuration and policy management, you can begin modeling your policy. The *Policy Managers Guide* provides some guidelines as to how to begin.

- [Chapter 1, “Modeling Policies,”](#) describes the process for creating a policy model, describes how to test the policy model, and then how to deploy.
- [Chapter 2, “Defining Rules to Enforce Policy,”](#) describes how to write policy rules and how to write role rules.
- [Chapter 3, “Create Policy Data Files,”](#) describes how to generate policy data outside of the Administration Application.
- [Chapter 4, “Importing Policy Data,”](#) details how to load policy data using the Policy Loader.
- [Chapter 5, “Exporting Policy Data,”](#) details how to export policy data from one an Administration Application for use with another instance of the Administration Application.

Using the Console

Overview

The Administration Console is a web browser-based, graphical user interface you use to manage application security and configure the BEA WebLogic Enterprise Security security service modules. Each module manages a separate policy domain that contains its own set of security providers. You deploy and manage your resources and policies as part of that policy domain.

These topics describe the types of console preferences you can set when working with the Administration Console. You can set the following preferences to control how the console displays information:

- Display or hide help text that appears on the Security Configuration tabs
- Define the page size

For information on how to use the console, see [Using the Administration Console](#).

Secure Sockets Layer (SSL) provides secure two-way connections by allowing the Administration Application to communicate with other components over a network connection to authenticate by encrypting the data exchanged. Authentication allows a server and optionally a client to verify the identity of the application on the other end of a network connection. Encryption makes data transmitted over the network intelligible only to the intended recipient. For detailed information on how to configure SSL, see [“Configuring Secure Sockets Layer for a Production Environment” on page 7-1](#).

Checking the Console Version Number

To check the version of the Administration Console:

1. Click the Administration Console icon. 

The Administration Console page displays three tabs labeled: Preferences, Failover and About.

2. Click the About tab.

The About tab displays the version number and release date for the BEA WebLogic Enterprise Security software you have installed, along with the BEA copyright statement.

Setting Console Preferences

The Preferences tab allows you to customize the way the Administration Console displays information. This is very useful if your policy database contains a large amount of data. By defining filter strings and restricting the page size, you can limit the information displayed on a page. Preferences are saved automatically in a cookie on your local machine so they are remembered between sessions. You can define the number of records displayed on a page for any of the policy elements, based on:

Filter String — the default pattern to search for when retrieving policy elements

Page Size — the number of items displayed per page of results

To set the Administration Console preferences:

1. Click Administration Console icon. 

The Console Preferences page displays three tabs labeled Preferences, Failover and About. The Preferences tab allows you to customize the way the Administration Console displays information.

2. To change the number of items displayed on a page:

- a. Using wildcard characters, define a filter for each item in the Filter String text box. The default wildcard (*) selects all available elements.

You can use special wildcard symbols to abbreviate names in searches, queries, and constraints. A wildcard behaves exactly like a file wildcard in DOS or Windows.

An asterisk (*) represents zero or more characters. For example, in a search, *Order would find the JobOrder but not JobOrders.

A question mark represents exactly one character. For example, a search for J?oe will find both JDoe and Jpoe but not JFloee.

Brackets [] mean match any character inside the brackets. For example, [Aac]* matches Apple, apple, and cat, but does not match Cat.

- b. To change the number of items displayed on a page, enter a different Page Size for each policy element.

The default number of items displayed on a page is set to 50, except for Policies (inquiries and verifications), and Role Policies which is set to 20. If there are more items on the page than can fit on the screen, a scroll bar appears on the right side of the page. You can slide the scroll bar to view the policy elements on the page. To page through the entire collection of results, click the page up and page down buttons.

3. Click **Apply**.

Note: If you want to restore the default settings for all policy elements, click Reset.

Starting the Administration Console

To ensure that your transactions are securely encrypted, the Administration Console uses Secure Socket Layers (SSL) to communicate with the administration server.

To start the Administration Console:

1. Do one of the following:
 - Click **Start**, and then select **Programs, BEA WebLogic Enterprise Security, Administration Console**.
 - Open a web browser, and enter the URL for your Administration Console:

`https://hostname:port`

or

`https://hostname:port/asi`

Where:

hostname — the Domain Name Server (DNS) name or IP address of the Administration Server.

port — the port number through which the administration server connects.

asi — the default virtual name for the Administration Console.

The Administration Console is configured to use demo certificates to connect to the Administration Server the first time. Secure Sockets Layer (SSL) provides secure two-way connections by allowing the Administration Application to communicate with other components over a network connection, encrypting the data exchanged. Authentication allows a server and optionally a client to verify the identity of the application on the other end of a network connection. Encryption makes data transmitted over the network intelligible only to the intended recipient. For detailed information on how to configure SSL, see [“Configuring Secure Sockets Layer for a Production Environment” on page 7-1](#).

When the login page appears, enter a Username and the Password that is granted to one of the security roles with a login privilege.

If this is the first time you are logging into the console, enter the default Username (system) and Password (weblogic). Default administrators are configured on install and can be used for the initial login. Once you have started the console, you should set up additional administrative users or configure an Authentication provider to authenticate console users to an external authentication source such as LDAP or Microsoft Windows NT, and update the administration policy accordingly. You should also change the Username and Password for the system user.

2. Click Sign In.

The Administration Console allows administrators to edit configurations or perform other operations based on security roles granted by the administration policy. If your security role does not permit editing of configuration data, for example, the data is displayed in the Administration Console but is not editable. If you try to perform an operation that is not permitted, the Administration Console displays an Access Denied error. For information on the default security roles, see [“Administration Policy.”](#)


3. When the Security Alerts appear, click Yes to accept the security certificate.

Logging out of the Administration Console

To log out of the Administration Console, do one of the following:

- On the title bar, click Logout. and then click OK.

-or-

- Click the Close button  in the upper-right hand corner of the browser.

Using the Administration Console

This section describes the various components of the Administration Console and how to use them.

Figure 5-1 Administration Console Layout

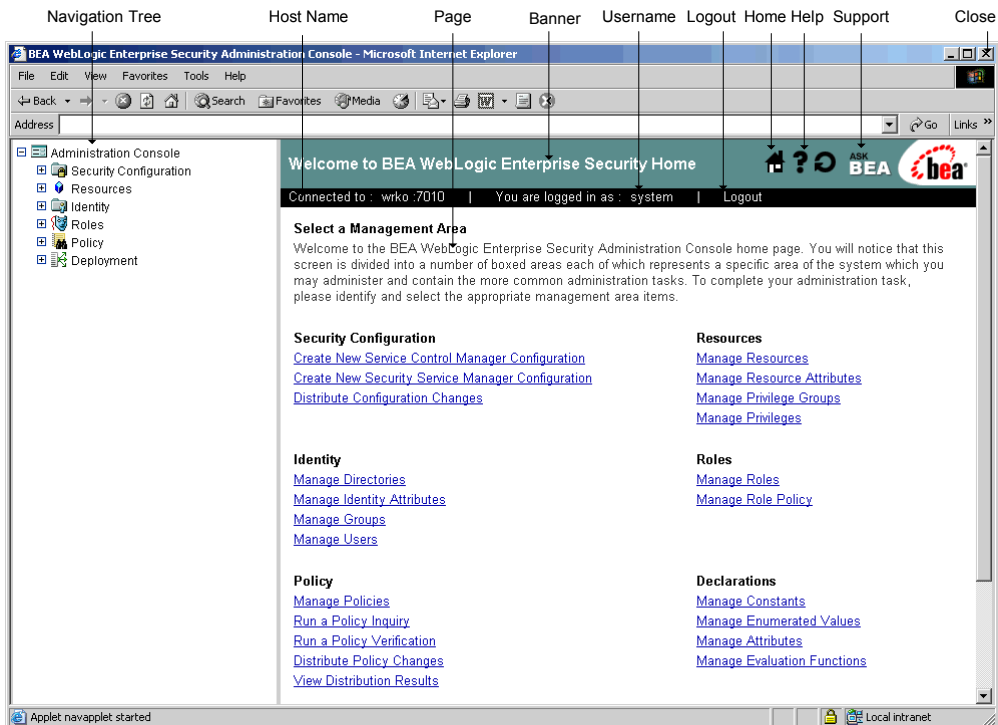


Table 5-1 Administration Console Components

Component	Description
-----------	-------------


Table 5-1 Administration Console Components (Continued)

Navigation Tree	The left panel of the Administration Console contains a navigation tree that you use to move from one console page to another. By selecting (left-clicking) a node (often referred to as a folder) in the tree, you can access the console pages related to that node, which then appear in the right panel of the console. If a node in the tree is preceded by a plus sign (+), you can click on the plus sign to expand the tree and access additional nodes.
Host Name:port/domain name	The name of the host or the IP address of the Administration Server, the corresponding port number, and the enterprise domain name.
Page	<p>The page being displayed. Some pages display one or more tabs where you configure your policy or security service module. After you select a node or folder from the navigation tree, a list of objects that you can configure appears as either a branch of the tree or as a series of tabs in the right panel of the console.</p> <p>When you open a Security Configuration folder, one or more Service Control Manager folders appear, displaying the names of each Security Service Module you have configured and bound. Opening a Security Service Module folder displays the security providers for that module: Adjudication, Auditing, Authentication, Authorization, Credential Mapping and Role Mapping.</p> <p>When you create a new provider, the provider tab displays a page containing one or more tabs (two of these tabs are called General and Details). These tabs contain the configuration attributes or controls for defining your security providers. You change configuration data by editing the attributes displayed in the right panel. After you make your changes, you must click the Apply button before the changes take effect. Some attributes are displayed in a light gray indicating that you cannot change the value. An attribute may or may not be editable depending on the value of another attribute.</p> <p>Opening other policy element folders, such as Roles, Policy or Identity, allows you to manage those policy elements and design your authorization policy.</p> <p>Most tabs and pages have buttons that define what actions you can perform (New, Add, Edit, Delete, Apply). See the online help for details on how to use any page, tab, or button.</p> <p>Each page saves your results if you navigate to another page, and then returns you to that page. However, any data that you entered on that page, like provider configuration information, is not retained unless you save it by clicking the Apply button.</p>
Username	The name you used to log in to the Administration Console.
Logout	Optional link to log out of the Administration Console.

Table 5-1 Administration Console Components (Continued)

Banner	<p>The banner includes the name of the product and the following icons:</p> <div data-bbox="467 374 537 444"></div> <p>Home Page</p> <p>The Administration Console home page is the first page displayed in the right panel when you start the console. You can always display this page by clicking the Home page icon in the banner area of the console. From this page, you can navigate quickly to the pages you use most frequently. You can also use the Navigation Tree to access these pages.</p> <div data-bbox="462 618 521 680"></div> <p>Help</p> <p>The Help button displays the Administration Console online help. See Getting Help for additional assistance on this topic.</p> <p>Note: Context sensitive help is not supported in the Administration Console.</p> <div data-bbox="462 840 521 900"></div> <p>Refresh</p> <p>The Refresh button refreshes the current page that you are working on, as well as all other data cached by the console. If you add a new policy element and it does not appear in the list immediately, you may need to click Refresh.</p> <div data-bbox="462 1017 577 1088"></div> <p>BEA E-support</p> <p>The ASK BEA button displays the BEA customer support web page. You can use this page to submit a question directly to our customer support personnel, obtain answers through our FAQ, access additional documentation for your product, or join one of our news groups.</p> <div data-bbox="462 1234 504 1272"></div> <p>Close</p> <p>The Close button provides a shortcut that lets you log you out of the Administration Console.</p>
--------	--

Getting Help

Documentation on how to use the Administration Console is included with the product. To get help, click the Help button  in the upper right-hand corner of the banner.

When you click the Help button, a new browser window opens containing help for the Administration Console. The text appearing in the right frame of the help window describes the functionality of the console page you have selected and may contain links to other related tasks.

Use the left frame of this window (the Navigation Tree) to navigate to other help topics using the Table of Contents, the alphabetical Index, or the Search function.

Use the <<<Back>> or <<Fwd>>> buttons to step through previously viewed pages.

For a list of general topics, click the Contents button and select a topic from the list displayed. The text of that topic appears in the right frame and a table of contents containing links to headings under that topic appears in the left frame.

Use the Print button to print the current topic.

Configuring the Administration Server for Failover

You can install two administration servers: a primary and a secondary. The secondary administration server is only used as a backup when the primary becomes unavailable. You must install the secondary server before you can configure it for backup purposes. For information on installing a secondary server, see the *BEA WebLogic Enterprise Security Administration Application Installation Guide*.

To configure the Administration Server for failover:

1. Click Administration Console icon. 

The Console Preferences page displays three tabs labeled: Preferences, Failover, and About.

2. Click the Failover tab.

This tab allows you to configure this Administration Server as either a primary or a secondary (backup) Administration Server. If this is a secondary server, you must specify all parameters so the primary server can be located and can periodically request a list of

trusted entities. This mechanism keeps the primary and secondary synchronized so that the secondary server can be designated as the primary Administration Server if necessary. If this is a primary server, you don't need to do anything except ensuring that the Primary option is checked.

3. Select Backup.
4. In the Primary URL text box, enter the URL for enrollment on the Secondary server. This URL is used to synchronize a trust relationship.
5. In the Username text box, enter the username to use when requesting synchronization of a trust relationship.
6. In the Enter Password and Confirm Password text boxes, enter the password to use when requesting synchronization of a trust relationship.
7. In the Synchronization interval text box, enter the interval between trust relationship synchronization attempts in the Synchronization Interval text box.

The value for this setting depends on how frequently Security Service Module or Service Control Manager instances are enrolled and unenrolled with the primary Administration Application.

8. Click Apply.

Additional BEA Documentation Available on the Internet

Additional documentation is also available on the BEA e-docs web site. Many help pages include links to related topics on the e-docs web site. These links are preceded with the label (e-docs). An Internet connection is necessary to view this documentation. BEA product documentation, along with other information about BEA software, is available from the BEA dev2dev web site:

<http://dev2dev.bea.com>.

Starting and Stopping Services

This chapter details how to start and stop processes and services on both the Administration Server and Security Service Modules.

- [“Starting and Stopping Administration Server Processes On Windows” on page 6-1](#)
- [“Starting and Stopping Administration Server Processes on Unix” on page 6-3](#)
- [“Starting and Stopping Security Service Module Processes” on page 6-5](#)

Starting and Stopping Administration Server Processes On Windows

Both the Service Control Manager and the Administration Server are registered with the Windows Service Manager. On install, they are not configured to start automatically. To have them start automatically, the system administrator must toggle the Manual start to Automatic in Windows. On Windows, to set services to start automatically, click Start>Settings>Control Panel>Administrative Tools>Services, and toggle the Startup Type from Manual to Automatic for each service that has a WLES Description prefix. Starting the Administration Server starts the following services, where *name* is the name of the machine on which you installed the product:

- WLES Authorization Role Mapping —ARME.admin.server.asi.*name*
- WLES Business Logic Manager —BLM.asi.*name*
- WLES Policy Distributor —PD.asi.*name*
- WLES Service Control Manager

- WLES WebLogic Server—WLS.asi.name

Table 6-1 lists the commands and Program menu options you use to manage Administration Server processes. To use commands to start and stop processes on Windows, open a command window, go to BEA_HOME\wles42-admin\bin (for Administration Server commands) or BEA_HOME\wles42-scm\bin (for Service Control Manager commands) and enter the command. Verify your services have started properly by checking the Services option in Windows.

Table 6-1 Windows Program Menu Options and Commands

Menu Option	Command	Description
Start Server	WLESadmin start	Starts the Administration Server and all other processes.
Start Server (Console)	WLESadmin console	<p>Starts the Administration Server and all other processes in separate console windows, except for the Service Control Manager. To stop a process in a console window, close the console window or press Ctrl+C. When the processes are started console mode you may see a message like:</p> <pre>08/25/04 18:21:11utc ERR [3040]iomanager.cpp(95): ***** Opening ERR Stream *****</pre> <p>This is not an error but is an Administration Server test that ensures that it can write to an error log; the error is standard output for the Administration Console. The message is normal and indicates that the various server components are running in console mode.</p> <p>Note: You must start the Service Control Manager process separately when using this command. This option is less secure as it is not started as either the asiadmin or scmuser.</p>
Stop Server	WLESadmin stop	Stops all Administration Server processes, with the exception of the Service Control Manager process, which you must shut down separately.
Refresh SCM	WLESscm refresh	Clears any cached configuration information and loads fresh Security Service Module configuration information from the Administration Server.

Table 6-1 Windows Program Menu Options and Commands (Continued)

Menu Option	Command	Description
Start SCM	WLEsscm start	Starts the Service Control Manager process. You only need to use this command if the SCM did not start along with other server processes.
Start SCM (console mode)	WLEsscm console	Starts the Service Control Manager process in a console window. To stop a process in a console window, close the console window or press Ctrl+C.
Stop SCM	WLEsscm stop	Stops the Service Control Manager process.

Starting and Stopping Administration Server Processes on Unix

Both the Service Control Manager and the Administration Server are registered with the Unix init subsystem. On install, they are not configured to start automatically. To have them start automatically, the system administrator link them into the correct init runlevel in Unix.

On Sun Solaris and Linux platforms, you must always start these servers as root. A common utility called `SUDO` (<http://www.courtesan.com/sudo/>) or something similar can be used to allow non-root users to start and stop these processes as root without having to give out the root password or violate the Application Security Infrastructure (ASI). [Table 6-2](#) lists the services for Sun Solaris and Linux platforms.

Note: To start and stop processes on Unix, go to `BEA_HOME/wles42-admin/bin` (for Administration Server commands or `BEA_HOME/wles42-scm/bin` (for Service Control Manager commands) and enter the shell script command as listed in [Table 6-2](#).

Table 6-2 Unix Commands

Command	Description
<code>WLESadmin.sh start</code>	Starts the Administration Server and all other processes.
<code>WLESadmin.sh console</code>	<p>Starts the Administration Server and all other processes in separate console windows, except for the Service Control Manager. To stop a process in a console window, close the console window or press Ctrl+C. When the processes are started console mode you may see a message like:</p> <pre>08/25/04 18:21:11utc ERR [3040] iomanager.cpp(95) : ***** Opening ERR Stream *****</pre> <p>This is not an error but is an Administration Server test that ensures that it can write to an error log; the error is standard output for the Administration Console. The message is normal and indicates that the various server components are running in console mode.</p> <p>Note: You must start the Service Control Manager process separately when using this command.</p>
<code>WLESadmin.sh stop</code>	Stops all Administration Server processes, with the exception of the Service Control Manager process, which you must shut down separately.
<code>WLESscm.sh refresh</code>	Clears any cached configuration information and loads fresh Security Service Module configuration information from the Administration Server.
<code>WLESscm.sh start</code>	Starts the Service Control Manager process. You only need to use this command if the SCM did not start along with other server processes.
<code>WLESscm.sh console</code>	Starts the Service Control Manager process in a console window. To stop a process in a console window, close the console window or press Ctrl+C.
<code>WLESscm.sh stop</code>	Stops the Service Control Manager process.

On Linux platforms, to allow the Administration Server start up after a reboot, you must properly set it to run on the proper init runlevel. This is not done by default. To properly register the Administration Server to start on boot, you must tell it to start on runlevel 3 and runlevel5, where runlevel 3 is the non-graphical runlevel and runlevel 5 is the graphical runlevel. A server comes

up and goes into runlevel 3 or 5 depending upon the configuration. To register the Administration Server on Linux, run the following command as root:

```
chkconfig --level 35 WLESadmin on
```

Note: The database configuration is available to these scripts on boot. The scripts use configurations located in the `/etc/profile` directory, where many database configurations are put. If you do not put the database configuration in `/etc/profile`, edit `bin/WLESadmin.sh` to set the appropriate environment variables and paths before rebooting.

To check the runlevel of the Administration Server, run:

```
chkconfig --list WLESadmin
```

Starting and Stopping Security Service Module Processes

After you install the Security Service Module, create the instance, and enroll it, you must start the necessary processes by running the appropriate batch or shell scripts. Before you start these processes, make sure that the Administration Server and all of its services are running.

For each machine, you must start the following processes:

- One Service Control Manager
- One Authorization and Role Mapping Engine (ARME) for each Security Service Module instance.

For instructions on how to start the required processes, see the following sections:

- [“Starting and Stopping Processes on Windows” on page 6-5](#)
- [“Starting and Stopping Processes on UNIX” on page 6-7](#)

Starting and Stopping Processes on Windows

There are two components that must be started for Security Service Module to function properly: the SCM process and the ARME process. You can use either the program menu options or the commands (listed in [Table 6-3](#)) to start these processes:

- WLES Service Control Manager
- WLES Authorization Role Mapping Engine—ARME.admin.server.asi.*name*

Note: To verify these services have started properly, check the Services option in Windows.

The Service Control Manager is registered with the Windows Services manager. On install, it is not configured to start automatically. To have it start automatically, the system administrator must toggle the Manual start to Automatic in Windows. The Authorization and Role Mapping Engine (ARME) is also not registered with the Windows Services manager. On Windows, to set the Service Control Manager to start automatically, click Start>Settings>Control Panel>Administrative Tools>Services, and toggle the Startup Type from Manual to Automatic for the WLES Service Control Manager.

Note: If you have installed the Security Service Module on the same machine as the Administration Server, you do not need to start the SCM; it is already started when you boot the Administration Server. If you installed the Security Service Module on a different machine, you must start the SCM and then the ARME process.

Table 6-3 lists the commands and program menu options you use to start and stop processes for a Service Control Manager and Security Service Module ARME instance. To use commands to manage processes on Windows, open a command window, go to `BEA_HOME\wles42-scm\bin` (for Service Control Manager commands) or to use ARME commands, open a command window, go to `BEA_HOME\wles42-ssm\<ssmtype>-ssm\instance\instancename\bin` (where `<ssmtype>` is either `java`, `iis`, `wls`, or `web service`), and enter the command as shown in Table 6-3. Verify your services have started properly by checking the Services option in Windows.

Note: You must start the Service Control Manager before starting the ARME process.

Table 6-3 Windows Program Menu Options and Commands

Menu Option	Command	Description
Refresh SCM	<code>WLEScm refresh</code>	Clears any cached configuration information and loads fresh Security Service Module configuration information from the Administration Server.
Start SCM	<code>WLEScm start</code>	Starts the Service Control Manager process.
Start SCM (console mode)	<code>WLEScm console</code>	Starts the Service Control Manager process in a console window. To stop a process in a console window, close the console window or press Ctrl+C.
Stop SCM	<code>WLEScm stop</code>	Stops the Service Control Manager process.
Refresh ARME	<code>WLESarme refresh</code>	Updates the ARME to include the most recent policy data from the Application Server.

Table 6-3 Windows Program Menu Options and Commands (Continued)

Menu Option	Command	Description
Start ARME	WLESarme start	Starts the ARME process for the instance, as a Windows Service.
Start ARME (console mode)	WLESarme console	Starts the ARME process for the instance in a console window. To stop a process in a console window, close the console window or press <code>Ctrl+C</code> .
Stop ARME	WLESarme stop	Stops the ARME process for the instance.

Starting and Stopping Processes on UNIX

To start and stop processes on Unix, go to `/opt/beahome/wles42-ssm/<ssmtype>-ssm/bin` (where `<ssmtype>` is either `apache`, `java`, `wls`, or `webService`), and enter the commands listed in [Table 6-4](#).

Note: You must start the Service Control Manager before starting the ARME process.

Note: For an additional Linux platform start-up option, see [“Start-Up Option on Linux Platforms” on page 6-8](#).

Table 6-4 Unix Commands

Command	Description
WLESscm.sh refresh	Clears any cached configuration information and loads fresh Security Service Module configuration information from the Administration Server.
WLESscm.sh start	Starts the Service Control Manager process.
WLESscm.sh console	Starts the Service Control Manager process in a console window. To stop a process in a console window, close the console window or press <code>Ctrl+C</code> .
WLESscm.sh stop	Stops the Service Control Manager process.
WLESarme.sh refresh	Updates the ARME to include the most recent policy data from the Application Server.

Table 6-4 Unix Commands

Command	Description
<code>WLESarme.sh start</code>	Starts the ARME process for the instance, as a Windows Service.
<code>WLESarme.sh console</code>	Starts the ARME process for the instance in a console window. To stop a process in a console window, close the console window or press <code>Ctrl+C</code> .
<code>WLESarme.sh stop</code>	Stops the ARME process for the instance.

Start-Up Option on Linux Platforms

On Linux platforms, to allow the Service Control Manager to start up after a reboot, you must set it to the proper init runlevel. This is not done by default. To properly register the Service Control Manager to start on boot, set it to start on runlevel 3 and runlevel 5, where runlevel 3 is the non-graphical runlevel and runlevel 5 is the graphical runlevel. A server comes up and goes into the appropriate runlevel depending upon the configuration. To register the Service Control Manager on Linux, run the following command as root:

```
chkconfig --level 35 WLESscm on
```

To check the runlevel of the Service Control Manager, run:

```
chkconfig --list WLESscm
```

Configuring Secure Sockets Layer for a Production Environment

BEA WebLogic Enterprise Security uses an implementation of the Transport Layer Security (TLS) 1.0 specification. The server (WebLogic Server 8.1) hosting the WebLogic Enterprise Security Administration Application supports TLS on a dedicated listen port which defaults to 7010. To establish a secure connection, a Web browser connects to the Administration Server by supplying the listen port and the secure address (HTTPS) in the connection URL, for example, `https://myserver:7010`. The Administration Server returns a certificate to identify itself to the client.

When you install the BEA WebLogic Administration Application, demo certificates are provided for working in a development environment. However, it is very important that these certificates *not* be used in a production environment.

The following sections describe how to configure Secure Sockets Layer (SSL) for a production environment:

- [“Some SSL Basics” on page 7-1](#)
- [“Configuring SSL” on page 7-3](#)
- [“SSL Certificate Validation” on page 7-12](#)
- [“Specifying the Version of the SSL Protocol” on page 7-16](#)

Some SSL Basics

The following topics provide some basic information about SSL and the WebLogic Enterprise Security Administration Application.

- [“Private Keys, Digital Certificates, and Trusted Certificate Authorities” on page 7-2](#)
- [“One-Way SSL Versus Two-Way SSL” on page 7-3](#)
- [“How WebLogic Enterprise Security Locates Trust” on page 7-3](#)

Private Keys, Digital Certificates, and Trusted Certificate Authorities

Private keys, digital certificates, and trusted certificate authorities establish and verify server identity. SSL uses public key encryption technology for authentication. With public key encryption, a public key and a *private key* are generated for a server. The keys are related so that data encrypted with the public key can only be decrypted using the corresponding private key and vice versa. The private key is carefully protected so that only the owner can decrypt messages that were encrypted using the public key.

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

The data embedded in a digital certificate is verified by a certificate authority and is digitally signed with the digital certificate of the certificate authority. Well-known certificate authorities include Verisign and Entrust. The trusted certificate authority (CA) certificate establishes trust for a certificate.

Web browsers, servers, and other SSL-enabled applications generally accept as genuine any digital certificate that is signed by a trusted certificate authority and is otherwise valid. For example, a digital certificate can be invalid because it has expired, or the digital certificate of the certificate authority used to sign it expired, or because the host name in the digital certificate of the server does not match the URL specified by the client.

One-Way SSL Versus Two-Way SSL

You can configure SSL to use either one-way or two-way authentication:

- With one-way SSL, to establish an SSL connection, the server is required to present a certificate to the client but the client is not required to present a certificate to the server. To successfully negotiate an SSL connection, the client must authenticate the server but the server accepts any client into the connection. One-way SSL is common on the Internet where customers want to create secure connections before sharing personal data. Often, clients use SSL to log on so that the server can authenticate them. The Administration Application is configured to use one-way SSL by default.
- With two-way SSL, to establish the SSL connection, the server is required to present a certificate to the client and the client is required to present a certificate to the server. WebLogic Enterprise Security can be configured to require clients to submit valid and trusted certificates before completing the SSL connection.

How WebLogic Enterprise Security Locates Trust

WebLogic Enterprise Security uses the following algorithm when it loads its trusted CA certificates:

1. If the keystore is specified by the `-Dweblogic.security.SSL.trustedCAkeystore` command-line argument, load the trusted CA certificates from that keystore.
2. Else, if the keystore is specified in the configuration file (`config.xml`), load trusted CA certificates from the specified keystore. If the server is configured with DemoTrust, trusted CA certificates will be loaded from the `WL_HOME\server\lib\DemoTrust.jks` and the JDK `cacerts` keystores, where `WL_HOME` is `BEA_HOME\weblogic81` by default.
3. Else load trusted CA certificates from `WL_HOME\server\lib\cacerts` keystore.

Configuring SSL

To configure SSL in the WebLogic Enterprise Security Administration Server for a production environment, perform the following tasks:

1. Obtain an identity (private key and digital certificates) and certificate from a trusted certificate authority, which signed the certificate. Use a reputable vendor such as Entrust or Verisign as the certificate authority. For instructions on how to perform this task, see [“Obtaining Private Keys, Digital Certificates, and Trusted Certificate Authorities” on page 7-4](#) for details on performing this task.

2. Store private keys, digital certificates, and trusted CA certificates in keystores. For instruction on how to perform this task, see [“Creating a Keystore and Loading Private Keys and Trusted Certificate Authorities” on page 7-4](#).
3. Configure the WebLogic Enterprise Security hosting the Administration Application to use these new identity and trust keystores. For instructions for performing these tasks, see the following topics:
 - [“Configuring Keystores” on page 7-8](#)
 - [“Configuring One-Way SSL” on page 7-10](#)
 - [“Configuring Two-Way SSL” on page 7-11](#).

Obtaining Private Keys, Digital Certificates, and Trusted Certificate Authorities

To use SSL, the server needs a private key, a digital certificate containing the matching public key, and the certificate for the certificate authority that signs the certificate. In one-way SSL, the client needs to trust the certificate authority; the server does not.

WebLogic Enterprise Security loads private keys, digital certificates, and trusted CA certificates from keystores. You can use the Sun Microsystems `keytool` utility to create the keystores.

This utility allows you to generate a private key, a self-signed digital certificate, and a Certificate Signing Request (CSR). Submit the CSR to a certificate authority to obtain a digital certificate. Use `keytool` to update the self-signed digital certificate with a new digital certificate signed by the certificate authority. For more information about the Sun Microsystems `keytool` utility, see [keytool—Key and Certificate Management Tool](#).

Note: WebLogic Enterprise Security does not support the use of the Digital Signature Algorithm (DSA). When using the `keytool` utility, the default key pair generation algorithm is DSA. Specify another key pair generation and signature algorithm when using WebLogic Enterprise Security.

Creating a Keystore and Loading Private Keys and Trusted Certificate Authorities

After you obtain private keys, digital certificates, and trusted CA certificates, you need to store them so that the Administration Server can use them to find and verify identity. Private keys, their associated digital certificates, and trusted CA certificates are stored in keystores. A keystore is a

mechanism designed to create and manage private key/digital certificate pairs and trusted CA certificates.

BEA recommends creating one keystore for identity (private key/digital certificate pairs) and another keystore for trust relationships (trusted CA certificates). However, one keystore can be used for both identity and trust.

By default, WebLogic Enterprise Security looks for an Identity keystore named `DemoIdentity.jks` in the `WL_HOME\server\lib` directory and a Trust keystore named `DemoTrust.jks` in the `WL_HOME\server\lib` directory (where `WL_HOME` is `BEA_HOME\weblogic81` by default) and `cacerts` in the `JAVA_HOME\jre\lib\security` directory (where `JAVA_HOME` is `BEA_HOME\jdk142_04` or `BEA_HOME\jdk142_05` by default).

All private key entries in a keystore are accessed through unique aliases. You specify the alias when loading the private key into the keystore. Aliases are case-insensitive; the aliases `Hugo` and `hugo` would refer to the same keystore entry. Aliases for private keys are specified in the Private Key Alias attribute when configuring SSL.

All certificate authorities in a keystore identified as trusted by WebLogic Enterprise Security are trusted. Although WebLogic Enterprise Security does not use the alias to access trusted CA certificates, the keystore does require an alias when loading a trusted CA certificate into the keystore.

You can use the following mechanisms to create a keystore and load private keys and trusted CA certificates into the keystore:

- Sun Microsystems `keytool` utility—You can use the `keytool` utility to generate a private key/digital certificate pair and then import the signed private key into the keystore. For more information, see [“Common Keytool Commands” on page 7-6](#). While you can use the `keytool` utility to generate new private keys and digital certificates and add them to a keystore, the utility does not allow you to take an existing private key from a file and import it into the keystore. Instead, use the WebLogic `ImportPrivateKey` utility.

Note: The `keytool` utility does allow you to import trusted CA certificates in a file into a keystore.

- `ImportPrivateKey` utility—You can use the `ImportPrivateKey` utility to load a private key into a private keystore file. For instructions on using the `ImportPrivateKey` utility, see [“Using the ImportPrivateKey Utility” on page 7-7](#).
- Custom utilities—This release of WebLogic Enterprise Security can use keystores created with custom tools or utilities. For information on how to use these utilities, see the user documentation provided with the utility.

Common Keytool Commands

[Table 7-1](#) describes the `keytool` commands for creating and using JKS keystores with WebLogic Enterprise Security.

Note: The `keytool` utility is a product of Sun Microsystems. Therefore, BEA does not provide complete documentation on the utility. For more information, see [keytool-Key and Certificate Management Tool](#), which is available from the Sun Microsystems web site.

Table 7-1 Commonly Used keytool Commands

Command	Description
<code>keytool -genkey -keystore keystorename -storepass keystorepassword</code>	Generates a new private key entry and self-signed digital certificate in a keystore. If the keystore does not exist, it is created.
<code>keytool -import -alias aliasforprivatekey -file privatekeyfilename.pem -keypass privatekeypassword -keystore keystorename -storepass keystorepassword</code>	Updates the self-signed digital certificate with one signed by a trusted CA.
<code>keytool -import -alias aliasfortrustedca -trustcacerts -file trustedcafilename.pem -keystore keystorename -storepass keystorepassword</code>	Loads a trusted CA certificate into a keystore. If the keystore does not exist, it is created.
<code>-certreq -alias alias -sigalg sigalg -file certreq_file -keypass privatekeypassword -storetype keystoretype -keystore keystorename -storepass keystorepassword</code>	Generates a CSR, using the PKCS#10 format. Sent the CSR to be sent to a trusted CA. The trusted CA authenticates the certificate requestor and returns a digital certificate to replace the existing self-signed digital certificate in the keystore.
<code>keytool -list -keystore keystorename</code>	Displays what is in the keystore.

Table 7-1 Commonly Used keytool Commands (Continued)

Command	Description
<code>keytool -delete -keystore keystorename -storepass keystorepassword -alias privatekeyalias</code>	Delete a private key/digital certificate pair for the specified alias from the keystore.
<code>keytool -help</code>	Provides online help for keytool.

Using the ImportPrivateKey Utility

You can use the `ImportPrivateKey` utility to load a private key into a private keystore file.

The syntax for using this utility is:

```
java -classpath %WL_HOME%\server\lib\weblogic.jar
utils.ImportPrivateKey keystorepass alias keypass certfile
keyfile
```

[Table 7-2](#) describes the `ImportPrivateKey` utility arguments.

Table 7-2 ImportPrivateKey Utility Arguments

Argument	Definition
<code>keystore</code>	Defines the name of the keystore file. A new keystore is created if one does not exist.
<code>keystorepass</code>	Defines the password to open the keystore file.
<code>alias</code>	Defines the name that is used to look up certificates and keys in the keystore.
<code>keypass</code>	Defines the password used to unlock the private key file and to protect the private key in the keystore.
<code>certfile</code>	The name of the certificate associated with the private key.
<code>keyfile</code>	The name of the file holding the protected private key.

Configuring Keystores

By default, WebLogic Enterprise Security is configured with two keystores:

- `DemoIdentity.jks`—Contains a demonstration private key for the Administration Server. This keystore contains the identity for WebLogic Enterprise Security.
- `DemoTrust.jks`—Contains the trusted certificate authorities from the `WL_HOME\server\lib\DemoTrust.jks` and the JDK `cacerts` keystores. This keystore establishes trust for WebLogic Enterprise Security Administration Server.

These keystores are located in the `WL_HOME\server\lib` directory, where `WL_HOME` is `BEA_HOME\weblogic81` by default.

For testing and development purposes, the keystore configuration is complete. However, the demonstration keystores should not be used in a production environment. All the digital certificates and trusted CA certificates in the keystores are signed by an Administration Server demonstration certificate authority. Therefore, all Administration Server installations trust each other. This configuration leaves your SSL connections vulnerable to a number of security attacks.

Use the steps in this section to configure Identity and Trust keystores for production use.

Before you perform the steps in this section, you need to perform the following tasks:

1. Obtain private keys and digital certificates from a reputable certificate authority such as Verisign, Inc. or Entrust. For more information, see [“Obtaining Private Keys, Digital Certificates, and Trusted Certificate Authorities”](#) on page 7-4.
2. Create Identity and Trust keystores and load the private keys and trusted CAs into the Identity and Trust keystores.. For more information, see [“Creating a Keystore and Loading Private Keys and Trusted Certificate Authorities”](#) on page 7-4.

To configure Identity and Trust keystores for WebLogic Enterprise Security:

1. To start the WebLogic Server Administration Console for WebLogic Enterprise Security, open an Internet Explorer browser and enter the following URL:

`https://hostname:port/console`

Where:

hostname is the Domain Name Server (DNS) name or IP address of the WebLogic Enterprise Administration Server.

port is the port number through which the WebLogic Server Administration Server connects. The default port is 7010.

WebLogic Enterprise Security is configured with a default identity keystore (DemoIdentity.jks) and a default trust keystore (DemoTrust.jks). In addition, WebLogic

2. Open the **Servers** folder.
3. Select the name of the server for which you want to configure keystores (for example, `asiAdminServer`).
4. Select the **Configuration-->Keystores and SSL** tab.
The information about the demonstration keystores is displayed in the Keystore Configuration.
5. Click the Change... link in the Keystore Configuration to configure new keystores.
6. Choose the type of keystore configuration being used. The following options are available:
 - Demo Identity and Demo Trust—The demonstration Identity and Trust keystores located in the `WL_HOME\server\lib` directory and configured by default and the `cacerts` file in the `JAVA_HOME\jre\lib\security` directory.
 - Custom Identity and Java Standard Trust—An Identity keystore you create and the trusted CAs defined in the `cacerts` file in the `JAVA_HOME\jre\lib\security` directory.
 - Custom Identity and Custom Trust—Identity and Trust keystores you create.
 - Custom Identity and Command-Line Trust—An Identity keystore you create and command-line arguments that specify the location of the Trust keystore.
7. Click **Continue**.
8. Define attributes for the Identity keystore.
 - Custom Identity Keystore File Name—The fully qualified path to the Identity keystore.
 - Custom Identity Keystore Type—The type of the keystore. Generally, this attribute is `jks`. If this attribute is not specified, the default keystore type defined in the security policy file for the JDK is used.

9. Define attributes for the Trust keystore.

If you choose Java Standard Trust, specify the password defined when creating the keystore. Confirm the password.

If you choose **Custom Trust**, define the following attributes:

- Custom Trust Keystore File Name—The fully qualified path to the trust keystore.
- Custom Trust Keystore Type—The type of the keystore. Generally, this attribute is `jks`. If this attribute is not specified, the default keystore type defined in the security policy file for the JDK is used.
- Custom Trust Keystore PassPhrase —This attribute is not necessary as WebLogic Server only reads from the keystore.

10. Click **Continue**.

11. Click **Finish**.

12. To have the changes take effect, stop and restart the WebLogic Enterprise Administration Server.

Configuring One-Way SSL

Use the steps in this section to configure SSL for a production environment.

To configure SSL:

1. To start the WebLogic Server Administration Console for WebLogic Enterprise Security, open an Internet Explorer browser and enter the following URL:

`https://hostname:port/console`

Where:

hostname is the Domain Name Server (DNS) name or IP address of the WebLogic Enterprise Administration Server.

port is the port number through which the WebLogic Server Administration Server connects. The default port is 7010.

WebLogic Enterprise Security is configured with a default identity keystore (DemoIdentity.jks) and a default trust keystore (DemoTrust.jks). In addition, WebLogic

2. In the Open the **Servers** folder.

3. Select the name of the server for which you want to configure keystores (for example, `asiAdminServer`).
4. Select the **Configuration-->Keystores and SSL** tab.
Information about the demonstration Identity and Trust keystores is displayed in the Keystore Configuration.
5. Configure new Identity and Trust keystores for WebLogic Enterprise Security. For more information, see [“Configuring Keystores” on page 7-8](#).
6. Click the **Change...** link in the SSL Configuration to configure attributes for SSL.
The Configure SSL page appears.
7. Use the Key Stores option to specify how the identity and trust for WebLogic Enterprise Security is stored.
8. Click **Continue**.
9. Specify the alias used to load the private key into the keystore in the Private Key Alias and the password used to retrieve the private key from the keystore in the Passphrase attribute. You may have specified this information when creating the Identity keystore; however, for the purpose of SSL configuration specify the information again. Skip to step 10.
Note: You do not have to specify this information for the Trust keystore because trusted CA certificates are not individually identified to WebLogic Enterprise Security Administration Server with aliases. All trusted CA certificates in a keystore identified as trusted by WebLogic Enterprise Security are trusted. Therefore, WebLogic Enterprise Security does not require an alias when retrieving a trusted CA certificate from the keystore.
10. Click **Continue**.
11. Click **Finish**.
12. To have the changes take effect, stop and restart the WebLogic Enterprise Administration Server.

Configuring Two-Way SSL

By default, WebLogic Enterprise Security Administration Server is configured to use one-way SSL (the server passes its identity to the client). For a more secure SSL connection, use two-way SSL. In a two-way SSL connection, the client verifies the identity and trust of the server and then

passes its identity to the server. The server then validates the identity and trust of the client before completing the SSL connection. The server determines whether or not two-way SSL is used.

Before configuring two-way SSL, ensure the Trust keystore for the server includes the certificate for the trusted certificate authority to be used to sign the certificates for the clients.

To enable two-way SSL:

1. Configure one-way SSL as described in [“Configuring One-Way SSL” on page 7-10](#).
2. Open the **Servers** folder.
3. Select the name of the server for which you want to configure two-way SSL (for example, asiAdminServer).
4. Select the **Configuration-->Keystores and SSL** tab.
5. Click the Show link under Advanced Options.
6. Go to the Server attributes section of the window.
7. Set the Two Way Client Cert Behavior attribute. The following options are available:
 - Client Certs Not Requested—The default (meaning one-way SSL).
 - Client Certs Requested But Not Enforced—Requires a client to present a certificate. If a certificate is not presented, the SSL connection continues.
 - Client Certs Requested And Enforced—Requires a client to present a certificate. If a certificate is not presented or if the certificate is not trusted, the SSL connection is terminated.
8. Click Apply.
9. Log out and restart the Administration Console.

SSL Certificate Validation

WebLogic Enterprise Security must verify all SSL certificates chains. Each CA certificate in a certificate chain must have a Basic Constraint extension defined, that designates it as a Certificate Authority. Any certificate authorities not meeting this criteria are rejected. This section describes the command-line argument that controls the level of certificate validation.

If the Administration Server is booted with certificate chains that do not pass the certificate validation, an information message is logged, noting that clients could reject it.

To assist you in managing SSL certificate validation, WebLogic Enterprise Security provides mechanisms for setting the level of the SSL certificate validation and for checking whether or not existing certificate chains will be rejected. For more information, see the following topics:

- [“Setting the Level of Certificate Validation” on page 7-13](#)
- [“Checking Certificate Chains” on page 7-15](#)
- [“Troubleshooting Problems with Certificates” on page 7-16](#)

Setting the Level of Certificate Validation

By default, WebLogic Enterprise Security rejects any certificates in a certificate chain that do not have the Basic Constraint extension defined as CA. However, you may be using certificates that do not meet this requirement or you may want to increase the level of security to conform to the IETF RFC 2459 standard. Use the following command-line argument to control the level of certificate validation performed:

```
-Dweblogic.security.SSL.enforceConstraints=option
```

[Table 7-3](#) describes the options for the command-line argument.

Table 7-3 Options for `-Dweblogic.security.SSL.enforceConstraints`

Option	Description
<code>strong</code> or <code>true</code>	<p>Use this option to check that the Basic Constraints extension on the CA certificate is defined as CA. This option does not enforce the IETF RFC 2459 standard.</p> <p>For example:</p> <pre>-Dweblogic.security.SSL.enforceConstraints=strong</pre> <p>or</p> <pre>-Dweblogic.security.SSL.enforceConstraints=true</pre> <p>By default, WebLogic Enterprise Security performs this level of certificate validation.</p>
<code>strict</code>	<p>Use this option to check the Basic Constraints extension on the CA certificate is defined as CA and set to critical. This option enforces the IETF RFC 2459 standard.</p> <p>For example:</p> <pre>-Dweblogic.security.SSL.enforceConstraints=strict</pre> <p>This option is not the default because a number of commercially available CA certificates do not conform to the IETF RFC 2459 standard.</p>
<code>off</code>	<p>Use this option to disable certificate validation. Use this option carefully. For example, if you purchased CA certificates from a reputable commercial certificate authority and the certificates do not pass the new validation, use this option. However, CA certificates from most commercial certificate authorities should work with the default <code>strong</code> option.</p> <p>For example:</p> <pre>-Dweblogic.security.SSL.enforceConstraints=off</pre> <p>BEA does not recommend use of this option in production environment. Instead, purchase new CA certificates that comply with the IETF RFC 2459 standard.</p>

Checking Certificate Chains

WebLogic Enterprise Security provides a `ValidateCertChain` command-line utility to check whether or not an existing certificate chain will be rejected. This utility uses certificate chains from PEM files, PKCS-12 files, PKCS-12 keystores, and JKS keystores. A complete certificate chain must be used with the utility. The following is the syntax for the `ValidateCertChain` command-line utility:

```
java utils.ValidateCertChain -file pemcertificatefilename
java utils.ValidateCertChain -pem pemcertificatefilename
java utils.ValidateCertChain -pkcs12store pkcs12storefilename
java utils.ValidateCertChain -pkcs12file pkcs12filename password
java utils.ValidateCertChain -jks alias storefilename [storePass]
```

Example of valid certificate chain:

```
java utils.ValidateCertChain -pem zippychain.pem
```

```
Cert [0]: CN=zippy,OU=FOR TESTING
ONLY,O=MyOrganization,L=MyTown,ST=MyState,C=US
```

```
Cert [1]: CN=CertGenCAB,OU=FOR TESTING
ONLY,O=MyOrganization,L=MyTown,ST=MyState,C=US
```

```
Certificate chain appears valid
```

Example of invalid certificate chain:

```
java utils.ValidateCertChain -jks mykey mykeystore
```

```
Cert [0]: CN=corba1,OU=FOR TESTING ONLY,
O=MyOrganization,L=MyTown,ST=MyState,C=US
```

```
CA cert not marked with critical BasicConstraint indicating it is a CA
```

```
Cert [1]: CN=CACERT,OU=FOR TESTING ONLY,
O=MyOrganization,L=MyTown,ST=MyState,C=US
```

```
Certificate chain is invalid
```

Troubleshooting Problems with Certificates

If SSL communications were working properly and start failing unexpectedly, the problem is mostly likely because the certificate chain used by the Administration Server is failing the validation.

Determine where the certificate chain is being rejected and decide whether to update the certificate chain with one that will be accepted, or change the setting of the `-Dweblogic.security.SSL.enforceConstraints` command-line argument.

To troubleshoot problems with certificates, use one of the following methods:

- If you know where the certificate chains for the processes using SSL communication are located, use the `ValidateCertChain` command-line utility (see [“Checking Certificate Chains” on page 7-15](#)) to check whether the certificate chains will be accepted.
- Turn on SSL debug tracing on the processes using SSL communication. The syntax for SSL debug tracing is:

```
-Dssl.debug=true -Dweblogic.StdoutDebugEnabled=true
```

The following message indicates the SSL failure is due to problems in the certificate chain:

```
<CA certificate rejected. The basic constraints for a CA certificate  
were not marked for being a CA, or were not marked as critical>
```

When using one-way SSL, look for this error in the client log. When using two-way SSL, look for this error in the client and server logs.

Specifying the Version of the SSL Protocol

WebLogic Enterprise Security supports both the SSL V3.0 and TLS V1.0 protocols. By default, the WebLogic Enterprise Security uses the SSL V3.0 protocol. While in most cases the SSL V3.0 protocol is acceptable, there are circumstances (such as compatibility, SSL performance, and environments with maximum security requirements), where the TLS V1.0 protocol is desired. The `weblogic.security.SSL.protocolVersion` command-line argument lets you specify what protocol to use for SSL connections.

When the Administration Server acts as a client, the SSL handshake starts with an SSL V2.0 hello from the Administration Server. The peer must respond with an SSL V3.0 or TLS V1.0 message or the SSL connection is dropped. This is the default behavior.

Note: The SSL V3.0 and TLS V1.0 protocols can not be interchanged. Only use the TLS V1.0 protocol if you are certain all desired SSL connections can be made using that protocol.

To specify which type of encrypted connection the Administration Server supports, use the following command-line arguments:

- `-Dweblogic.security.SSL.protocolVersion=SSL3`—Only SSL V3.0 messages are sent and accepted.
- `-Dweblogic.security.SSL.protocolVersion=TLS1`—Only TLS V1.0 messages are sent and accepted.
- `-Dweblogic.security.SSL.protocolVersion=ALL`—This is the default behavior.

Configuring Secure Sockets Layer for a Production Environment

Enabling Single Sign On

Configuring Single Sign On with Microsoft Clients

Using a Single Pass Negotiate Identity Asserter shipped with WebLogic Enterprise Security, Version 4.2 SP01, you can achieve cross-platform authentication, single sign on (SSO), integration with Microsoft Internet Explorer browser clients and Microsoft .NET web services.

Cross-platform authentication is achieved by emulating the negotiate behavior of native Windows-to-Windows authentication services. The servlet container in this release was modified to handle the necessary header manipulation required by the Windows negotiate protocol, also known as Simple and Protected Negotiate (SPNEGO).

The negotiate identity asserter is an implementation of the Security Service Provider Interface (SSPI) as defined by the WebLogic Security Framework and provides the necessary logic to authenticate a client based on the client's SPNEGO token.

Requirements

The environment for cross-platform authentication requires the following components.

- Domain controller back-end system
 - Windows 2000 or greater Active Directory
 - Service accounts for mapping Kerberos services
 - Service Principal Names (SPNs) properly configured
 - Keytab files created and inserted, based on platform environment of your server system

- WebLogic application server system
 - WebLogic Enterprise Security, Version 4.2, SP01 installed
 - WebLogic 8.1 Security Service Module installed and an instance created
 - WebLogic application server configured to use Active Directory realm
 - Keytab import and configuration
 - MIT Kerberos V5 Generic Security Service API (GSS-API)
- Client systems
 - .NET Framework 1.1
 - Windows 2000 Professional SP2 or greater
 - Internet Explorer 5.01, Internet Explorer v5.5 SP2, Internet Explorer 6.0 SP1
 - Proper configuration of the Internet Explorer browser
 - Proper configuration of web services clients

Note: Client users must be logged on to a Windows 2000 domain or realm, having acquired Kerberos credentials from the Active Directory domain. Local logons do not work.

The following sections describe how to configure the SPNEGO provider and how to set up the necessary components.

- [“Enabling a Web Service or Web Application” on page 8-2](#)
- [“Configure the Client .NET Web Service” on page 8-6](#)
- [“Configure the Internet Explorer Client Browser” on page 8-7](#)

Enabling a Web Service or Web Application

To enable a particular web service, web application, or other protected resource for single sign on, you must use the Single Pass Negotiate Identity Asserter provider in conjunction with client certification set as the login configuration in your standard J2EE `web.xml` descriptor file.

Configuring the SPNEGO Provider

Configure the Single Pass Negotiate Identity Assertion provider using the Administration Console. To configure the provider, create an instance of the SPNEGO provider for the WebLogic Server 8.1 Security Service Module. For information on how to configure the provider, see [“Configuring a Single Pass Negotiate Identity Asserter” on page 9-30](#).

Editing the Descriptor File

The following example shows a sample `web.xml` file for a protected WebLogic Web Service resource (Conversation) with the login configuration set to `CLIENT-CERT`.

```
<!DOCTYPE web-app PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application
2.3//EN" "http://java.sun.com/dtd/web-app_2_3.dtd">

<web-app>

  <security-constraint>

    <display-name>Security Constraint on Conversation</display-name>

    <web-resource-collection>

      <web-resource-name>Conversation web
service</web-resource-name>

      <description>Only those granted the ConversationUsers role
may access the Conversation web service.</description>

      <url-pattern>/async/Conversation.jws/*</url-pattern>

      <http-method>GET</http-method>

      <http-method>POST</http-method>

    </web-resource-collection>

    <auth-constraint>

      <role-name>ConversationUsers</role-name>

    </auth-constraint>

  </security-constraint>

  <login-config>

    <auth-method>CLIENT-CERT</auth-method>

  </login-config>
```

```

    <security-role>
      <description>Role description</description>
      <role-name>ConversationUsers</role-name>
    </security-role>
  </web-app>

```

You can use any role to protect your web resource collection. You want to make the corresponding security and run-as role assignments in your `weblogic.xml` descriptor as needed. For the previous example, the `weblogic.xml` file contains:

```

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE weblogic-web-app
  PUBLIC "-//BEA Systems, Inc.//DTD Web Application 7.0//EN"
  "http://www.bea.com/servers/wls700/dtd/weblogic700-web-jar.dtd" >
<weblogic-web-app>

  <security-role-assignment>
    <role-name>ConversationUsers</role-name>
    <principal-name>weblogic</principal-name>
  </security-role-assignment>

</weblogic-web-app>

```

For more information on configuring protected resources for WebLogic Server, see [Securing WebLogic Resources](#).

Configuring the Active Directory Authentication

This procedure requires the use of the following Active Directory utilities:

- `setspn` – found on the Windows 2000 Resource Kit
- `ktpass` – found on the Windows 2000 distribution CD in `\Program Files\Support Tools`

The first three steps assume you have two domains: one represents the WebLogic application server domain (`bea.com`) and one controlled by Active Directory (`magellan.corp`).

1. Create a user account for the hostname of the web server machine in Active Directory, by using the Active Directory Users and Computers Snap-in.

Click Start->Programs->Administrative Tools->Active Directory Users and Computers.

Use the simple name of the WebLogic server host. For example, if the host you are running the WebLogic application on is called `myhost.bea.com`, create a new user in Active Directory called `myhost`. Do not select "User must change password at next logon." Make a note of the password for use in step 3.

2. Create the Service Principal Names (SPNs) for this account:

```
setspn -A host/myhost.bea.com myhost
setspn -A HTTP/myhost.bea.com myhost
```

3. Create your user mapping and export the keytab files using the `ktpass` utility:

```
ktpass -princ host/myhost@MAGELLAN.CORP -pass <password> -mapuser myhost
-out c:\temp\myhost.host.keytab

ktpass -princ HTTP/myhost@MAGELLAN.CORP -pass <password> -mapuser myhost
-out c:\temp\myhost.HTTP.keytab
```

Note: If you generated the keytab files for a WebLogic server on a Unix host, copy the keytab files securely to the Unix host. Login as root and then merge them into a single keytab using the `ktutil` utility:

```
ktutil: "rkt myhost.host.keytab"
ktutil: "rkt myhost.HTTP.keytab"
ktutil: "wkt mykeytab"
ktutil: "q"
```

If your WebLogic server is running on a Windows platform, generate the keytab from that machine using the `ktab`.

4. Verify that authentication works.

- To verify that Kerberos authentication is working on the Unix system, run the `kinit` utility:

```
kinit -t mykeytab myhost
```

You are prompted for the password and if authentication succeeds, the command prompt returns without an error message.

- To verify that Kerberos authentication is working on a Windows system, use the `ktab` utility locally on the WebLogic server host to create the keytab file in the WebLogic server domain directory:

```
setEnv
ktab -k mykeytab -a myhost@MAGELLAN.CORP <password>
```

5. Create a JAAS login configuration file.

For either a Windows or a Unix server host, you need a JAAS login configuration file. You also need to set some system properties to direct WebLogic server to allow the proper Kerberos authentication to occur. A sample login configuration file called `krb5Login.conf` looks like this:

```
com.sun.security.jgss.initiate
{
com.sun.security.auth.module.Krb5LoginModule required principal=
"myhost@MAGELLAN.CORP" useKeyTab=true keyTab=mykeytab storeKey=true;
};
com.sun.security.jgss.accept
{
com.sun.security.auth.module.Krb5LoginModule required principal=
"myhost@MAGELLAN.CORP" useKeyTab=true keyTab=mykeytab storeKey=true;
};
```

6. Add the following system properties to the start line of your WebLogic server:

```
-Djava.security.krb5.realm=MAGELLAN.CORP
-Djava.security.krb5.kdc=ADhostname
-Djava.security.auth.login.config=krb5Login.conf
-Djavax.security.auth.useSubjectCredsOnly=false
```

For a web service client, complete the steps described in [“Configure the Client .NET Web Service” on page 8-6](#). For Internet Explorer configuration, complete the steps described in [“Configure the Internet Explorer Client Browser” on page 8-7](#).

Configure the Active Directory Authentication Provider

To populate groups properly in the authenticated subject and to use the keystores, you must configure the Active Directory Authentication Provider. For information on how to configure this provider, see [“Configuring an Active Directory Authentication Provider” on page 9-15](#).

Configure the Client .NET Web Service

To configure the client .NET web service:

1. Open the `web.config` file for the client web service.

2. Set the authentication mode to Windows for IIS and ASP.NET. This is usually the default.

```
<authentication mode="Windows" />
```

3. Add the statement needed for the web services client to pass to the proxy web service object so that the credentials are sent through SOAP.

For example, if you have a web service client for the conversation web service represented by the proxy object `conv`, then setting the web services client credentials in C# looks like this:

```
/*
 * Explicitly pass credentials to the Web Service
 */
conv.Credentials = System.Net.CredentialCache.DefaultCredentials;
```

Configure the Internet Explorer Client Browser

If you are configuring Internet Explorer, you must perform the following steps:

- [“Configure the Sites” on page 8-7](#)
- [“Configure Intranet Authentication” on page 8-8](#)
- [“Verify the Proxy Settings” on page 8-8](#)
- [“Set the Internet Explorer 6.0 Configuration Settings” on page 8-8](#)

Configure the Sites

To configure the sites:

1. In Internet Explorer, click Tools, and then click Internet Options.
2. Click the **Security** tab.
3. Click **Local intranet**.
4. Click **Sites**.
5. Ensure that **Include all sites that bypass the proxy server is checked, and then click Advanced**.
6. In the Local intranet (Advanced) dialog box, enter all relative domain names that will be used on the intranet (e.g. myhost.bea.com).
7. Click **OK** to close the dialog boxes.

Configure Intranet Authentication

To configure Intranet Authentication:

1. Click the **Security** tab, click **Local intranet**, and then click **Custom Level**.
2. In the Security Settings dialog box, scroll down to the User Authentication section of the list.
3. Select **Automatic logon only in Intranet zone**. This setting prevents users from having to re-enter logon credentials; a key piece to this solution.
4. Click **OK** to close the Security Settings dialog box.

Verify the Proxy Settings

To verify the Proxy Settings:

1. In Internet Explorer, click Tools, and then click Internet Options.
2. Click the **Connections** tab.
3. Click **LAN Settings**.
4. Verify that the proxy server address and port number are correct.
5. Click **Advanced**.
6. In the Proxy Settings dialog box, ensure that all desired domain names are entered in the **Exceptions** field.
7. Click **OK** to close the Proxy Settings dialog box.

Set the Internet Explorer 6.0 Configuration Settings

In addition to the previous settings, one additional setting is required if you are running Internet Explorer 6.0.

1. In Internet Explorer, click **Tools**, and then click **Internet Options**.
2. Click the **Advanced** tab.
3. Scroll down to the **Security** section.
4. Make sure that **Enable Integrated Windows Authentication (requires restart)** is checked, and then click **OK**.

5. If this box was not checked, restart the browser.

Security Configuration

Overview

This topic describes how to use security providers to implement your security policies and how to configure these providers. Before you can configure providers, you must install the Security Service Module, instantiate the providers for that module, and then assign a Security Service Module configuration. You must also configure the Service Control Manager. For a step-by-step description of the process, see [“Security Configuration” on page 9-1](#).

Security Configuration

You must follow these steps to initialize a new Security Service Module and configure your providers.

1. [“Configuring a Service Control Manager” on page 9-2](#)
2. [“Configuring a Security Service Module” on page 9-4](#)
3. [“Binding a Security Service Module to a Service Control Manager” on page 9-8](#)
4. [“Configuring Security Providers” on page 9-9](#)
5. [“Distributing Configuration” on page 11-4](#)

Note: Changes made to a provider do not take effect until after it is explicitly deployed and the associated Security Service Module is restarted.

Understanding the Service Control Manager

The Service Control Manager is an essential component for remote administration of the BEA WebLogic Enterprise Security services. The Administration Application uses the Service Control Manager to distribute Security Service Module configurations throughout the enterprise—to all modules embedded in applications, application servers, and web servers.

To facilitate the management of a potentially large number of distributed Security Service Modules, the Administration Application uses a provisioning mechanism to deploy configuration data to each server containing a Service Control Manager, which in turn passes that configuration data to each module it controls. The Administration Application can deploy configurations to many different machines, but each machine has only one Service Control Manager. When you start each module, the module passes its Configuration ID to the Service Control Manager instance running on the same machine, and thereafter, the Service Control Manager assumes management of the module.

The Service Control Manager receives and stores configuration updates, both full and incremental, initiated by the Administration Application. Each module receives its initial configuration data from the Service Control Manager instance running on the same machine. When a configuration change relevant to a module is made by the Administration Application, it is provisioned to the Service Control Manager through the Policy Distributor. The provisioning mechanism ensures that only the configuration data absolutely required by a Service Control Manager is provisioned to that module. Likewise, the Service Control Manager ensures that only the configuration data absolutely required by a managed module is made available to that module. For a description of the BEA WebLogic Enterprise Security architecture, see the [Introduction to WebLogic Enterprise Security](#).

Configuring a Service Control Manager

A Service Control Manager maintains the configuration data that each Security Service Module retrieves. It receives this information from the Administration Application whenever you distribute the configuration changes.

Before you begin, the Service Control Manager you want to initialize must be installed, and you must know the Configuration Name and Configuration ID used when it was installed. If you specify this incorrectly in the console, the Service Control Manager will not be able to receive any configuration data.

To configure a Service Control Manager:

1. Open the Security Configuration folder.
2. Open the Service Control Managers folder, and click Service Control Manager.

The Service Control Managers page displays the General tab, containing the name of each Service Control Manager configuration available.

3. Do one of the following:
 - To create a new one, click Configure a new Service Control Manager, enter a name for the configuration, and then click Create.

The Name you assign must match the name of the Service Control Manager that you entered during the installation of a Security Service Module. The Administration Application distributes the configuration changes to Service Control Manager for all Security Service Module configurations that are bound to that specific Service Control Manager. If there is a mismatch in the names of the Service Control Manager or if there is a mismatch in the Configuration IDs, then the Service Control Manager will not receive the configuration changes for the Security Service Module.

Note: You must bind each Security Service Module configuration to a Service Control Manager configuration before you can distribute it.

- To bind the Service Control Manager to a Security Service Module, click the Binding tab and select the Configuration Name.

The Service Control Managers page displays the name of the Service Control Manager configuration in the configuration table. Additionally, the name of the new Service Control Manager is added as a folder under the Service Control Managers folder in the navigation pane.

Note: You must distribute the configuration before it becomes available to the Security Service Module. For instructions on distribution configuration, see [“Distributing Configuration” on page 11-4](#).

Understanding the Security Service Module

BEA WebLogic Enterprise Security supports a variety of Security Service Modules that you integrate with the security framework and provision as needed. You may instantiate and configure as many modules as you need to secure the enterprise, and manage them directly through a central Administration Application. The distributed nature of the architecture allows you to configure, manage and distribute configuration and policy throughout the enterprise.

BEA WebLogic Enterprise Security provides the following types of Security Service Modules:

- WebLogic Server 8.1 Security Service Module
- Internet Information Services (IIS) Web Server Security Service Module
- Apache Web Server Security Service Module
- Web Services Security Service Module
- Java Security Service Module

Configuration data for each module is maintained within each machine and handled by a Service Control Manager. One additional benefit of this architecture is that even if the Administration Server goes down (either for maintenance or failure), there is no impact on the applications or security services provided by those modules. For a complete description of the BEA WebLogic Enterprise Security architecture, see the [Introduction to WebLogic Enterprise Security](#).

Configuring a Security Service Module

Before you can use a new Security Service Module, you must first initialize it through the Administration Console. Initialization of a Security Service Module involves:

- Defining the set of providers used by the Security Service Module
- Binding the Security Service Module configuration to a Service Control Manager configuration
- Distributing the configuration data to all Security Service Modules with the name to which the Security Service Module configuration is bound.

After you initialize the module and distribute the configuration, the Security Service Module can be started and the providers initialized with the specified configuration. You can have multiple Security Service Modules managed by a single Service Control Manager.

Note: Before you begin, the module you want to initialize must be installed and you must know the Configuration ID and Configuration Name used when the module was installed. If you enter this information incorrectly in the console, the Security Service Module will not receive any configuration data or may receive the incorrect configuration. For a description of the BEA WebLogic Enterprise Security architecture, see the [Introduction to WebLogic Enterprise Security](#).

To initialize a Security Service Module:

1. Open the Security Configuration folder.

2. Click Unbound Configurations.

The Unbound Security Service Module Configurations pages appears.

3. Click Create a new Security Service Module Configuration.

The Create a Security Service Module Configuration page appears.

4. On the General tab, in the Configuration ID text box, enter the identifier that was entered when this module was installed.

5. In the Description text box, enter a description for the Security Service Module configuration.

6. Click **Create**.

The Edit Security Service Module Configuration page displays the Configuration ID and Description you entered and a new Security Service Module configuration appears under the Unbound Configurations folder in the navigation pane.

7. Click the Providers tab.

The Providers page displays six tabs, one for each type of provider available for configuration. [Table 9-1](#) lists the name of each provider along with a description of each one.

Table 9-1 Security Providers

Provider Type	Provider Name	Description
Adjudicators	ASI Adjudication Provider	Provides an authorization decision based on results from one or more authorization providers.
Auditors	Log4j Audit Channel Provider	Processes audit event data and stores it using the Apache Log4j logging libraries.
	Resource Deployment Audit Provider	Publishes WebLogic resources to the Administration Application.
Authenticators and Identity Asserters	WebLogic Authentication Provider	Supports authentication through an embedded LDAP Directory for WebLogic Server.

Table 9-1 Security Providers (Continued)

Provider Type	Provider Name	Description
	ALES Identity Asserter Provider	Supports web server authentication and Web single sign-on between web server Security Service Modules (SSMs) and also between web server SSMs and WebLogic Server 8.1 SSMs. Use this provider in conjunction with the ALES Credential Mapping provider.
	Database Authentication Provider	Supports authentication using a relational database.
	SAML Identity Assertion Provider	Supports identity assertion through Security Assertion Markup Language v1.0.
	Open LDAP Authentication Provider	Supports authentication based on Open LDAP.
	Active Directory Authentication Provider	Supports authentication based on Microsoft Active Directory.
	Windows NT Authentication Provider	Supports Microsoft Windows NT authentication.
	Netscape iPlanet Authentication Provider	Supports authentication based on the iPlanet LDAP.
	Novell Authentication Provider	Supports authentication based on Novell LDAP
	Single Pass Negotiate Identity Asserter	The Single Pass Negotiate Identity Assertion provider supports identity assertion using HTTP authentication tokens from the SPNEGO protocol. The provider supports the identity assertion using the Kerberos tokens contained within the SPNEGO token.
	X.509 Identity Assertion Provider	Supports identity assertion through an X.509 digital certificate, supporting ASN.1 encoding and decoding.

Table 9-1 Security Providers (Continued)

Provider Type	Provider Name	Description
Authorizers	WebLogic Authorization Provider	Provides enforcement of authorization for the security policy in WebLogic Server. The Authorization Provider returns an access decision that determines which resources are protected and if a particular user is allowed access to a resource.
	ASI Authorization Provider	Provides enforcement of authorization for the security policy with which it is used. The ASI Authorization Provider returns an access decision that determines which resources are protected and if a particular user is allowed access to a resource.
Credential Mappers	WebLogic Credential Mapping Provider	Provides authentication credentials for a user (username and password) for single sign-on use into another application or resource.
	ALES Credential Mapping Provider	Provides authentication credentials for a user (a ALES authentication assertion) for passing to another web server application. Use this provider in conjunction with the ALES Identity Assertion provider.
	SAML Credential Mapping Provider	Provides authentication credentials for a user (a SAML authentication assertion) for passing to another application.
Role Mappers	ASI Role Mapping Provider	Provides assignment to roles based on the security policy with which it is used. The ASI Role Mapping provider returns a set or roles granted to a user on a protected resource.
	WebLogic Role Mapping Provider	Provides assignment to roles based on the security policy defined for use with WebLogic Server 8.1. The WebLogic Role Mapping Provider returns a set or roles granted to a user on a protected resource.

Note: If the current security configuration does not meet your requirements, you can change the configuration properties (as permitted) for each provider. Or, you can create a new

provider as described in [Developing Security Providers for BEA WebLogic Enterprise Security](#).

8. Configure each provider according to your needs.

Note: You may configure the desired providers now or do it later. If you are using WebLogic Enterprise Security with the WebLogic Portal product, you must use a WebLogic Role Mapping provider with a WebLogic Authorization provider for authorization.

For general information on how to configure a provider, see [“Configuring Security Providers” on page 9-9](#).

9. Bind the service module to a Service Control Manager configuration as described in [“Binding a Security Service Module to a Service Control Manager” on page 9-8](#).

Note: Changes made to a provider do not take effect until after it is explicitly deployed and the associated Security Service Module is restarted.

Binding a Security Service Module to a Service Control Manager

Before you can use a Security Service Module, you must bind it to a Service Control Manager. You may have multiple modules of the same type or different types bound to the same Service Control Manager. Each module can have its own configuration with different providers and provider settings. To bind a configuration:

1. Open the Security Configuration folder.
2. Open the Service Control Managers folder, and click Service Control Manager.
The Service Control Manager page displays the name of each Service Control Manager Configuration available for binding.
3. From the Service Control Manager Configuration table, select the Service Control Manager to which you want to bind the Security Service Module.

The Edit a Service Control Manager Configuration page appears.

4. Click the Binding tab.
5. From the drop-down menu, select the Security Service Module, and then click **Bind**.


The Security Service Module configuration is added to the Configuration Name table.

Unbinding a Security Service Module from a Service Control Manager

You can change the configuration of a Security Service Module, for example, if you want to change the configuration or Service Control Manager to which the module is bound. To unbind a Security Service Module from the Service Control Manager:

1. Open the Security Configuration folder.
2. Open the Service Control Managers folder, and click Service Control Manager.
3. Select the Service Control Manager from which to unbind the Security Service Module.

The Edit a Service Control Manager Configuration page appears.

4. Click the Binding tab.
5. Select the configuration to unbind, and then click the Unbind icon .

Configuring Security Providers

When you configure a security provider, you assign values to the configuration properties for each provider. Each provider has at least two tabs: General and Details. The General tab displays the name of the provider, a description of what the provider is used for, and the version number. The Detail tab displays additional attributes that you can configure. Depending on the security provider you are configuring, there may be additional attributes available. Attributes vary depending on the provider you are configuring. Not all attributes are configurable.

Note: If you are using WebLogic Enterprise Security with the WebLogic Portal product, you must use a WebLogic Role Mapping provider with a WebLogic Authorization provider for authorization. For information on configuring the WebLogic providers, see [“Configuring a WebLogic Server Security Service Module” on page 9-47](#).

- [“Configuring an Authentication Provider” on page 9-10](#)
- [“Configuring an ALES Identity Assertion Provider” on page 9-28](#)
- [“Configuring a SAML Identity Assertion Provider” on page 9-29](#)
- [“Configuring an X.509 Identity Assertion Provider” on page 9-31](#)
- [“Configuring a Single Pass Negotiate Identity Asserter” on page 9-30](#)
- [“Configuring an ASI Authorization Provider” on page 9-38](#)

- [“Configuring an ASI Role Mapping Provider” on page 9-41](#)
- [“Configuring an ASI Adjudication Provider” on page 9-40](#)
- [“Configuring an ALES Credential Mapping Provider” on page 9-32](#)
- [“Configuring a Database Credential Mapper” on page 9-33](#)
- [“Configuring a SAML Credential Mapping Provider” on page 9-37](#)
- [“Configuring a Log4j Audit Channel Provider” on page 9-44](#)
- [“Configuring a Resource Deployment Audit Provider” on page 9-43](#)

Configuring an Authentication Provider

An Authentication Provider allows BEA WebLogic Enterprise Security to establish trust by validating a user. There must be at least one Authentication Provider in a Security Service Module configuration if it is required to perform authentication. The LDAP authentication providers are designed to access external LDAP stores.

The following authentication provider support is available: Open LDAP, Netscape iPlanet, Microsoft Active Directory and Novell NDS stores. You can also authenticate against a database or other kinds of repository, using the Database Authentication Provider or Microsoft Windows NT authentication.

Identity assertion involves establishing the client identity using client-supplied tokens that may exist outside of the request. Thus, the function of an identity assertion provider is to validate and map a token to a username by which a user can be validated. You can configure multiple identity assertion providers, but you only need one if you want to authenticate users. An identity assertion provider is used in conjunction with an authentication provider and, in the case of the ALES Identity Assertion provider, in conjunction with the ALES Credential Mapping provider.

The order in which the authentication providers are configured is the order in which each authentication provider is called. Therefore, the order in which the authentication and identity assertion providers are configured can affect the outcome of the authentication process. When one or more authentication providers are configured, the authentication provider tabs display key information about each one and allow you to control their order. Click the appropriate link to configure an Authentication Provider and an Identity Assertion Provider. If custom Authentication or Identity Assertion providers are available, you may be able to configure them as well. The ordering of the authentication providers can be changed at any time. For more information, see [“Changing the Order of Authentication Providers” on page 9-11](#).

Note: The WebLogic Enterprise Security AuthenticationService API has identity assertion functionality to translate WebLogic subjects into `AuthenticIdentity` objects for enhanced compatibility with WebLogic Servers. You may use this functionality or provide a custom identity assertion provider to handle the assertion of WebLogic subjects. For more information on this topic, see the *Programming Security for Java Applications* and the Javadocs for the AuthenticationService API and the methods it supports.

To configure Authentication providers, see the following links:

- [“Configuring an Active Directory Authentication Provider” on page 9-15](#)
- [“Configuring a Novell LDAP Authentication Provider” on page 9-20](#)
- [“Configuring an iPlanet LDAP Authentication Provider” on page 9-16](#)
- [“Configuring a Windows NT Authentication Provider” on page 9-14](#)
- [“Configuring an Open LDAP Authentication Provider” on page 9-13](#)
- [“Configuring a Database Authentication Provider” on page 9-23](#)
- [“Configuring an ALES Identity Assertion Provider” on page 9-28](#)
- [“Configuring a SAML Identity Assertion Provider” on page 9-29](#)
- [“Configuring a Single Pass Negotiate Identity Asserter” on page 9-30](#)
- [“Configuring an X.509 Identity Assertion Provider” on page 9-31](#)
- [“Configuring the WebLogic Authentication Provider” on page 9-49](#)

Changing the Order of Authentication Providers

The way you configure multiple authentication providers can affect the overall outcome of the authentication process, which is especially important for multi-part authentication.

Authentication providers are called in the order in which they are configured. The Authentication providers table lists the providers in the order they were configured. The setting of the Authentication provider Control Flag attribute effects the outcome of the authentication process. For more information, see [“Setting the JAAS Control Flag” on page 9-12](#).

To change the ordering of authentication providers:

1. Open the Security Configuration folder.

2. Open the Service Control Managers folder containing the Security Service Module whose providers you want to configure or change.
3. Open the Security Service Module folder that contains the providers you want to configure or change.
4. Open the Authentication folder, and click Authentication.
The Authentication page appears.
5. Click Re-order the Configured Authentication Providers.
The Edit Authentication Provider Order page appears.
6. Select the Authentication provider from the list of configured Authentication providers.
7. Use the arrow buttons to move the Authentication provider up or down in the list.
8. Click Apply to save your changes.

Note: Changes made to a provider do not take effect until after it is explicitly deployed and the associated Security Service Module is restarted.

Setting the JAAS Control Flag

If a policy domain has multiple authentication providers configured, the Control Flag attribute (located on the General tab for the Edit Authenticator page) determines the user authentication requirements for each configured authentication provider. Additionally, the order in which the authentication providers are configured in the Administration Console is the order in which the authentication providers are called. You can change the order of the authentication providers at any time.

The values for the Control Flag attribute are:

- **REQUIRED**—The Authentication provider is always called and the user must always pass its authentication test.
- **REQUISITE**—If the user passes the authentication test of this Authentication provider, other providers are executed but can fail (except for authentication providers with the JAAS Control Flag set to **REQUIRED**).
- **SUFFICIENT**—If the user passes the authentication test of the Authentication provider, no other Authentication providers are executed (except Authentication providers with the JAAS Control Flag set to **REQUIRED**) because the user was sufficiently authenticated.

- **OPTIONAL**—The user is allowed to pass or fail the authentication test of this Authentication provider. However, if all Authentication providers configured in a security realm have the JAAS Control Flag set to **OPTIONAL**, the user must pass the authentication test of at least one of the configured providers.

Note: If you define multiple authentication providers, to boot the server, the user from which the server is booted must be defined as a user in all the authentication providers that have the Control Flag attribute set to **REQUISITE** or **REQUIRED**. For more information, see [“Changing the Order of Authentication Providers” on page 9-11](#).

Configuring an Open LDAP Authentication Provider

To configure an Open LDAP Authentication provider:

1. Open the Security Configuration folder.
2. Open the Service Control Manager folder that contains the Security Service Module whose provider you want to configure or change.
3. Open the Security Service Module folder that contains the provider you want to configure or change.

Note: If you have not bound the Security Service Module, open the Unbound Configuration folder that contains the providers you want to configure.
4. Open the Authentication folder, and click Authentication.
The Authentication page appears.
5. Do one of the following:
 - To configure a new provider, click Configure a New Open LDAP Authenticator.
 - To change an existing provider, select the Open LDAP Authentication provider to configure.
6. Do one of the following:
 - If you are adding a new provider, on the General tab, assign a name for the provider, and then click Create.
 - If you are using multiple authentication providers, define a value for the Control Flag attribute on the General tab.
7. Click the OpenLDAP tab and edit the settings as needed.
8. Click the Users tab and edit the settings as needed.

9. Click the Groups tab and edit the settings as needed.
 10. Click the Membership tab and edit the settings as needed.
 11. Click the Details tab and define any additional attributes.
 12. Optionally, click the Failover tab and edit the settings as needed.
 13. On each tab you change, click Apply to save your changes.
- Note:** Changes made to a provider do not take effect until after it is explicitly deployed and the associated Security Service Module is restarted.

Configuring a Windows NT Authentication Provider

To configure the Windows NT Authentication provider:

1. Open the Security Configuration folder.
2. Open the Service Control Manager folder that contains the Security Service Module whose provider you want to configure or change.
3. Open the Security Service Module folder that contains the provider you want to configure or change.

Note: If you have not bound the Security Service Module, open the Unbound Configuration folder that contains the providers you want to configure.
4. Open the Authentication folder, and click Authentication.

The Authentication page appears.
5. Do one of the following:
 - To configure a new provider, click Configure a new NT Authenticator.
 - To change an existing provider, select the NT Authentication provider to configure.
6. Do one of the following:
 - If you are adding a new provider, on the General tab, assign a name for the provider, and then click Create.
 - If you are changing an existing provider, edit the settings on the General tab as needed, and then click Apply.
7. Click the Details tab and define any additional attributes.

8. Click Apply to save your changes.

Note: Changes made to a provider do not take effect until after it is explicitly deployed and the associated Security Service Module is restarted.

Configuring an Active Directory Authentication Provider

To configure for an Active Directory Authentication provider:

1. Open the Security Configuration folder.
2. Open the Service Control Manager folder that contains the Security Service Module whose providers you want to configure or change.
3. Open the Security Service Module folder that contains the providers you want to configure or change.

Note: If you have not bound the Security Service Module, open the Unbound Configuration folder that contains the providers you want to configure.

4. Open the Authentication folder, and click Authentication.

The Authentication page appears.

5. Do one of the following:
 - To configure a new provider, click Configure a new Active Directory Authenticator.
 - To change an existing provider, select the Active Directory Authentication provider to configure.
6. If you are adding a new provider, on the General tab, assign a name for the provider, and then click Create.
7. If you are using multiple authentication providers, define a value for the Control Flag attribute on the General tab. For more information, [“Setting the JAAS Control Flag” on page 9-12](#).
8. Click the Active Directory tab and edit the settings as needed.
9. Click the Users tab and edit the settings as needed.
10. Click the Groups tab and edit the settings as needed.
11. Click the Membership tab and edit the settings as needed.
12. Click the Details tab and define any additional attributes.

13. Click the Failover tab and edit the settings as needed.

14. On each tab you change, click Apply to save your changes.

Note: Changes made to a provider do not take effect until after it is explicitly deployed and the associated Security Service Module is restarted.

Configuring an iPlanet LDAP Authentication Provider

To configure a Netscape iPlanet LDAP Authentication provider:

1. Open the Security Configuration folder.
2. Open the Service Control Manager folder that contains the Security Service Module whose providers you want to configure or change.
3. Open the Security Service Module folder that contains the providers you want to configure or change.

Note: If you have not bound the Security Service Module, open the Unbound Configuration folder that contains the providers you want to configure.

4. Open the Authentication folder, and click Authentication.

The Authentication page appears.

5. Do one of the following:
 - To configure a new provider, click Configure a new iPlanet Authenticator.
 - To change an existing provider, select the iPlanet LDAP provider to configure.
6. If you are adding a new provider, on the General tab, assign a name for the provider, and then click Create.

If you are using multiple authentication providers, define a value for the Control Flag attribute on the General tab. For more information, [“Setting the JAAS Control Flag” on page 9-12](#).

7. Click the iPlanet LDAP tab and edit the settings as needed.
8. Click the Users tab and edit the settings as needed.
9. Click the Groups tab and edit the settings as needed.
10. Click the Membership tab and edit the settings as needed.
11. Click the Details tab and define any additional attributes.

12. Click the Failover tab and edit the settings as needed.

For additional information on configuring failover for the LDAP provider, see [“Configuring Failover for LDAP Authentication Providers” on page 9-17](#).

13. Click **Apply** to save your changes.

Note: Changes made to a provider do not take effect until after it is explicitly deployed and the associated Security Service Module is restarted.

Configuring Failover for LDAP Authentication Providers

You can configure any of the LDAP providers with multiple LDAP servers and enable failover if one LDAP server is not available. To configure failover of the LDAP servers configured for an LDAP Authentication provider, perform the following steps:

1. Open the Security Configuration folder.
2. Open the Service Control Manager folder that contains the Security Service Module whose provider you want to configure or change.
3. Open the Security Service Module folder that contains the provider you want to configure or change.

Note: If you have not bound the Security Service Module, open the Unbound Configuration folder that contains the providers you want to configure.

4. Open the Authentication folder, click Authentication, and select the LDAP Authenticator you want to configure.
5. Click the Failover tab.
6. Check the Automatic Failover Enabled option to enable failover.
7. In the Backup Host text box, enter the name or IP address of the backup LDAP server. For example:

`directory.knowledge.com:1050 people.catalog.com 199.254.1.2`

8. In the Backup Port text box, enter the port number on which the backup LDAP server is listening.
9. To enable a secure connection when connecting to the backup LDAP server, check Backup SSL Enabled.

10. In the Backup Principal text box, enter the distinguished name (DN) of the LDAP user that is used by WebLogic Server to connect to the backup LDAP server.
11. In the Backup Credential and Confirm Backup Credential text boxes, enter the credential (generally a password) used to authenticate the backup LDAP user that is defined in the Principal attribute.
12. In the Primary Retry Interval text box, enter the length of the interval, in seconds, that the backup LDAP server uses before retrying the primary LDAP server.

This option specifies the number of seconds to delay when making concurrent attempts to connect to multiple servers.

13. Click Apply.
14. Click Details.
15. Set the Connection Timeout attribute.

Specify the maximum number of seconds to wait for the connection to the LDAP server to be established. If the attribute is set to 0, there is no maximum time limit and WebLogic Server waits until the TCP/IP layer times out to return a connection failure. This attribute may be set to a value over 60 seconds depending upon the configuration of TCP/IP.

16. Click Apply.

Note: Changes made to a provider do not take effect until after it is explicitly deployed and the associated Security Service Module is restarted.

Refer to [Table 9-2](#) and [Table 9-3](#) for two examples that describe what can occur when the LDAP attributes are set for LDAP failover. In the first example, WebLogic Server attempts to connect to `directory.knowledge.com`. After 10 seconds, the connect attempt times out and WebLogic Server attempts to connect to the next host specified in the Host attribute (`people.catalog.com`). WebLogic Server then uses `people.catalog.com` as the LDAP Server for this connection.

Table 9-2 Example 1: LDAP Attributes and Failover

LDAP Attribute	Value
Host	directory.knowledge.com:1050 people.catalog.com 199.254.1.2 The LDAP servers are working as follows: directory.knowledge.com:1050 is down people.catalog.com is up 199.254.1.2 is up
Parallel Connect Delay	0
Connect Timeout	10

In the following example, WebLogic Server attempts to connect to `directory.knowledge.com`. After 1 second, the connect attempt times out and WebLogic Server tries to connect to the next host specified in the Host attribute (`people.catalog.com`) and `directory.knowledge.com` in parallel. If the connection to `people.catalog.com` succeeds, WebLogic Server uses `people.catalog.com` as the LDAP Server for this connection. WebLogic Server cancels the connection to `directory.knowledge.com` after the connection to `people.catalog.com` succeeds.

Table 9-3 Example 2: LDAP Attributes and Failover

LDAP Attribute	Value
Host	directory.knowledge.com:1050 people.catalog.com 199.254.1.2 The LDAP servers are working as follows: directory.knowledge.com:1050 is down people.catalog.com is up 199.254.1.2 is up
Parallel Connect Delay	1
Connect Timeout	10

Configuring a Novell LDAP Authentication Provider

To configure a Novell LDAP Authentication provider:

1. Open the Security Configuration folder.
2. Open the Service Control Manager folder that contains the Security Service Module whose providers you want to configure or change.
3. Open the Security Service Module folder that contains the providers you want to configure or change.

Note: If you have not bound the Security Service Module, open the Unbound Configuration folder that contains the providers you want to configure.

4. Open the Authentication Providers folder, and click Authentication.
The Authentication page appears.
5. Do one of the following:
 - To configure a new provider, click Configure a new Novell Authenticator.
 - To change an existing provider, select the Novell Authentication provider to configure.
6. If you are adding a new provider, on the General tab, assign a name for the provider, and then click Create.

7. If you are using multiple authentication providers, define a value for the Control Flag attribute on the General tab. For more information, [“Setting the JAAS Control Flag” on page 9-12.](#)
8. Click the Novell LDAP tab and edit the settings as needed.
9. Click the Users tab and edit the settings as needed.
10. Click the Groups tab and edit the settings as needed.
11. Click the Membership tab and edit the settings as needed.
12. Click the Details tab and define any additional attributes.
13. Click the Failover tab and edit the settings as needed.

For additional information on configuring failover for the LDAP provider, see [“Configuring Failover for LDAP Authentication Providers” on page 9-17.](#)

14. Click Apply to save your changes.

Note: Changes made to a provider do not take effect until after it is explicitly deployed and the associated Security Service Module is restarted.

Configuring Failover for the Database Authentication Provider

If you have an authentication backup database store, you can configure the Failover tab to enable the failover capability. Database replication procedures are provided to create a replica of the metadirectory (both primary and backup). For additional information, see [“Failover and System Reliability.”](#)

To configure a Database Authentication provider for failover:

1. Open the Security Configuration folder.
2. Open the Service Control Manager folder that contains the Security Service Module whose provider you want to configure or change.
3. Open the Security Service Module folder that contains the provider you want to configure or change.

Note: If you have not bound the Security Service Module, open the Unbound Configuration folder that contains the provider you want to configure.

4. Open the Authentication Providers folder, and click Authentication.

The Authentication page appears.

5. Select the Database Authentication provider to configure.
6. Click the Failover tab.
7. Check Enable Automatic Failover.

A check in the attribute Enable Automatic Failover enables the failover capability. When enabled, the provider connects to backup database when the primary database is unavailable.

8. In the Primary Retry Interval text box, enter the value in seconds to wait before checking the primary pool (for the purpose of failing back to the original source).

The provider retries the primary database connection to the primary database when the period of time as set in Primary Retry Interval is reached.

9. In the Backup Database URL text box, type the name of the URL assigned to the backup database when you configured the backup metadirectory.

The backup database must be of the same type as the primary (Oracle or Sybase), because the backup connection uses the same database JDBC driver. The backup connection is specified by the Backup Database URL.

10. In the Backup Database User Name text box, type the user name required to access the backup metadirectory.

In many case, the Backup Database User Name is the same as user name for the primary database. If it is different, you or your DBA need to make sure that this user has read access to the tables specified in the three SQL queries on the Details tab.

11. In the Backup Database Password and Confirm Backup Database Password text boxes, type the password required to access the backup metadirectory.

12. In the Backup Database Properties text box, enter any `NAME=VALUE` properties to use when obtaining the JDBC connection to the backup database.

13. Click Apply to save your changes.

Note: Changes made to a provider do not take effect until after it is explicitly deployed and the associated Security Service Module is restarted.

Configuring a Database Authentication Provider

You can configure a Database Authentication provider by configuring a database, including the policy database as the source of the authentication data.

To configure a Database Authentication provider:

1. Open the Security Configuration folder.
2. Open the Service Control Managers folder that contains the Security Service Module whose provider you want to configure or change, and click Service Control Manager.
3. Open the Security Service Module folder that contains the provider you want to configure or change.

Note: If you have not bound the Security Service Module, open the Unbound Configuration folder that contains the providers you want to configure.

4. Open the Authentication Providers folder, and click Authentication.

The Authentication page appears.

5. Do one of the following:
 - To configure a new provider, click Configure a new Database Authenticator.
 - To change an existing provider, select the Database Authentication provider to configure.
6. Do one of the following:
 - If you are adding a new provider, on the General tab, assign a name for the provider, and then click Create.
 - If you are changing an existing provider, edit the settings on the General tab as needed, and then click Apply.
7. The Control Flag attribute determines the order of execution. If you are using multiple authentication providers, define a value for the Control Flag attribute on the General tab. For more information, see [“Setting the JAAS Control Flag” on page 9-12](#).
8. Click the Details tab and define any additional attributes.

For both Oracle or Sybase, set the DatabaseUserLogin and DatabaseUserPassword configuration attributes to a valid database login for your database with read access to the tables that involves the authentication process.

The JDBC configuration is different depending on which database your system is configured to use and which type of JDBC driver you want to configure. For details on the Oracle setup, see [“Oracle Database Configuration” on page 9-24](#). For details on the Sybase setup, see [“Sybase Database Configuration” on page 9-25](#).

9. Optionally, to configure database failover, click the Failover tab and define any additional attributes.

For additional information, see [“Configuring Failover for the Database Authentication Provider” on page 9-21](#).

10. Click Apply to save your changes.

Note: Changes made to a provider do not take effect until after it is explicitly deployed and the associated Security Service Module is restarted.

Oracle Database Configuration

To configure Oracle, you must know the hostname of the database server, the port number and Oracle SID to which to connect. Each Oracle service is identified by a global database ID or SID. An Oracle system identifier or SID is a unique identifier for a database or a unique identifier for a node within an Oracle Parallel Server Database. From the client perspective, the database service names is the Local Net Service Name For the Security Service Module to use Oracle OCI client, the Oracle client (OCI library) must be installed and configured in the same machine as the security Service Module.

The following setting is the default value for an Oracle configuration:

```
JDBCClassName = oracle.jdbc.driver.OracleDriver
```

Oracle Thin Client (Type IV)

```
JDBCConnectionUrl = jdbc:oracle:thin:@<namehostname or IP  
address>:<port>:<sid>
```

Oracle OCI Client (Type II)

```
JDBCConnectionUrl = jdbc:oracle:oci8:@<database service name>
```

Note: For best performance, BEA recommends you use the native (Type II) database driver configuration. For more information on database connection configuration, please refer to the documentation specific to your database.

Sybase Database Configuration

To configure Sybase, you must know the hostname of the database server and the port number to which to connect. The following setting is the default value for a Sybase configuration. If the `<database name>` is not specified, the default database for DatabaseUserLogin is used:

```
JDBCClassName = com.sybase.jdbc2.jdbc.SybDriver
```

Sybase Thin Client (Type IV)

```
JDBCConnectionUrl = jdbc:sybase:Tds:<hostname or IP address>:<port>
```

or

```
JDBCConnectionUrl = jdbc:sybase:Tds:<hostname or IP  
address>:<port>/<database name>
```

The `JDBC Connection Properties` allows you to configure additional properties specific to your database server. These include SSL connection properties or other parameters as specified in your database documentation. You must enter parameters as a comma-delimited list of `NAME=VALUE` pairs.

Use the default values for the `ConnectionPoolCapacity` and `ConnectionPoolTimeout` configuration attributes or set them to meet your own requirements. You may want to increase the `ConnectionPoolCapacity` value to equal the number of threads running in your server.

The `VerifyUserForIdentityAssertion` attribute is not required unless you are using identity assertions and want to validate those users in the database.

The `AddGroupsFromIdentityAssertion` attribute enables that the groups obtained from identity assertion provider be added to the authenticated user.

The `AddGroupsFromLocalDBMS` attribute enables that the groups retrieved from this authentication database using the `SQLQueryToRetrieveGroups` be added to the authenticated user.

Specifying SQL Query Strings and Provider Extensions

Configure your SQL query strings to correspond to the database schema for your authentication tables. The user and group name format and password hashing algorithm require the provider extension class be consistent with the query. For additional information, see [“Provider Extensions.”](#)

Metadirectory Configuration

If you want to configure the provider to authenticate against the metadirectory, use the following settings:

Note: For the SQL query strings, prefix each table reference with the schema owner. If the login id used defaults to the correct schema owner, then this attribute is optional.

SQL Query to Verify User (SQLQueryToVerifyUser)

```
select userid from <schema owner>.adminuser where status = 'A' and userid = ?
```

SQL Query to Retrieve Password (SQLQueryToRetrievePassword)

```
select password from <schema owner>.adminuser where status = 'A' and userid  
= ?
```

SQL Query to Retrieve Groups (SQLQueryToRetrieveGroups)

```
SELECT r.qualified FROM <schema owner>.subject_curr r, <schema  
owner>.sgrpmemberexp_curr rm, <schema owner>.subject_curr m WHERE  
m.qualified = ? AND m.id = rm.member_id AND rm.sgrp_id = r.id AND r.subj_type  
= 'R'
```

Each query takes single input parameter. The question mark (?) is the place holder for an input parameter and it takes the formatted user name. In the metadirectory, the formatted user name contains a qualifier prefix, the `IdentityScope`, and the user id.

The `IdentityScope` attribute is required and must specify the correct directory containing the users you want to authenticate. Directory is a scoping term in the metadirectory. You cannot authenticate through multiple directories using a single Database Authentication provider; you can specify one and only one directory for this provider.

Note: When using the Database Authentication Provider with the ASI Authorization and ASI Role Mapping providers, the `IdentityScope` must match the `IdentityDirectory` set in the ASI Authorization and ASI Role Mapping providers.

The user password is stored in the metadirectory by hashing the clear text using SHA1 hashing algorithm. Together, with the specific user and group name format, you need to specify:

```
com.bea.security.providers.authentication.dbms.DefaultDBMSPluginImpl
```

as the value for the `AuthenticationPluginClass` attribute.

You can set the `GroupMembershipSearching` attribute to `limited` or `unlimited` and `MaxGroupMembershipSearchLevel` to any number. These two attributes have no effect on the

metadirectory, because the `SQLQueryToRetrieveGroups` retrieves all groups of all levels for the authenticated user.

Other Database Stores

When configuring other database stores, you are required to implement your own authentication plug-in class, since the password in your store may be hashed differently, or the password may be in clear text and user/group name is not formatted the same way. The `IdentityScope` attribute is optional and only required if using a database authentication plug-in that recognizes scope. It is very easy to implement your own plug-in class. For additional information, see [“Provider Extensions.”](#)

Note: When using the Database Authentication Provider with the ASI Authorization and ASI Role Mapping providers, the `IdentityScope` must match the `IdentityDirectory` set in the ASI Authorization and ASI Role Mapping providers.

The `AuthenticationPluginClass` attribute should be set to the Java class name for your plug-in implementation. The `IdentityScope` attribute should be set to the value that your plug-in understands.

The authentication provider uses compiled JDBC statements based on the specified SQL query strings. All three query strings are populated with the authenticating user identifier at runtime. To specify where in the string to populate the id, use the question mark (?) symbol. Here is a sample SQL query:

```
SELECT password FROM Principals WHERE principal = ?
```

SQL queries for the Database Authentication Provider only accept a single input parameter, where the question mark (?) place holder is for the encrypted password. This parameter is the formatted user name performed in your plug-in implementation class. A query requiring multiple input parameters or no input parameters results in a runtime error and users are not authenticated. Your database administrator can assist you in determining the correct SQL syntax for your database.

Each query should return only one column of data in the `SELECT` statement.

`SQLQueryToVerifyUser` returns a matching user name, `SQLQueryToRetrievePassword` returns a password, and `SQLQueryToRetrieveGroups` returns formatted groups. After their retrieval, the group names are un-formatted in your plug-in implementation class before they are associated with the authenticated user.

If your database store supports hierarchical (or recursive) groups, you can set attributes `GroupMembershipSearching` and `MaxGroupMembershipSearchLevel` to the values of your choice. When recursive group searching is enabled, the formatted group is used as the input parameter in `SQLQueryToRetrieveGroups` to retrieve formatted groups of next level.

If the authentication plug-in is specified, the `SQLQueryToRetrievePassword` query string is optional. If this string is left blank, the plug-in must handle the retrieval of user's password. See "Provider Extensions" in the *BEA WebLogic Enterprise Security Administration Guide*, for more information on plug-in configuration.

Configuring an ALES Identity Assertion Provider

The ALES Identity Assertion provider is used in Web Server Security Service Modules (SSMs) and WebLogic 8.1 SSMs for authentication and Web single sign-on (SSO). You use this provider in conjunction with the ALES Credential Mapping provider. If you configure one, you must also configure the other.

To configure an ALES Identity Assertion provider:

1. Open the Security Configuration folder.
2. Open the Service Control Manager folder that contains the Security Service Module whose providers you want to configure or change.
3. Open the Security Service Module folder that contains the providers you want to configure or change.
Note: If you have not bound the Security Service Module, open the Unbound Configuration folder that contains the providers you want to configure.
4. Open the Authentication Providers folder, and click Authentication.
The Authentication page appears.
5. Do one of the following:
 - To configure a new provider, click Configure a new ALES Identity Asserter.
 - To change an existing provider, select the ALES Identity Asserter to configure.
6. Do one of the following:
 - If you are adding a new provider, on the General tab, assign a name for the provider, and then click Create.
 - If you are changing an existing provider, edit the settings on the General tab as needed, and then click Apply.
7. Click the Details tab and configure the attributes. The certificate and keystore attributes are required. If you are configuring this provider for an IIS or Apache Web Server Security Service Module (SSM), the setting of the Base64 Decoding Required attribute has no effect, so the setting does not matter. However, if you are using this provider with a WebLogic Server 8.1 SSM to support web single sign-on with Web Server SSMs, leave the Base64 Decoding Required attribute set to unchecked, which is the default setting.
8. Click Apply to save your changes.

Note: Changes made to a provider do not take effect until after it is explicitly deployed and the associated Security Service Module is restarted.

Configuring a SAML Identity Assertion Provider

To configure a SAML Identity Assertion provider:

1. Open the Security Configuration folder.
2. Open the Service Control Manager folder that contains the Security Service Module whose providers you want to configure or change.
3. Open the Security Service Module folder that contains the providers you want to configure or change.

Note: If you have not bound the Security Service Module, open the Unbound Configuration folder that contains the providers you want to configure.

4. Open the Authentication Providers folder, and click Authentication.

The Authentication page appears.

5. Do one of the following:
 - To configure a new provider, click Configure a new SAML Identity Asserter.
 - To change an existing provider, select the SAML Identity Asserter to configure.
6. Do one of the following:
 - If you are adding a new provider, on the General tab, assign a name for the provider, and then click Create.
 - If you are changing an existing provider, edit the settings on the General tab as needed, and then click Apply.

The following types of SAML Assertions are supported:

`SAML.Challenge`—The token is a Challenge token that has implemented the `com.bea.security.internal.SAMLChallenge` interface.

`SAML.Assertion`—SAML Assertion that contains the full CertPath. Same as `SAML.Assertion.Certpath`.

7. Click the Details tab and define any additional attributes.
8. Verify the setting of the Base64 Decoding Required attribute.

If the authentication type in a web application is set to `CLIENT-CERT`, the web application container in WebLogic Server performs identity assertion on values from request headers and cookies. If the header name or cookie name matches the active token type for the configured Identity Assertion provider, the value is passed to the provider.

The Base64 Decoding Required attribute determines whether the request header value or cookie value must be decoded using Base64 before sending it to the Identity Assertion provider. The setting is enabled by default for purposes of backward compatibility, however, for most Identity Assertion providers, you can disable this attribute.

9. Configure the required keystore attributes.

10. Click Apply to save your changes.

Note: Changes made to a provider do not take effect until after it is explicitly deployed and the associated Security Service Module is restarted.

Configuring a Single Pass Negotiate Identity Asserter

The Single Pass Negotiate Identity Assertion provider supports identity assertion using HTTP authentication tokens from the SPNEGO protocol. The provider supports the identity assertion using the Kerberos tokens contained within the SPNEGO token.

To configure a Single Pass Negotiate Identity Assertion provider:

1. Open the Security Configuration folder.
2. Open the Service Control Manager folder that contains the Security Service Module whose providers you want to configure or change.
3. Open the Security Service Module folder that contains the providers you want to configure or change.

Note: If you have not bound the Security Service Module, open the Unbound Configuration folder that contains the providers you want to configure.

4. Open the Authentication Providers folder, and click Authentication.

The Authentication page appears.

5. Do one of the following:
 - To configure a new provider, click Configure a new Single Pass Negotiate Identity Asserter.
 - To change an existing provider, select the Single Pass Negotiate Identity Asserter to configure.
 6. Do one of the following:
 - If you are adding a new provider, on the General tab, assign a name for the provider, and then click Create.
 - If you are changing an existing provider, edit the settings on the General tab as needed, and then click Apply.
 7. For Active Types, check to ensure that `Authorization` is already set in the Active Types Chosen list box.

Note: For compatibility, the `SPNEGO.AtnAssertion` active type is also listed in the Chosen list box, however, it is no longer used.
 8. Click Apply to save your changes.
- Note:** Changes made to a provider do not take effect until after it is explicitly deployed and the associated Security Service Module is restarted.

Configuring an X.509 Identity Assertion Provider

To configure an X.509 Identity Assertion provider:

1. Open the Security Configuration folder.
 2. Open the Service Control Manager folder that contains the Security Service Module whose providers you want to configure or change.
 3. Open the Security Service Module folder that contains the providers you want to configure or change.

Note: If you have not bound the Security Service Module, open the Unbound Configuration folder that contains the providers you want to configure.
 4. Open the Authentication Providers folder, and click Authentication.
- The Authentication page appears.

5. Do one of the following:
 - To configure a new provider, click Configure a new X.509 Identity Asserter.
 - To change an existing provider, select the X.509 Identity Asserter to configure.
 6. Do one of the following:
 - If you are adding a new provider, on the General tab, assign a name for the provider, and then click Create.
 - If you are changing an existing provider, edit the settings on the General tab as needed, and then click Apply.
 7. In the `User Name Mapper Class Name` attribute, define the name of the Java class that maps X.509 digital certificates and X.501 distinguished names to WebLogic Enterprise Security user names. Additional configuration is available on the Details tab.
 8. In the `Trusted Client Principals` attribute define the list of client principals that can use CSIV2 identity assertion. You can use an asterisk (*) to specify all client principals.
 9. Define the active token type for the X.509 Identity Assertion provider. The list of token types supported by the Identity Assertion provider is displayed in the Available list box. To define the active token type for the X.509 Identity Assertion provider:
 - a. In the Available list box, select the desired token type.
 - b. Click the right arrow to move token type to the Chosen list box.
 10. Click Apply to save your changes.
 11. Click the Details tab and define any additional attributes.
 12. Click Apply to save your changes.
- Note:** Changes made to a provider do not take effect until after it is explicitly deployed and the associated Security Service Module is restarted.

Configuring an ALES Credential Mapping Provider

The ALES Credential Mapping provider is used in conjunction with the ALES Identity Assertion provider. If you configure one, you must also configure the other.

To configure an ALES Credential Mapping provider:

1. Open the Security Configuration folder.

2. Open the Service Control Manager folder that contains the Security Service Module whose providers you want to configure or change.
3. Open the Security Service Module folder that contains the providers you want to configure or change.
Note: If you have not bound the Security Service Module, open the Unbound Configuration folder that contains the providers you want to configure.
4. Open the Credential Mapping folder, and click Credential Mapping.
The Credential Mapping page appears.
5. Do one of the following:
 - To configure a new provider, click Configure a new ALES Credential Mapper.
 - To change an existing provider, select the ALES Credential Mapper to configure.
6. Do one of the following:
 - If you are adding a new provider, on the General tab, assign a name for the provider, and then click Create.
 - If you are changing an existing provider, edit the settings on the General tab as needed, and then click Apply.
7. Click the Details tab and define the certificate and keystore attributes.
8. Click Apply to save your changes.

Note: Changes made to a provider do not take effect until after it is explicitly deployed and the associated Security Service Module is restarted.

Configuring a Database Credential Mapper

The Database Credential Mapper uses a two-way cipher secured by a pass-phrase or shared secret. By using a shared secret rather than an asymmetric key like 3DES (which requires a keystore), the provider configuration is self-contained and requires no external files. Another by-product of this mechanism is that you can encrypt your own passwords, by-passing the Administration Console, provided you know the shared secret and use the same JCE packaging as BEA WebLogic Enterprise Security.

The standard JCE Password-Based encryption algorithm `PBEWithMD5AndDES` is used. This creates a secret key from a salt, an iteration, and a pass-phrase/shared secret. In conjunction with

the CIPHER of the same name, this method encrypts the password in the database preventing any party who does not know the shared secret to decrypt it.

The salt is eight bytes that are randomly generated. The iteration is a random value from 1 to 32:

```
[8 byte salt in clear] [1 byte iteration] [encrypted bytes]
\ 8 bytes of salt + password bytes \
```

The salt is prepended to the password to be encrypted and the whole value is ciphered. The resulting bytes are prepended with the original salt and iteration count in clear text and base64 encoding is applied. The tag {PBEWithMD5AndDES} is prepended to the resulting string to indicate the cipher and key algorithm.

The resulting value:

```
{PBEWithMD5AndDES}Iz3zv1XFirQ2ohr2fkrdrQbase64encodedAAAA=
```

To configure the Database Credential Mapping provider:

1. Open the Security Configuration folder.
2. Open the Service Control Manager folder that contains the Security Service Module whose provider you want to configure or change.
3. Open the Security Service Module folder that contains the provider you want to configure or change.

Note: If you have not bound the Security Service Module, open the Unbound Configuration folder that contains the provider you want to configure.

4. Open the Credential Mapping folder, and click Credential Mapping.

The Credential Mapping page appears.

5. Do one of the following:
 - To configure a new provider, click Configure a new Database Credential Mapper.
 - To change an existing provider, select the Database Credential Mapper to configure.
6. Do one of the following:
 - If you are adding a new provider, on the General tab, assign a name for the provider, and then click Create.
 - If you are changing an existing provider, edit the settings on the General tab as needed, and then click Apply.
7. Click the Details tab and define any additional attributes.

These attributes specify the types of credentials this provider is allowed to retrieve. If the `ALLOWED TYPE` attribute on the Details tab is set to a value of asterisk (*), then the provider attempts to serve all types of credentials using the query to limit the credentials returned.

8. Optionally, to configure failover, click the Failover tab and define additional attributes. See [“Configuring Failover for the Database Credential Mapper Provider” on page 9-36](#) for details on how to configure these attributes.
9. To define a query, click the Administration tab and perform the following tasks:
 - a. Configure the database to which to connect.

Note: You must have database administrator privileges to configure the database attributes.

- b. Configure your queries.

These queries are executed within the administration application only. They are used to count, list, retrieve, modify and add credential mappings. The queries operate in conjunction with the configured database connection and do not use the connection pool configured for the provider. You can select the credentials to retrieve, by either username (Select By Ident) or group (Select By Ident Group).

Count Record Query

List Records Query

Retrieve Record Query

Delete Record Query

Save Record Query

Save Record With Password Query

Add Record Query

- c. After you have located the user you want to map, you can map the credentials. To map credentials, click Edit Mappings. To add a new Mapping, click Add a new mapping, and define the attributes as needed.
 - d. Click Save and then click Close Editor.
10. Click Apply to save your changes.

Note: Changes made to a provider do not take effect until after it is explicitly deployed and the associated Security Service Module is restarted.

Configuring Failover for the Database Credential Mapper Provider

If you have an backup database store, you can configure the Failover tab to enable this feature. To configure failover:

1. Click the Failover tab.

2. Check `EnableAutomaticFailover` to enable the failover capability.

When enabled, the provider connects to the backup database when the primary database is unavailable.

3. Set the `FailoverThreshold` value.

The Failure Threshold defines the number of database errors that must occur sequentially on a connection pool before that pool is considered failed.

4. Set the `PrimaryRetryInterval` value.

The provider retries the connection to the primary database when the period of time as set in `PrimaryRetryInterval` is reached.

5. Specify the attributes `BackupDatabaseUserName` and `BackupDatabasePassword`. These are the username and password of the database administrator of the backup database. In many case, the `BackupDatabaseUserName` is the same as user name in the primary database. If it is different, you or your DBA need to make sure that this user have read access to the tables specified in the three SQL queries specified in the Details tab.

6. Specify `BackupDatabaseProperties` with necessary properties. These properties are used to get the JDBC connection to the backup database. These properties are entered as `NAME=VALUE`.

7. Enter the `BackupDatabaseURL`. This is the JDBC URL to use to connect to the backup database. The backup database should be from the same database vendor, as the backup connection uses the same database JDBC driver.

8. Enter the `BackupConnectionPoolMin`. This represents the minimum number of connections to allow in the backup connection pool.

9. Enter the `BackupConnectionPoolMax`. This represents the maximum number of connections to allow in the backup connection pool.

10. Click Apply to save your changes.

Note: Changes made to a provider do not take effect until after it is explicitly deployed and the associated Security Service Module is restarted.

Configuring a SAML Credential Mapping Provider

To configure the SAML Credential Mapping provider:

1. Open the Security Configuration folder.
2. Open the Service Control Manager folder that contains the Security Service Module whose providers you want to configure or change.
3. Open the Security Service Module folder that contains the providers you want to configure or change.

Note: If you have not bound the Security Service Module, open the Unbound Configuration folder that contains the providers you want to configure.

4. Open the Credential Mapping folder, and click Credential Mapping.

The Credential Mapping page appears.

5. Do one of the following:
 - To configure a new provider, click Configure a new SAML Credential Mapper.
 - To change an existing provider, select the SAML Credential Mapper to configure.
6. Do one of the following:
 - If you are adding a new provider, on the General tab, assign a name for the provider, and then click Create.
 - If you are changing an existing provider, edit the settings on the General tab as needed, and then click Apply.
7. Click the Details tab and define any additional attributes.
8. Click Apply to save your changes.

Note: Changes made to a provider do not take effect until after it is explicitly deployed and the associated Security Service Module is restarted.

Configuring an ASI Authorization Provider

The ASI Authorization provider works in conjunction with the ASI Role Mapping provider. You must configure both providers using the same values for each attribute. You can include any number of custom ASI Authorization and ASI Role Mapping providers within one Security Service Module.

To configure an ASI Authorization provider:

1. Open the Security Configuration folder.
2. Open the Service Control Manager folder that contains the Security Service Module whose providers you want to configure or change.
3. Open the Security Service Module folder that contains the providers you want to configure or change.

Note: If you have not bound the Security Service Module, open the Unbound Configuration folder that contains the providers you want to configure.

4. Open the Authorization folder, and click Authorization.

The Authorization page appears.

5. Do one of the following:
 - To configure a new provider, click Configure a new ASI Authorization Provider.
 - To change an existing provider, select the ASI Authorization provider to configure.
6. Do one of the following:
 - If you are adding a new provider, on the General tab, assign a name for the provider, and then click Create.
 - If you are changing an existing provider, edit the settings on the General tab as needed, and then click Apply.
7. Click the Details tab and define any additional attributes.
8. If you have configured a metadirectory, click the Metadirectory tab and define the metadirectory store. The metadirectory is required if you are writing policies that use delegation or user attributes. In addition to completing the fields required by the Metadirectory tab, you must configure the password used by the metadirectory to log into the policy database. For instructions on how to configure this password, see [“Using the asipasswd Utility to Configure the Metadirectory Password” on page 9-39.](#)

9. For improved performance, click the Performance tab and define any additional attributes.
10. If you are using a resource converter, attribute retriever, or attribute converter plug-in, click the Advanced tab and define the additional attributes. For a description of the attributes defined using the Advanced tab, see the in-line help text for the Advanced tab in the Administration Console. If you are using an ASI Role Mapping provider with this ASI Authorization provider, the values you enter on this tab must be the same as those you enter on the corresponding ASI Role Mapping provider Advanced tab. For instructions on how to implement Java-based plug-ins, refer to [“Using Java-Based Plug-ins” on page 12-2](#).

11. Click the Binding tab, select the resources to protect, and then click Bind.

Use this tab to bind resources to this provider. An ASI Authorization provider and an ASI Role Mapping provider for a Security Service Module only contain a subset of the overall policy. Binding applications to the provider determines that subset of policy. Applications not bound to the provider cannot be queried by the Security Service Module. A resource can be bound to only one Security Service Module, one ASI Authorization provider, and one ASI Role Mapper provider. The resources you bind must be the same for both providers.

Note: To unbind an ASI Authorization Provider from a Security Service Module, select the module to unbind, click the Binding tab, and then click the Unbind icon.

12. Click Apply to save your changes.

Note: Changes made to a provider do not take effect until after it is explicitly deployed and the associated Security Service Module is restarted.

Using the asipasswd Utility to Configure the Metadirectory Password

If you configure an ASI Authorization provider in a Security Service Module to use a metadirectory for user information, you must configure the login password used by the metadirectory to log into the policy database. To configure the password you must configure two files:

- `password.xml`—This file contains the encrypted database login password.
- `password.key`—This file contains the key used to decrypt the database login password.

You use the `asipasswd` utility to configure these files.

To run the `asipasswd` utility, perform the following steps:

1. Open a command window, go to `BEA_HOME\wles42-wls-ssm\instance\instancename\adm`, and enter the following command:

```
asipassword.bat username ..\ssl\password.xml ..\ssl\password.key
```

where:

instancename is the name of the instance of the Security Service Module.

username is the policy database login username.

2. When prompted, enter and confirm the policy database login password.

Configuring an ASI Adjudication Provider

When multiple authorization providers are configured, each one may return a different answer to the “is access allowed” question for a given resource. This answer may be `PERMIT`, `DENY`, or `ABSTAIN`. Determining what to do if multiple authorization providers do not agree on the answer is the primary function of the ASI Adjudication provider. Adjudication providers resolve authorization conflicts by weighing each authorization provider’s answer and returning a final decision.

The Adjudication provider behaves as follows:

- If all Authorization Providers return `PERMIT`, then `PERMIT`.
- If any Authorization Providers return `DENY`, then `DENY`.
- If `PERMIT` Authorization Providers return `ABSTAIN` and others return `PERMIT`, then `PERMIT` if unanimous permit is not required; otherwise `DENY`.

To configure an Adjudication provider:

1. Open the Security Configuration folder.
2. Open the Service Control Manager folder that contains the Security Service Module whose providers you want to configure or change.
3. Open the Security Service Module folder that contains the Security Service Module whose providers you want to configure or change.

Note: If you have not bound the Security Service Module, open the Unbound Configuration folder that contains the providers you want to configure.

4. Open the Adjudication folder, and click Adjudication.

The Adjudication page appears.

5. Do one of the following:
 - To configure a new provider, click Configure a new ASI Adjudicator.
 - To change an existing provider, select the ASI Adjudication provider to configure.
6. Do one of the following:
 - If you are adding a new provider, on the General tab, assign a name for the provider, and then click Create.
 - If there is an Adjudication provider configured, click Replace with a new ASI Adjudicator. The Edit ASI Adjudicator page appears.
7. If desired, change the name of the provider.
8. Optionally, set the Require Unanimous Permit attribute.

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

If the attribute is enabled, all Authorization providers must vote `PERMIT` in order for the Adjudication provider to vote true. By default, the Require Unanimous Permit attribute is enabled.

If the attribute is disabled, `ABSTAIN` votes are counted as `PERMIT` votes. To disable the Require Unanimous Permit attribute, click the check box.

Note: There are no attributes on the Details tab.

9. Click Apply to save your changes.

Note: Changes made to a provider do not take effect until after it is explicitly deployed and the associated Security Service Module is restarted.

Configuring an ASI Role Mapping Provider

The ASI Role Mapping provider is used in conjunction with the ASI Authorization provider. You must configure both providers using the same values for each attribute.

You can include any number of custom ASI Authorization and ASI Role Mapping providers within one Security Service Module. See [“Configuring an ASI Authorization Provider” on page 9-38](#) for further details.

To configure an ASI Role Mapping provider:

1. Open the Security Configuration folder.
2. Open the Service Control Manager folder that contains the Security Service Module whose providers you want to configure or change.
3. Open the Security Service Module folder that contains the provider you want to configure or change.

Note: If you have not bound the Security Service Module, open the Unbound Configuration folder that contains the provider you want to configure.

4. Open the Role Mapping folder, and click Role Mapping.
The Role Mapping page appears.
5. Do one of the following:
 - To configure a new provider, click Configure a new ASI Role Mapper Provider.
 - To change an existing provider, select the ASI Role Mapping provider to configure.
6. Do one of the following:
 - If you are adding a new provider, on the General tab, assign a name for the provider, and then click Create.
 - If you are changing an existing provider, edit the settings on the General tab as needed, and then click Apply.
7. Click the Details tab and define additional attributes. The values you enter here must be the same as those you entered for the corresponding ASI Authorization provider.
8. For improved performance, click the Performance tab and define additional attributes.
9. If you are using a resource converter, attribute retriever, or attribute converter plug-in, click the Advanced tab and define the additional attributes. For a description of the attributes defined using the Advanced tab, see the online help for the Advanced tab in the Administration Console Help. The values you enter on this tab must be the same as those you enter on the corresponding ASI Authorization provider Advanced tab. For instructions on how to implement Java-based plug-ins, refer to [“Using Java-Based Plug-ins” on page 12-2](#).
10. Click the Bindings tab and select the resources to protect, and then click Bind.

Use this tab to bind resources to this provider. An ASI Authorization provider and an ASI Role Mapper provider for a Security Service Module only contain a subset of the overall policy. Binding applications to the provider determines that subset of policy. Applications

not bound to the provider cannot be queried by the Security Service Module. A resource can be bound to only one Security Service Module, one ASI Authorization provider, and one ASI Role Mapper provider. The resources you bind must be the same for both providers.

11. Click Apply to save your changes.

Note: Changes made to a provider do not take effect until after it is explicitly deployed and the associated Security Service Module is restarted.

Configuring a Resource Deployment Audit Provider

Warning: Using a Resource Deployment Audit provider affects the performance of Administration Console even if only a few events are logged. It also affects the performance of the Security Service Module.

To configure the Resource Deployment Audit Provider:

1. Open the Security Configuration folder.
2. Open the Service Control Manager folder that contains the Security Service Module whose providers you want to configure or change.
3. Open the Security Service Module folder that contains the providers you want to configure or change.

Note: If you have not bound the Security Service Module, open the Unbound Configuration folder that contains the providers you want to configure.

4. Open the Auditing folder, and click Auditing.

The Auditing page appears.

5. Do one of the following:
 - To configure a new provider, click Configure a new Resource Deployment Auditor.
 - To change an existing provider, select the Resource Deployment Auditor to configure.
6. Do one of the following:
 - If you are adding a new auditor, on the General tab, assign a name for the provider, and then click Create.
 - If you are changing an existing auditor, edit the settings on the General tab as needed, and then click Apply.

7. If you are using a resource converter or attribute converter plug-in, click the Details tab and define the additional attributes. For a description of the attributes defined using the Details tab, see the online help for the Advanced tab in the Administration Console Help. For instructions on how to implement a Java-based plug-in, refer to [“Using Java-Based Plug-ins” on page 12-2](#).

8. Click Apply to save your changes.

Note: Changes made to a provider do not take effect until after it is explicitly deployed and the associated Security Service Module is restarted.

Configuring a Log4j Audit Channel Provider

The Log4j Audit Channel provider allows you to set the severity level at which auditing is initiated for the current Security Service Module. The output can be directed to an Output Stream, a `java.io.Writer`, a remote log4j server, a remote Unix Syslog daemon, or even a NT Event logger among many other output targets. Refer to the Apache Log4j documentation for detailed information on configuring Log4j.

Warning: Using an auditing provider affects the performance of Administration Console even if only a few events are logged. It also affects the performance of the Security Service Module.

To configure the Log4j Audit Channel provider:

1. Open the Security Configuration folder.
2. Open the Service Control Manager folder that contains the Security Service Module whose providers you want to configure or change.
3. Open the Security Service Module folder that contains the providers you want to configure or change.

Note: If you have not bound the Security Service Module, open the Unbound Configuration folder that contains the providers you want to configure.

4. Open the Auditing folder, and click Auditing.

The Auditing page appears.

5. Do one of the following:
 - To configure a new provider, click Configure a new Log4j Auditor.
 - To change an existing provider, select the Log4j Audit provider to configure.

6. Do one of the following:
 - If you are adding a new auditor, on the General tab, assign a name for the provider, and then click Create.
 - If you are changing an existing auditor, edit the settings on the General tab as needed, and then click Apply.

7. Click the Details tab and define any additional attributes.

Use this tab to configure the severity level at which auditing is initiated by the Log4j Audit Channel provider and to define what events are audited for the current Security Service Module. For a description of the attributes defined using the Details tab, see the online help for the Details tab in the Administration Console Help. For instructions on how to implement custom audit plug-ins, refer to [“Custom Audit Plug-ins” on page 12-9](#). For additional information on audit events see [“What is an AuditEvent?” on page 13-1](#) and [“What Events are Audited?” on page 13-3](#).

8. Click the Advanced tab and define any additional custom Log4j configuration attributes. For a description of the attributes defined using the Advanced tab, see the online help for the Advanced tab in the Administration Console Help. For instructions on how to implement custom audit plug-ins, refer to [“Custom Audit Plug-ins” on page 12-9](#).

9. Click Apply to save your changes.

Note: Changes made to a provider do not take effect until after it is explicitly deployed and the associated Security Service Module is restarted.

Configuring a Custom Security Provider

If the default security configuration does not meet your requirements, you can change the configuration properties (as permitted) for each provider. Or, you can create a new provider as described in [Developing Security Providers for BEA WebLogic Enterprise Security](#).

To configure a custom security provider:

1. Write a custom security provider.
2. Put the JAR file and any libraries used by the provider into the `WLES_HOME/lib/providers` directory on both the Administration Server and on the machine on which the Security Service Module is installed.
3. Start the Administration Console.
4. Open the Security Configuration folder, and click Security Control Managers.

The Service Control Manager page displays a list of folders representing each Service Control Manager configuration within your Administration Application.

5. Open the Security Service Module folder that contains the providers you want to configure or change, and click the folder name.

The Edit a Security Service Module Configuration page appears.

6. Open the folder for the provider you want to configure.

For example, expand the Authentication node to configure a custom Authentication provider.

7. Click the link representing the provider you want to configure.
8. Define values for the attributes on the each tab you have created.
9. Click Apply to save your changes.

Note: Changes made to a provider do not take effect until after it is explicitly deployed and the associated Security Service Module is restarted.

Deleting a Security Provider

To delete a security provider:

1. Open the Security Configuration folder.
2. Open the Service Control Manager folder that contains the Security Service Module whose provider you want to delete.
3. Open the Security Service Module folder that contains the providers you want to delete and click the provider type name.
4. Open the folder that contains the security provider you want to delete, and click the provider type name, for example: Authentication.

The configured providers are displayed in a table.

5. Select the provider to delete, and then click the Trash Can icon.

A confirmation message appears.

6. Click Yes.
7. To confirm that the provider is deleted, expand the provider node that previously contained the deleted provider.

Configuring a WebLogic Server Security Service Module

Before you can use the WebLogic Server v8.1 Security Service Module, you must configure the Security Service Module and the associated WebLogic Security Providers through the Administration Console. To configure a WebLogic Server v8.1 Security Service Module, perform the following tasks:

- [“Configuring a Service Control Manager” on page 9-2](#)
- [“Configuring a Security Service Module” on page 9-4](#)
- [“Configuring the WebLogic Security Providers” on page 9-47](#)
- [“Binding a Security Service Module to a Service Control Manager” on page 9-8](#)
- [“Distributing Configuration” on page 11-4](#)

Note: Changes made to a provider do not take effect until after it is explicitly deployed and the associated Security Service Module is restarted.

Configuring the WebLogic Security Providers

This procedure shows how to create the initial configuration for use with the WebLogic Server v8.1 Security Service Module. Configuring these providers ensures the WebLogic Server v8.1 Security Service Module starts properly. After you configure the WebLogic security providers, if desired, you can replace them with the BEA WebLogic Enterprise Security providers described in [Table 9-1](#).

To create an initial WebLogic Server v8.1 Security Service Module configuration:

1. Open the Security Configuration folder.
2. Open the Unbound Configurations folder, and click on Unbound Configurations.
The Unbound Security Service Module Configurations pages appears.
3. Click Create a new Security Service Module Configuration.
The Edit Security Service Module Configuration page appears.
4. On the General tab, in the Configuration ID text box, enter the identifier that was entered when this module was installed.
5. In the Description text box, enter a description for the service module configuration.
6. Click Create.

The Edit Security Service Module Configuration page displays and the new Security Service Module configuration appears under the Unbound Configurations folder in the navigation pane.

7. Click the Providers tab.

The Providers page displays six tabs, one for each type of provider available for configuration. [Table 9-4](#) lists the name of each WebLogic provider you can configure along with a description of each one.

Table 9-4 WebLogic Security Providers

Provider Name	Description
WebLogic Authentication Provider	Supports Embedded LDAP authentication. For information on configuring a WebLogic Authentication Provider, see “Configuring the WebLogic Authentication Provider” on page 9-49 .
WebLogic Authorization Provider	Provides enforcement of authorization for the security policy with which it is used. The Authorization Provider returns an access decision that determines which resources are protected and if a particular user is allowed access to a resource. For information on configuring a WebLogic Authorization Provider, see “Configuring the WebLogic Authorization Provider” on page 9-50 .
WebLogic Role Mapping Provider	Provides assignment to roles based on the security policy with which it is used. The WebLogic Role Mapping Provider returns a set or roles granted to a user on a protected resource. For information on configuring a WebLogic Role Mapping Provider, see “Configuring a WebLogic Role Mapping Provider” on page 9-50 .
WebLogic Credential Mapping Provider	Provides authentication credentials for a username and password. For information on configuring a WebLogic Credential Mapping Provider, see “Configuring the WebLogic Credential Mapping Provider” on page 9-51 .

8. To configure each security provider listed in [Table 9-4](#), refer to the following topics:
- [“Configuring the WebLogic Authentication Provider” on page 9-49](#)
 - [“Configuring the WebLogic Authorization Provider” on page 9-50](#)
 - [“Configuring a WebLogic Role Mapping Provider” on page 9-50](#)
 - [“Configuring the WebLogic Credential Mapping Provider” on page 9-51](#)

Configuring the WebLogic Authentication Provider

The WebLogic Authentication provider is case insensitive and you must ensure that all user names are unique. The WebLogic Authentication provider allows you to edit, list, and manage users and group membership. User and group membership information for the WebLogic Authentication provider is stored in an embedded LDAP server.

To configure the WebLogic Authentication provider:

1. Open the Security Configuration folder.
2. Open the Service Control Manager folder that contains the Security Service Module whose provider you want to configure or change.
3. Open the Security Service Module folder that contains the provider you want to configure or change.

Note: If you have not bound the Security Service Module, open the Unbound Configuration folder that contains the providers you want to configure.

4. Open the Authentication folder, and click on Authentication.

The Authentication page appears.

5. Do one of the following:
 - If you are adding a new provider, click Configure a New WebLogic Authenticator, assign a name for the provider, and then click Create.
 - To change an existing provider, select the WebLogic Authentication provider to configure.
6. If you are using multiple authentication providers, define a value for the Control Flag attribute on the General tab. For more information, [“Setting the JAAS Control Flag” on page 9-12](#).
7. Optionally, set the Minimum Password Length to specify the minimum number of characters required in a password processed by this WebLogic Authentication provider.

This password is the password used to define users in the embedded LDAP server used by the WebLogic Authentication provider to store user and group information.

8. Click **Apply** to save your changes.

Note: Changes made to a provider do not take effect until after it is explicitly deployed and the associated Security Service Module is restarted.

Configuring the WebLogic Authorization Provider

To configure the WebLogic Authorization provider:

1. Open the Security Configuration folder.
2. Open the Service Control Manager folder that contains the Security Service Module whose provider you want to configure or change.
3. Open the Security Service Module folder that contains the provider you want to configure or change.

Note: If you have not bound the Security Service Module, open the Unbound Configuration folder that contains the providers you want to configure.

4. Open the Authorization folder, and click Authorization.

The Authorization page appears.

5. Do one of the following:
 - If you are adding a new provider, click Configure a New WebLogic Authorizer, assign a name for the provider, and then click Create.
 - To change an existing provider, select the WebLogic Authorization provider to configure.

By default, the provider stores security policies that are created while deploying a Web application or an Enterprise JavaBean (EJB). To disable this feature, clear the Policy Deployment Enabled check box.

6. Click Apply to save your changes.

Note: Changes made to a provider do not take effect until after it is explicitly deployed and the associated Security Service Module is restarted.

Configuring a WebLogic Role Mapping Provider

The WebLogic Role Mapping provider can be used in conjunction with the WebLogic Authorization provider or an ASI Authorization provider.

Note: If you are using WebLogic Enterprise Security with the WebLogic Portal product, you must use a WebLogic Role Mapping provider with a WebLogic Authorization provider for authorization.

To configure WebLogic Role Mapping provider:

1. Open the Security Configuration folder.
2. Open the Service Control Manager folder that contains the Security Service Module whose providers you want to configure or change.
3. Open the Security Service Module folder that contains the provider you want to configure or change.
Note: If you have not bound the Security Service Module, open the Unbound Configuration folder that contains the provider you want to configure.
4. Open the Role Mapping folder, and click Role Mapping.
5. Do one of the following:
 To configure a new provider, click Configure a new WebLogic Role Mapper, assign a name for the provider, and then click Create.
 To change an existing provider, select the WebLogic Role Mapping provider to configure.
6. Click Apply to save your changes.
Note: Changes made to a provider do not take effect until after it is explicitly deployed and the associated Security Service Module is restarted.

Configuring the WebLogic Credential Mapping Provider

To configure the WebLogic Credential Mapping provider:

1. Open the Security Configuration folder.
2. Open the Service Control Manager folder that contains the Security Service Module whose provider you want to configure or change.
3. Open the Security Service Module folder that contains the provider you want to configure or change.
Note: If you have not bound the Security Service Module, open the Unbound Configuration folder that contains the providers you want to configure.
4. Open the Credential Mapping folder.
5. Do one of the following:
 - To configure a new provider, click Configure a New WebLogic Credential Mapper, assign a name for the provider, and then click Create.
 - To change an existing provider, select the WebLogic Credential Mapper to configure.

The Credential Mapping Deployment Enabled attribute specifies whether or not this Credential Mapping provider imports credential maps from a `weblogic-ra.xml` deployment descriptor file. To support the Credential Mapping Deployment Enabled attribute, the provider must implement the `DeployableCredentialProvider` SSPI. By default, the WebLogic Credential Mapping provider has this attribute enabled. The credential mapping information is stored in the embedded LDAP server.

6. Click Apply to save your changes.

Note: Changes made to a provider do not take effect until after it is explicitly deployed and the associated Security Service Module is restarted.

Performance and Caching

Authorization Caching

Authorization caching allows the ASI Authorization and ASI Role Mapper providers to cache the result of an authorization call, and use that result if future identical calls are made. The cache match is based on a combination of the following:

- Subject
- Resource
- Privilege
- Roles
- Applicable Context (the portion used in making the original decision)

Additionally cache will automatically invalidate itself if there is a policy or user profile change.

Configuring Authorization Caching

Authorization caching is on by default. It may be configured from within the Administration Console through the ASI Authorization and ASI Role Mapper provider configuration. [Table 10-1](#) lists the switches affect the authorization cache.

Table 10-1 Authorization Caching

Setting	Default Value	Description
AccessAllowedCaching	True	Enables/disables caching of authorization decisions.
GetRolesCaching	True	Enables/disables caching of role mapping decisions.
SessionExpiration	60	Defines the number of seconds each user session is valid for. Cached authorization decisions is reset each time the session expires. You can increase this value to improve performance.
SubjectDataCacheExpiration	60	Defines how long user profile data will be cached. Cached authorization decisions are reset each time this data cache expires. You can increase this value to improve performance.

The properties listed in [Table 10-2](#) can be entered as advanced configuration properties to further tune the cache.

Table 10-2 Advanced Configuration Properties

Setting	Default Value	Description
ASI.AuthorizationCacheLimit	1000	Determines the maximum number of cached decisions per user session. Once exceeded, old cached values are overwritten.

Table 10-2 Advanced Configuration Properties

ASI.AuthorizationCacheDynamicAttributeLimit	10	Determines the maximum number of context attributes a decision may use and still be stored in the cache.
ASI.PolicyCacheInvalidatorPollingInterval	1000	Determines how often the cache checks for policy distributions. The value is in milliseconds

Authorization Caching Expiration Functions

There is a small subset of data that may change without the knowledge of the cache. This includes internally computed time values, as well as custom evaluation plug-ins.

As the cache is not aware of changes in these values, it does not automatically invalidate a cached decision when they change. For this reason a series of evaluation functions is provided to control the period of cache validity. These functions are only needed in rules that make explicit use of internally computed time values or custom evaluation plug-ins.

[Table 10-3](#) lists the internally computed time values. If referenced in a rule, you should also explicitly set the cache validity for the rule.

Table 10-3 Time Values used with Expiration Functions

Credential	Value	Range or Format
time24	integer	0–2359
time24gmt	integer	0–2359
dayofweek	Dayofweek_type	Sunday–Saturday
dayofweekgmt	Dayofweek_type	Sunday–Saturday
dayofmonth	integer	1–31

Table 10-3 Time Values used with Expiration Functions

dayofmonthgmt	integer	1–31
dayofyear	integer	1–366
dayofyeargmt	integer	1–366
daysinmonth	integer	28–31
daysinyear	integer	365–366
minute	integer	0–59
minutegmt	integer	0–59
month	month_type	January–December
monthgmt	month_type	January–December
year	integer	0–9999
yeargmt	integer	0–9999
timeofday	time	“HH:MMAM” or “HH:MMPM”
timeofdaygmt	time	“HH:MMAM” or “HH:MMPM”
hour	integer	0–23
hourgmt	integer	0–23
date	Date	“MM/DD/YYYY”
dategmt	Date	“MM/DD/YYYY”

The following is a list of expiration functions for the authorization cache. They let you set an expiration time for the decision. This way you can instruct the cache to only hold the value for a given period of time, or to not hold it at all. They roughly correspond to each of the internally computed time types.

Table 10-4 Expiration Functions

Function	Argument Type	Description
valid_for_mseconds	integer	Valid for a given number of milliseconds
valid_for_seconds	integer	Valid for a given number of seconds
valid_for_minutes	integer	Valid for a given number of minutes
valid_for_hours	integer	Valid for a given number of hours
valid_until_timeofday	time	Valid until the specified time on the date the evaluation is performed
valid_until_time24	integer	Valid until the specified time on the date the evaluation is performed
valid_until_hour	integer	Valid until the specified hour on the date the evaluation is performed
valid_until_minute	integer	Valid until the specified minute of the hour the evaluation is performed
valid_until_date	Date	Valid until the specified date
valid_until_year	integer	Valid until the specified year
valid_until_month	month_type	Valid until the specified month of the year the evaluation is performed
valid_until_dayofyear	integer	Valid until the specified day of the year the evaluation is performed

Table 10-4 Expiration Functions

valid_until_dayofmonth	integer	Valid until the specified day of the month the evaluation is performed
valid_until_dayofweek	Dayofweek_type	Valid until the specified day of the week the evaluation is performed
valid_until_timeofday_gmt	time	Valid until the specified time on the date the evaluation is performed in GMT time.
valid_until_time24_gmt	integer	Valid until the specified time on the date the evaluation is performed in GMT time.
valid_until_hour_gmt	integer	Valid until the specified minute of the hour the evaluation is performed in GMT time
valid_until_minute_gmt	integer	Valid until the specified minute of the hour the evaluation is performed in GMT time.
valid_until_date_gmt	Date	Valid until the specified date in GMT time.
valid_until_year_gmt	integer	Valid until the specified year in GMT time.
valid_until_month_gmt	month_type	Valid until the specified month of the year the evaluation is performed in GMT time.
valid_until_dayofyear_gmt	integer	Valid until the specified day of the year the evaluation is performed in GMT time.

Table 10-4 Expiration Functions

<code>valid_until_dayofmonth_gmt</code>	integer	Valid until the specified day of the month the evaluation is performed in GMT time.
<code>valid_until_dayofweek_gmt</code>	Dayofweek_type	Valid until the specified day of the week the evaluation is performed in GMT time.

For example, if you had the following rule:

```
GRANT(/priv/order,//app/resturant/breakfast,//sgrp/customers/allusers/
) if hour < 11;
```

When authorization caching is enabled, you write the rule as:

```
GRANT(/priv/order,//app/resturant/breakfast,//sgrp/customers/allusers/
) if hour < 11 and valid_until_hour(11);
```

With authorization caching the result of this rule is cached in the provider until 11:00 AM, at which time, it expires. Not calling `valid_until_hour` results in this rule being cached until the next policy distribution. So, you can see that if you are using authorization caching, it is important to update your time dependent rules appropriately.

Deployment

This topic describes how to distribute configuration and policy data.

- [“Understanding Deployment” on page 11-1](#)
- [“Managing Deployment” on page 11-2](#)
- [“Distributing Policy” on page 11-2](#)
- [“Distributing Configuration” on page 11-4](#)
- [“Viewing Distribution Results” on page 11-5](#)
- [“Viewing Deployment Status” on page 11-6](#)

Deployment

Understanding Deployment

You must distribute both policy and configuration data before they can take effect. You can distribute policy data and configuration data together, or you can distribute only configuration data as structural changes. An application must be bound to an ASI Authorization and ASI Role Mapping provider before you can distribute a policy to the associated Security Service Module. Your policy data consists of the collection of rules that define your authorization policy.

When you distribute configuration data, you distribute the Security Service Module configurations to the Service Control Managers to which they are bound. Configuration data

defines how the security providers are configured to protect your application. Configuration data define the set of resources used by your provider configuration.

In the case of security configuration, you must select the Security Service Modules to receive the pending configuration. Results of all deployments for a session are displayed on the Distribution Results page. Structural changes (adding or removing bindings) are distributed automatically when either policy information or configuration information is distributed.

When you distribute policy or configuration changes, all structural changes are distributed together to maintain configuration and policy referential integrity within the system. If you make a change to a configuration, it is distributed the next time policy information is distributed. When you make a change to a policy binding, it is distributed the next time configuration information is distributed.

Distributing structural changes allows an administrator to provision changes to policy bindings and configuration bindings without distributing changes to policies or configurations themselves. In this way, a binding that is removed can be provisioned to the running Security Service Modules without having to distribute configuration or policy changes that may be waiting for distribution at another time.

Managing Deployment

[“Distributing Policy” on page 11-2](#)

[“Distributing Configuration” on page 11-4](#)

[“Distributing Structural Changes” on page 11-5](#)

[“Viewing Distribution Results” on page 11-5](#)

[“Viewing Deployment Status” on page 11-6](#)

Distributing Policy

After you design your policy, you must deploy it to the appropriate Security Service Modules before it can take effect. When you distribute policy data, you define the resources to be protected by that policy through the associated Security Service Module. An application must be bound to an Authorization and Role Mapping provider before you can distribute a policy to it; policy is not distributed to an application or resource.

To distribute policy data:

1. Open the Deployment folder.

The Deployment window displays three tabs: Policy, Configuration, and Structural Changes.

2. Click the Policy tab.

The Policy tab displays a tree-like-structure of resources available to receive the policy. You define these resources as distribution points when you create them. If there is a "+" symbol for an item, you can expand that item to view its child nodes. Each resource item in the list has a check box.

The following figure illustrates a policy distribution with several resource nodes: *WLES* (with five child nodes) and *WLESRecovery* (with no child nodes).

Figure 11-1 Policy Distribution Results



3. Check the nodes to which you want to distribute the policy.

The security policy is applied to the resource you select and is inherited by all resources below this distribution point (to all child nodes).

4. Click **Distribute Policy** to distribute the policy to the selected resources.

After the distribution starts, the Deployment Status page appears. If you have already done a distribution during this console session, the Deployment Status page displays the status results from the previous distributions.

5. Click **Deployment Status** to view the latest distribution status and ensure that the distribution was successful.
6. Click **Refresh** to update the results of the distribution.

Distributing Configuration

To distribute the Security Service Module configurations to the Service Control Managers to which they are bound:

1. Open the Deployment folder.

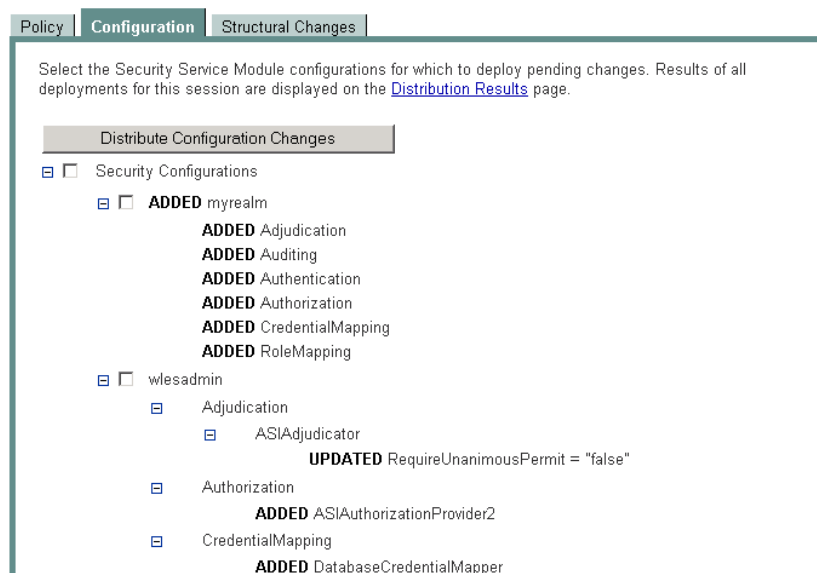
The Deployment window displays three tabs: Policy, Configuration, and Structural Changes.

2. Click the Configuration tab.

The Configuration tab displays a tree-like-structure of Configurations, Providers, and Attributes that are available for policy distribution. If there is a "+" symbol beside an item, you can expand that item to view its child nodes. Each configuration, provider, and attribute item in the list has a check box.

The following figure illustrates a typical configuration distribution, representing a Security Service Module called myrealm that has six newly configured providers (ADDED). In the second Security Service Module configuration called wlesadmin, the ASI Adjudicator has been modified (UPDATED), and an Authorization provider (ASIAuthorizationProvider2) and Credential Mapping provider (DatabaseCredentialMapper) were added (ADDED).

Figure 11-2 Configuration Status



3. Check the configurations to which to deploy pending changes.
4. Click **Distribute Configuration Changes** to distribute the policy to the selected nodes.
5. Click **Distribution Results** to view the latest distribution status.
6. Click **Refresh** to update the status of the distribution.

Distributing Structural Changes

Structural changes (removal of bindings) are distributed automatically when either policy information or configuration information is distributed. When policy or configuration changes are distributed, all structural changes are distributed also to maintain configuration and policy referential integrity within the system.

Making a structural change to a configuration (unbinding a configuration from a Service Control Manager), means the change is distributed the next time policy information is distributed. Making a structural change to a policy binding (unbinding an application from an Authorization provider), makes the change the next time configuration information is distributed.

To distribute only structural changes:

1. Open the Deployment folder.

The Deployment window displays three tabs: Policy, Configuration, and Structural Changes.

2. Select the Structural Changes tab.
3. Click **Distribute Structural Changes Only**.

Distributing structural changes only, allows an administrator to provision changes in policy bindings and configuration bindings without distributing changes to policies or configurations themselves. In this way, a binding that is removed can be provisioned to the running Security Service Modules without having to distribute configuration or policy changes that may be waiting for distribution at another time.

Viewing Distribution Results

After you deploy your policy or configuration, you should verify that it was properly distributed.

To view the results of your deployment:

1. Open the Deployment folder.

2. Open the **Distribution Results** folder.

The Distribution Results page displays the results of any deployments done during your current working session.

The following figure illustrates the status of a distribution, showing the distribution ID, the username of the person who performed the distribution, the percentage of the distribution that completed, and the date and time of the distribution. The status report also indicates the components that received the policy, along with the group name and machine name (Host). The Instance represents the Service Control Manager and Authorization and Role Mapping provider that received the distribution.

Figure 11-3 Distribution Results

Status of Last Distribution

Distribution ID	9		
Username	//user/wles/system/		
% Complete	100		
Notes	distribution done		
Time	01/09/04 20:07:12 UTC		
Instance	Group	Host	Success
SCM.xxxconfig.BYRNESPC	//bind/xxxconfig	/BYRNESPC	true
WLES_ame_wlesadmin_BYRNESPC	//bind/wlesadmin	/BYRNESPC	true
SCM.xxxconfig.kirkwood	//bind/xxxconfig	/wrko.bea.com	true
WLES_ame_wlesadmin_kirkwood	//bind/wlesadmin	/wrko.bea.com	true

3. Click **Refresh** to update the results of the distribution.

Viewing Deployment Status

The Deployment Status page displays the name of each Service Control Manager and ARME instance registered with the Administration Application, indicating whether or not the policy or configuration data are synchronized with the database. When the configuration data for a Service Control Manager or policy data for the Security Service Module instance are not synchronized with the database, you may need to redeploy the data or refresh the instance to ensure that it is updated properly.

You can also use this page to remove any Security Service Module or Service Control Manager instance that you have uninstalled and is no longer in use.

To unregister and remove a Service Control Manager or ARME instance from the Administration Application:



1. Uninstall the Service Control Manager and ARME from the machine on which they are installed.
1. Open the Deployment folder.
2. Click **Deployment Status**.

The **Deployment Status** page displays the name of each Service Control Manager and ARME you have registered and deployed with the Administration Application. The following figure illustrates the status of a Service Control Manager and ARME registered for use with the Administration Application, including their Configuration Name, Instance Name, Distribution ID, and Address. In this case, there is one Service Control Manager (SCM.asiconfig.wrko) and one ARME (WLES_armed_wlesadmin_wrko) instance.

Figure 11-4 Distribution Status

There are no unsynchronized instances.

Synchronized instances

Configuration Name	Instance Name	Distribution ID	Address	
asiconfig	SCM.asiconfig.wrko	13	https://wrko:8017/armedPortType	
wlesadmin	WLES_armed_wlesadmin_wrko	11	https://wrko.bea.com:8010	

3. From the Configuration Name list, click the trash can icon for the instance you want to remove.

To synchronize an instance:

1. Log onto the machine on which the SCM and ARME instances are installed.
2. To refresh the SCM instance, run:

```
WLES_SCM refresh
```

On Windows, you can run this command from the Program menu.

3. To refresh the ARME , run:

```
WLES_armed refresh
```

On Windows, you can run this command from the Program menu.

Provider Extensions

The following topics are covered in this section:

- [“What is a Provider Extension?” on page 12-1](#)
- [“Authorization and Role Mapping Extensions” on page 12-2](#)
- [“Custom Audit Plug-ins” on page 12-9](#)
- [“Database Authentication Plug-in” on page 12-10](#)

What is a Provider Extension?

A provider extension is a plug-in function that you write to extend the capabilities of the existing providers. You can use these plug-ins to manipulate existing policy data in a way that is not already provided or to retrieve data from external sources to add to an authorization or role mapping decision or a deployment audit. You can use these plug-ins with the ASI Authorization, ASI Role Mapping, Resource Deployment Audit, Log4j Audit Channel, and Database Authentication providers.

While the security providers supplied with WebLogic Enterprise Security are configurable, the plug-ins enable you to customize them to add additional functionality. For example, you may want some form of special business logic to retrieve additional data that you want to use before the authorization decision is made or for the custom processing of data, such as the audit context. Plug-ins are provided for a variety of functions:

- You can use Java-based plug-ins to perform attribute retrieval, attribute conversion, and resource conversion. You can use attribute retrievers to retrieve embedded data from

complex data objects or external data sources. You can use resource converters to convert WebLogic Server and WebLogic Enterprise Security data to an internal resource format. You can use attribute converters to convert context data to an internal attribute format.

- You can use C++ language extensions plug-ins to add your own custom authorization and role mapping evaluation functions to the standard ones provided. After you develop a function, administrators can manipulate its input using the Administration Console. The plug-in appears to the administrator as simply new evaluation functions or newly available dynamic attributes.
- You can use the audit plug-ins to help format audit events that are generated by the Security Framework, the runtime API, or custom implementations.
- You can use the database authentication plug-in with the Database Authentication provider to customize authentication features.

The following sections provide more information on the plug-ins and how to use them.

- [“Authorization and Role Mapping Extensions” on page 12-2](#)
- [“Custom Audit Plug-ins” on page 12-9](#)
- [“Database Authentication Plug-in” on page 12-10](#)

Authorization and Role Mapping Extensions

WebLogic Enterprise Security supports using Java-based plug-ins and language extensions with security providers. You can use these plug-ins to performed custom functions for authorization, role mapping, and resource deployment auditing.

The following types of plug-ins are supported:

- [“Using Java-Based Plug-ins” on page 12-2](#)
- [“Using Language Extensions” on page 12-7](#)

Using Java-Based Plug-ins

WebLogic Enterprise Security providers support three types of Java-based plug-ins: resource converters, attribute retrievers, and attribute converters. [Table 12-1](#) shows the types of Java-based plug-ins that each security provider supports.

Table 12-1 Java-based Plug-in Support for Security Providers

Security Provider	Supports Resource Converter Plug-in	Supports Attribute Retrievers Plug-in	Supports Attribute Converter Plug-in
ASI Authorization provider	Yes	Yes	Yes
Resource Deployment Audit provider	Yes	No	Yes
ASI Role Mapping provider	Yes	Yes	Yes

To use these plug-ins, you must perform the following tasks:

1. Write a Java class that implements the plug-in interface.
2. Place the Java class in the appropriate directory of the Security Service Module with which you intend to use the plug-in. A single Java class may be used with more than one Security Service Module.
3. Use the Administration Console to register the Java class on the desired Security Service Module(s).

For instructions for performing these tasks, refer to the following sections:

- [“Using the Java-based Plug-in Interfaces” on page 12-3](#)
- [“Resource Converter” on page 12-4](#)
- [“Attribute Retriever” on page 12-5](#)
- [“Attribute Converter” on page 12-6](#)

Using the Java-based Plug-in Interfaces

To implement a Java-based plug-in interface, you must perform the following steps:

1. Refer to the description of the plug-in interface you want to use and write a Java class to implement the interface. The following sections provide descriptions of each type of plug-in interface:
 - [“Resource Converter” on page 12-4](#)

- “Attribute Converter” on page 12-6
 - “Attribute Retriever” on page 12-5
2. Use the Java class to create a JAR file and place the JAR file in the `/lib/providers` directory in the installation directory for the Security Service Module on the machine on which the Security Service Module is installed. For example, the default location of this directory for the WebLogic Server 8.1 Security Service Module is
`C:\bea\wles42-wls-ssm\lib\providers`.
 3. Refer to the following sections and use the Administration Console to register the Java plug-ins in the desired security providers for the desired Security Service Modules:
 - “Configuring an ASI Authorization Provider” on page 9-38
 - “Configuring an ASI Role Mapping Provider” on page 9-41
 - “Configuring a Resource Deployment Audit Provider” on page 9-43

Resource Converter

Resource converters are used by ASI Authorization, ASI Role Mapping, and Resource Deployment Audit providers to convert WebLogic Server resources into an internal resource format that is recognized by WebLogic Enterprise Security. For a description of the policy data formats, see the *BEA WebLogic Enterprise Security Policy Managers Guide*.

`ResourceConverter` is an interface in the `com.bea.security.providers.authorization.asi` package. This interface is used to implement plug-ins for converting from the Security Framework defined resource interface into an access query. There is no standard for resource definitions so plug-ins are needed to handle each of the resource types. The set of resource types is not fixed and you can define your own resource, in which case, you need to define a resource converter to allow the ASI Authorization provider to protect the resource. Numerous resource converters are supplied for your use, one for each defined WebLogic Server and WebLogic Enterprise Resource type. [Table 12-2](#) lists and describes the methods provided by the `ResourceConverter` interface.

Table 12-2 ResourceConverter Interface Methods

Method	Description
<code>String[] getHandledTypes()</code>	This method is called when the plug-in is instantiated and is used to determine what resource types the converter knows how to handle. The Security Framework represents resource types internally as strings.
<code>AccessElement convertResource(Resource resource, ContextHandler contextHandler) throws ResourceConversion Exception</code>	<p>This method extracts enough information from a <code>Resource</code> and <code>ContextHandler</code> to form an access query. The minimum amount of required information to be extracted is the resource object and privilege. Additional information that can be included is the application name and input attributes extracted from the <code>Resource</code> or <code>ContextHandler</code>:</p> <p>If the application is not specified, then the provider uses the following rules for selecting one:</p> <ul style="list-style-type: none"> • If no application is specified, then the object is queried under the shared resource node as specified in the provider configuration. • If an unqualified application is specified, the object is queried under the default deployment node, plus the application, plus the object. • If a fully qualified application is specified, then the object is queried under that node. <p>If the resource converter is unable to generate an access query from the information provided in the <code>Resource</code>, it throws a <code>ResourceConversionException</code> indicating to the provider and framework that this query cannot be answered by this provider.</p>
<code>Object getAttributeValue(Resource resource, String name, ContextHandler contextHandler)</code>	This method finds the value of a missing attribute. It is left up to you as the developer of the <code>ResourceConverter</code> plug-in to determine how the <code>ResourceConverter</code> gets the required value. The plug-in may return null if the value is not found.

Attribute Retriever

Attribute retrievers are used by ASI Authorization and ASI Role Mapping providers to retrieve attributes for use by WebLogic Enterprise Security authorization and role mapping.

`AttributeRetriever` is an interface in the `com.bea.security.providers.authorization` package that you can use to implement plug-ins for retrieving attributes. You use an implementation of the `AttributeRetriever` interface to get embedded data from complex data

objects. For example, if the `ContextHandler` includes an address element, you can use an attribute retriever to make the zip code portion of the address available separately. You can also use an attribute retriever to fetch data from external data sources, for example, JDBC data stores.

Note: It is generally not necessary to write attribute retrievers for objects that appear directly in the `ContextHandler`; attribute retrievers are used to extract embedded or otherwise inaccessible data.

You can register multiple attribute retrievers with the same attribute name. If you do so, the attribute retrievers are called in order until one of them returns a non-null result.

[Table 12-3](#) lists and describes the methods provided by the `AttributeRetriever` interface.

Table 12-3 AttributeRetriever Interface Methods

Method	Description
<code>String[] getHandledAttributeNames()</code>	This method returns the names of attributes handled by this object. An empty or <code>null</code> value indicates that the retriever is considered capable of handling any attribute name.
<code>Object getAttributeValue(String name, Subject subject, Map roles, Resource resource, ContextHandler contextHandler)</code>	<p>This method retrieves the value of the named attribute. Additional authorization request data is made available to allow for more complex attribute retrieval. The parameters are as follows:</p> <ul style="list-style-type: none">• <code>name</code>—name of the needed attribute• <code>subject</code>—subject associated with the request• <code>roles</code>—role membership of the subject, or <code>null</code> if this is a role mapping call• <code>resource</code>—resource associated with the request• <code>contextHandler</code>—context associated with the request; may be <code>null</code> if non-existent <p>This method returns the attribute value, or <code>null</code> if the attribute is not found.</p>

Attribute Converter

Attribute converters are used by ASI Authorization, ASI Role Mapping, and Resource Deployment Audit providers to convert context data to an internal attribute format. For a description of the policy data formats, see the [BEA WebLogic Enterprise Security Policy Managers Guide](#).

To create attribute converters, you implement the `TypeConverter` interface. This interface converts between native Java types and ASI formatted Strings. If you create a new ASI type, you may want to create a Java class to handle it and implement a `TypeConverter` interface to handle that class. ASI types are the credential types that are visible through the console such as integer, date, and string types, and so on, versus Java data types. [Table 12-4](#) lists and describes methods provided by the `TypeConverter` interface.

Table 12-4 TypeConverter Interface Methods

Method	Description
<code>Class getType()</code>	This method returns the type which this converter converts.
<code>String getASITypeName()</code>	This method returns the ASI type name.
<code>String convertToASI(Object javaFormat) throws UnsupportedOperationException</code>	This method converts a java object into a ASI string.
<code>Object convertFromASI(String asiFormat) throws TypeConversionException</code>	This method converts a ASI string to a Java Object.

Using Language Extensions

The ASI Authorization and ASI Role Mapping providers support the use of C++ plug-ins for custom rule extensions for evaluation and credential functions. The functions available for use are described in [“Function Reference” on page 14-1](#).

This section contains a description of how to create an extension library for the ASI Authorization and ASI Role Mapper engine (ARME) used by the ASI Authorization provider. This example works with BEA WebLogic Enterprise Security, Version 4.2.

The product installation includes an example of code and build commands for an extension library located in the `/examples` subdirectory in the Administration Application installation directory.

For instructions for building, deploying, and using extensions, see the following sections:

- [“Building an Extension” on page 12-8](#)
- [“Deploying the Extension” on page 12-8](#)

- [“Using the Extension” on page 12-9](#)

Building an Extension

To build the extension library, you need to have the following header files, installed in the `asi` subdirectory:

- `armeapi.h`
- `AttributeValue.h`
- `session.h`
- `defs.h`
- `exception.h`

The extension needs to be linked with the following libraries, included with the ARME executable:

- `utilmd.lib` or `libutil.so`
- `pluginmd.lib` or `libplugin.so`

The compiler used must generate library binaries compatible with the compiler used to compile the ARME server.

- On Windows platform, Visual C++ 6.0 SP5
- On Linux Red Hat, AS 2.1: gcc 2.96
- On Solaris, Forte 7.0, or a binary compatible mode for later versions

Deploying the Extension

To deploy the extension, do the following:

1. Place the compiled library (for example, `arme_extension.dll`) into the same path accessible by the ARME process.
2. Configure the initialization function in the ARME local configuration file, using the following syntax:

```
<ARME.tag>.plugin[1..4] <path>/<DLL plug_in filename>(initialize 'arg')
```

For example:

```
ARME.wlesadmin.plugin1 arme_extension.dll(initialize 'test')
```

This example assumes that the `arme_extension.dll` is in the path for the ARME process. In this example, `initialize` is the name of the routine in the extension library that is called when the library is loaded to perform initialization. `<ARME.tag>` is a parameter passed in the command line of the ARME process. This parameter defines the scope for the configuration parameters used from a local file. You may use an empty scope for these keywords; that is, just `plugin[1..4]`.

Using the Extension

The extension library adds credential functions (custom dynamic attributes) and evaluation functions. To use them in your policy, you need to add declarations for them. For example, if an extension library defines the `custom-attribute` credential function, you need to add a declaration for a `custom-attribute` in the Administration Console with a dynamic type, and an appropriate data type (string, integer, and so on.). Then, you can use this attribute in rule constraints.

Custom Audit Plug-ins

The Log4j Audit Channel provider uses Log4j renderer classes that convert the associated audit event object into a simple string representation. However, you can write custom renderers that convert the audit event object to something other than the default string representation and register them as plug-ins using the Administration Console.

Refer to the following topics for information how to write and register custom audit plug-ins:

- [“Using the Custom Audit Plug-in” on page 12-9](#)
- [“Audit Plug-in Renderer Class” on page 12-10](#)

Using the Custom Audit Plug-in

To implement an audit plug-in interface, you must perform the following steps:

1. Refer to [“Audit Plug-in Renderer Class” on page 12-10](#) for a description of the audit plug-in renderer class and write a Java class to implement a new renderer class.
2. Use the Java class to create a JAR file and place the JAR file in the `/lib/providers` directory in the installation directory for the Security Service Module on the machine on which the Security Service Module is installed. For example, the default location of this directory for the WebLogic Server 8.1 Security Service Module is:
`C:\bea\wles42-wls-ssm\lib\providers.`

3. For instructions on using the Administration Console to register the audit plug-in for the desired Log4j Audit Channel provider, refer to “[Configuring a Log4j Audit Channel Provider](#)” on page 9-44.

Audit Plug-in Renderer Class

To write a plug-in renderer class, you must implement the `org.apache.log4j.or.ObjectRenderer` interface and then register the renderer class to the type of Audit Event class for which you want to use that renderer. For example,

```
weblogic.security.spi.MyAuditEvent=com.bea.security.providers.  
audit.MyAuditEventRenderer
```

For instructions on how to write a renderer for a custom object, see the Log4j documentation located at <http://logging.apache.org/log4j/docs/documentation.html>.

Table 12-5 lists and describes a sample `AuditEventRenderer` class.

Table 12-5 AuditEventRenderer Class Method

Method	Description
<pre>public class MyAuditAtnEventRenderer implements org.apache.log4j.or.ObjectRenderer { public String doRender(Object o) { String eventStr = null; if(o instanceof MyAuditEvent) { MyAuditEvent event = (MyAuditEvent) o; eventStr = event.getEventType()+" -- "+event.toString(); } return eventStr; } }</pre>	In this sample, this method renders the <code>AuditEvent</code> object as a simple string. To render the Audit Event as something other than a simple string, modify this method to form your own string representation.

Database Authentication Plug-in

The Database Authentication extension is used by the Database Authentication provider to customize authentication features. The default database authentication extension (located in the `com.bea.security.providers.authentication.dbms.DefaultDBMSPluginImpl` package) is designed to authenticate the user against the policy database. This implementation uses a specific password hashing algorithm, namely, SHA1 and SHA-1. It also uses a special

format for the user name and the group name that is pertinent to the policy database. The hashing algorithm used is:

```
{Algorithm} + 4 byte Salt+passwordhash
```

The policy database uses name scope (for example, directory name) and a qualified name format to store the user and group. See the [BEA WebLogic Enterprise Security Policy Managers Guide](#) for details.

If you are authenticating users against another database that uses a different password hashing algorithm and a different user/group name format, you may decide to implement your own plug-in by following the guidelines provided with the plug-in.

A custom database authentication plug-in must also implement the DBMSPlugin Interface (located in the `com.bea.security.providers.authentication.dbms.DBMSPlugin` package). The DBMSPlugin Interface implementation must include the methods described in [Table 12-6](#).

To use your plug-in implementation, you need to deploy the plug-in class (or its JAR file) in the classpath of the Database Authentication provider and use the Administration Console to configure the Database Authentication provider to use the plug-in.

[Table 12-6](#) lists and describes the methods provided by the DBMSPlugin interface.

Table 12-6 DBMSPlugin Interface Methods

Method	Description
<code>public void initialize()</code>	This method is executed when the authorization provider is initialized on startup.
<code>public void shutdown()</code>	This method is executed when the authorization provider is shut down.

Table 12-6 DBMSPlugin Interface Methods

Method	Description
<code>public boolean authenticate(String user, char[] password, char[] databasePassword, Map options)</code>	<p>When the Database Authentication provider attempts to authenticate a user, the <code>authenticate</code> method is called on the plug-in. This method may be called in one following two scenarios. If the provider is configured with the SQL Query to retrieve password, the password (<code>databasePassword</code>) is retrieved from the database using this query and is provided to this method. This <code>authenticate</code> method must determine if the user provides the correct password (<code>password</code>) and return <code>true</code>, if authenticated, or <code>false</code>.</p> <p>The options map contains a <code>TRUE</code> value for key = <code>"QueryPassword"</code>. If no SQL Query string is configured for retrieving the password, the Database Authentication provider assumes that the authentication plug-in retrieves the password and then authenticates the user. The options map contains values for these keys, <code>"scope"</code> and <code>"connection"</code>, and a <code>FALSE</code> value for key = <code>"QueryPassword"</code>. Also, <code>databasePassword</code> = <code>null</code>.</p>
<code>public String formatUser(String user, Map options)</code>	<p>This method is executed before any call to the database. The user string is the one passed into the login module. This method returns a formatted user name, which is later used as the input parameter in all the SQL queries to verify user, to retrieve password, and to retrieve groups. The options Map contains values for these keys, <code>"scope"</code> and <code>"connection"</code>, and the configured string of the SQL query to verify user with key = <code>"SQL_QUERY"</code>.</p>
<code>public Vector formatGroups(String user, Vector groups, Map options)</code>	<p>This method is executed after the call to retrieve groups from the database. A vector of strings containing the groups the user belongs to are passed in. Any formatting of group names that is required before inserting these into the Subject should be done and the resulting vector passed back. The options Map contains values for these keys, <code>scope</code> and <code>connection</code>, and the configured string of the SQL query to retrieve groups with key = <code>"SQL_QUERY"</code>.</p>

The `options` object is a map containing optional information that the plug-in may want to use. The most common options of use and their keys for retrieval are:

- `key = scope`—the configured scope for the Database Authentication provider.

- `key = QueryPassword`—the `java.lang.Boolean` value that indicates whether the password SQL Query String was configured and executed. If it is false, then the password was not retrieved from the database. This key is only present for the authentication method.
- `key = connection`—an open JDBC `java.sql.Connection` object. Do not close this object; it is returned to the pool after authentication.

Audit Events

The following topics are covered in this section:

- “What is an AuditEvent?” on page 13-1
- “What Events are Audited?” on page 13-3
- “Custom Audit Context Extensions” on page 13-4
- “Audit Event Interfaces and Audit Events” on page 13-5
- “Additional Audit Event Interfaces” on page 13-16
- “Using Custom Audit Providers” on page 13-21

What is an AuditEvent?

The `AuditEvent` interface provides a mechanism for passing additional audit information to Auditing providers during a `writeEvent` operation. This is the base interface that is extended by components in the Security Framework to compose specific audit event types. Extending this interface helps auditing providers determine the calling security component.

If you implement this interface and you expect to receive a `ContextHandler` argument from a caller, you can extend the `AuditContext` interface to provide more information. Some of the sub-interfaces defined by the security SPI are listed in [Table 13-1](#). [Table 13-1](#) also indicates which sub-interfaces implement the `AuditContext` interface. These interfaces are documented in the *BEA WebLogic Enterprise Security Provider SSPI 4.2 API Reference*.

Table 13-1 Audit Events

Audit Event Name	Interface Class	Interfaces Implemented	
		AuditEvent	AuditContext
Authentication Audit Event	weblogic.security.spi.AuditAtnEvent	Yes	No
Authentication Audit Event V2	weblogic.security.spi.AuditAtnEventV2	Yes	Yes
Authorization Audit Event	weblogic.security.spi.AuditAtzEvent	Yes	Yes
Role Mapping Audit Event	weblogic.security.spi.AuditRoleEvent	Yes	Yes
Credential Mapping Audit Event	weblogic.security.spi.AuditCredentialMappingEvent	Yes	Yes
Management Audit Event	weblogic.security.spi.AuditMgmtEvent	Yes	No
Policy Audit Event	weblogic.security.spi.AuditPolicyEvent	Yes	No
Role Deployment Audit Event	weblogic.security.spi.AuditRoleDeploymentEvent	Yes	No
Provider Audit Record	com.bea.security.spi.ProviderAuditRecord	Yes	Yes

The providers implement the appropriate `AuditEvent` interfaces and post those events to the Audit provider. The `AuditEvents` that also implement the `AuditContext` interface can provide more information via a `ContextHandler`.

The `ContextHandler` interface provides a way for an internal WebLogic container to pass additional information to a WebLogic Security Framework call, so that a security provider can obtain additional context information beyond what is provided by the arguments to a particular method. A `ContextHandler` is essentially a name/value list. The name/value list is also called a context element, and is represented by a `ContextElement` object.

What Events are Audited?

Depending on the interface that the `AuditEvent` has implemented, different information is audited. For all audit events, the `toString()` method is called on the event and that string is audited. Some audit events have a `ContextHandler`, such as the `AuditAtzEvent` and `AuditRoleEvent`, in which case the context is audited in addition to calling the `toString()` method on the `AuditEvent`. You can have many `ContextElements`, but each NAME/VALUE pair must be iterated over and audited.

The Log4j Audit Channel provider ships with Log4j renderers that are aware of these interfaces and know how to extract the appropriate audit information. You can change this behavior by writing custom renderers and updating the Custom Log4j Renderer Properties text box on the Advanced tab for the Log4j Auditor page in the Administration Console. A custom renderer is useful if only a particular subset of context elements are required or if the default style of audit events needs to be changed.

Each audit record has the following format:

```
2004-04-22 12:21:55,833 [Thread-27] SUCCESS ASI_AUDIT - My Custom Event -
Custom Event msg -- <attr1 = value1><attr2 = value2>
```

A custom renderer may require square brackets [] instead of angle brackets <>.

To be audited, you can select which severity the audit event must equal or be greater than; and you can select the types of `AuditEvents` by setting the Custom Audit Events attribute. If an `AuditEvent` implements or is an instance of any of the classes listed, then you can audit it. Only new custom events need to be listed here. The default events already exist and are controlled by selecting either: `DISABLED`, `WITH_CONTEXT`, or `WITHOUT_CONTEXT` on the Details tab for the Log4j Auditor page in the Administration Console. For a list of audit events, see [“Audit Events” on page 13-1](#).

Note: Printing the entire context by enabling `WITH_CONTEXT` can be an expensive task and is proportional to the number of context elements contained in the `ContextHandler`.

All audit events generated through the Java API are called through the Provider Audit Records interface using the `AuditRecord` method. This includes `PolicyAdministrationEvent` and `ARMEAuthorizationEvent`. A `PolicyAdministrationEvent` is generated when a policy change is made through the Administration Console. An `ARMEAuthorizationEvent` is generated when the ARME makes a authorization request for a policy change.

All audit events can be `DISABLED` or `WITHOUT_CONTEXT`. For those that have context, you can select `WITH_CONTEXT`. The `AuditAtzEvents` have more options than all the other types, you can select the events to audit based on the following options:

- `DISABLE`—No auditing occurs.
- `WITHOUT_CONTEXT`—Audits what is in the event message.
- `WITH_REQUEST_CONTEXT`—Audits the event message plus the request context.
- `WITH_RESPONSE_CONTEXTS`—Audits the event message plus all the response contexts. Only contains the context that was populated with responses from the ASI Authorization provider. There can be many contexts returned for a single query and hence the `CONTEXTS`.
- `WITH_ALL_CONTEXTS`—Audits the event message plus all the contexts (request as well as response contexts).

Custom Audit Context Extensions

The Log4J Audit Channel provider is used to audit events that are generated by the Security Framework, the runtime API, or custom implementations based on the `weblogic.security.spi.AuditEvent` interface `AuditEvent` class.

Audit plug-ins can be used to audit with minimal awareness of the audit data formats being passed in by the calling Security Framework component. Additionally, Log4j plug-ins written or supplied by third parties can implement actions (such as paging security personal) based on audit severity/criteria you set in the Log4j Audit Channel provider Details tab in the Administration Console. Some general descriptions or suggestions for the information suitable for auditing by `AuditEvent` are as follows:

- Audit events are structured to have a two-tier model. There is a `weblogic.security.spi.AuditEvent` interface that defines the minimum requirements for an audit event. This interface includes `type`, `severity`, `toString()`, and, if there was an exception associated with the event, a reference to the exception.
- In addition to the core `AuditEvent` interface, several additional interfaces are defined to further elaborate on the audit types, and, for providers that need to retrieve audit properties that are specific to the audit type, interfaces exist that allow the providers to extract these values.
- A provider that is not reporting specific event properties can be coded to only recognize the core `AuditEvent` class and to use `toString` to output its representation of the event as a `String`.
- Audit providers that need to do other things (such as selectively log events based on event properties) must be specifically coded to the interfaces described so that they know how to extract these event values from the audit event.

Audit Event Interfaces and Audit Events

In the security provider interface package, WebLogic Security defines one top-level base interface (`AuditEvent`) with seven different derived interfaces that represent the different types of audit events. The following sections describe when the security framework and security providers post these audit events.

- [AuditAtnEvent](#)
- [AuditAtzEvent](#)
- [AuditMgmtEvent](#)
- [AuditCredentialMappingEvent](#)
- [AuditPolicyEvent](#)
- [AuditRoleDeploymentEvent](#)
- [AuditRoleEvent](#)

For a list of the events that are audited for the default Admin policy, see “[Admin Policy Audit Events](#)” on page 13-9.

AuditAtnEvent

Authentication audit events are posted by the security framework. [Table 13-2](#) describes the conditions under which the event is posted and severity level of the event.

Table 13-2 Authentication Audit Events

Component	Description	Severity
Security Framework	Posted after successful authentication of a user.	Success
Security Framework	Posted after unsuccessful authentication (a <code>LoginException</code> thrown from JAAS login method). This <code>LoginException</code> can be thrown by either JAAS framework or by JAAS <code>LoginModule</code> of WebLogic Server authentication provider.	Failure
Security Framework	Posted after an identity assertion to an anonymous user.	Success
Security Framework	Posted after an unsuccessful identity assertion (<code>IdentityAssertionException</code> thrown from identity assertion method).	Failure

Table 13-2 Authentication Audit Events (Continued)

Security Framework	Posted after an unsuccessful identity assertion (IOException is thrown by identity assertion callback handler when retrieving username from callback).	Failure
Security Framework	Posted after an unsuccessful identity assertion (UnsupportedCallbackException is thrown by identity assertion callback handler when retrieving username from callback).	Failure
Security Framework	Posted after an unsuccessful identity assertion (when username returned from identity assertion callback handler is null or zero length).	Failure
Security Framework	Posted after a successful identity assertion.	Success
Security Framework	Posted after an unsuccessful identity assertion.	Failure
Security Framework	Posted after a successful impersonate identity (anonymous identity).	Success
Security Framework	Posted after a successful impersonate identity.	Success
Security Framework	Posted after an unsuccessful impersonate identity.	Failure
Security Framework	Posted after a failure of principal validation.	Failure

AuditAtzEvent

Authorization audit events are posted by the security framework. [Table 13-3](#) describes the conditions under which the events are posted and severity level of the event.

Table 13-3 Authorization Audit Events

Component	Description	Severity
Security Framework	Posted if access is not allowed to resource (exception thrown by authorization provider).	Failure
Security Framework	Posted if access is allowed to resource.	Success
Security Framework	Posted if access is not allowed to resource.	Failure

AuditCredentialMappingEvent

Credential Mapping audit events are posted by the security framework. [Table 13-4](#) describes the condition under which the events are posted and severity level of the event.

Table 13-4 Credential Mapping Audit Events

Component	Description	Severity
Security Framework	Posted after each successful get of credentials.	Success

AuditMgmtEvent

Management audit events are not currently posted by either the security framework or by the supplied providers.

AuditPolicyEvent

AuditPolicyEvent are posted by the security framework and the WebLogic Authorization provider. The security framework posts audit policy events when policies are deployed to or undeployed from an authorization provider. The WebLogic Server authorization provider posts audit policy events when creating, deleting, or updating policies. [Table 13-5](#) describes the conditions under which the events are posted and lists the event severity level.

Table 13-5 Audit Policy Events

Component	Description	Severity
Security Framework	Posted after successful deploy of policy.	Success
Security Framework	Posted after unsuccessful deploy of policy.	Failure
Security Framework	Posted after successful undeploy of policy.	Success

Table 13-5 Audit Policy Events (Continued)

Component	Description	Severity
Security Framework	Posted after an unsuccessful undeploy of policy.	Failure
WebLogic Authorization Provider	Posted after the following events occur: <ul style="list-style-type: none"> • A successful create of policy from console • An unsuccessful create of policy from console (various exceptions) • A successful remove of policy from console • An unsuccessful remove of policy from console (various exceptions) • A successful update of policy from console • An unsuccessful update of policy from console (various exceptions) 	Success

AuditRoleDeploymentEvent

The security framework posts audit role deployment events when roles are deployed to or undeployed from a role mapping provider. [Table 13-6](#) describes the conditions under which the events are posted and lists the event severity level.

Table 13-6 Audit Role Deployment Events

Component	Description	Severity
Security Framework	Posted after each successful role deployment to a role mapping provider.	Success
Security Framework	Posted after each unsuccessful role deployment to a role mapping provider.	Failure
Security Framework	Posted after each successful role undeployment from a role mapping provider.	Success
Security Framework	Posted after each unsuccessful role undeployment from a role mapping provider.	Failure

AuditRoleEvent

The WebLogic Role Mapping provider posts audit role events when roles are created, deleted, or updated. [Table 13-7](#) describes the conditions under which the events are posted and lists the event severity level.

Table 13-7 Audit Role Events

Component	Description	Severity
WebLogic Role Mapping Provider	Posted after the following events occur: <ul style="list-style-type: none"> • A successful create of role from console • An unsuccessful create of role from console (various exceptions) • A successful remove of role from console • An unsuccessful remove of role from console (various exceptions) • A successful update of role from console • An unsuccessful update of role from console (various exceptions) 	Success

Admin Policy Audit Events

[Table 13-8](#) lists and describes the administration policy events that are audited.

Table 13-8 Admin Policy Audit Events

Policy Element	Action	Type	Event Description
Declaration/Attribute	create	declaration	Create a new attribute declaration.
	delete	declaration	Delete an attribute declaration.
	rename	declaration, new_name	Rename an attribute declaration.
	modify	declaration	Modify an attribute declaration.
Declaration/Constant	create	declaration, value	Create a new constant.
	delete	declaration, value	Delete a constant.
	rename	declaration, value, new_name	Rename a constant.

Table 13-8 Admin Policy Audit Events (Continued)

Policy Element	Action	Type	Event Description
Declaration/ Enumeration	modify	declaration, value, new_value	Modify a constant.
	create	declaration, value	Create a new enumeration.
	delete	declaration, value	Delete an enumeration.
	rename	declaration, value, new_name	Rename an enumeration.
Declaration/ Evaluation Function	modify	declaration, value, new_value	Modify an enumeration.
	create	declaration	Create an evaluation function.
	delete	declaration	Delete an evaluation function.
	rename	declaration, new_name	Rename an evaluation function.
Identity/Directory/ Instance	create	directory	Create a directory.
	delete	directory	Delete a directory.
	cascade Delete	directory	Delete a directory and all its users.
	rename	directory, new_name	Rename a directory.
Identity/Directory/ AttributeMapping/ Single	create	attribute, default_value, directory	Add a scalar attribute to a directory attribute schema.
	delete	attribute, default_value, directory	Delete a scalar attribute from a directory attribute schema.
	modify	attribute, default_value, directory, new_default_value	Modify a scalar attribute in a directory attribute schema.
Identity/Directory/ AttributeMapping/ List	create	attribute, default_value, directory	Add a vector attribute to a directory attribute schema.
	delete	attribute, default_value directory	Delete a vector attribute from a directory attribute schema.

Table 13-8 Admin Policy Audit Events (Continued)

Policy Element	Action	Type	Event Description
Identity/Subject/ User	modify	attribute, default_value, directory, new_default_value	Modify a vector attribute in a directory attribute schema.
	create	subject_name	Create a new user.
	copy	subject_name, new_subject_name	Copy a user.
	delete	subject_name	Delete a user.
	cascade Delete	subject_name	Cascade a user and all rules associated with the user.
Identity/Subject/ Group	rename	subject_name, new_subject_name	Rename a user.
	create	subject_name	Create a new group.
	delete	subject_name	Delete a group.
	rename	subject_name, new_subject_name	Rename a group.
	addMember	subject_name, member_subject	Add a member to a group.
Identity/Subject/ Attribute Assignment/ Single	remove Member	subject_name, member_subject	Remove a member from a group.
	create	attribute, value, subject_name	Set a value to a currently unset scalar subject attribute.
	delete	attribute, value, subject_name	Unset a currently set scalar subject attribute.
Identity/Subject/ Attribute Assignment/ List	modify	attribute, value, subject_name, new_value	Modify the value of a currently set scalar subject attribute.
	create	attribute, value, subject_name	Set a value to a currently unset vector subject attribute.

Table 13-8 Admin Policy Audit Events (Continued)

Policy Element	Action	Type	Event Description
	delete	attribute, value, subject_name	Unset a currently set vector subject attribute.
	modify	attribute, value, subject_name, new_value	Modify the value of a currently set vector subject attribute.
	modify	subject_name	Modify the user password. The “subject_name” attribute contains the name of the user with which the password is associated.
Resource/Instance	create	resource, resource_type	Create a new resource.
	delete	resource	Delete a resource.
	cascade Delete	resource	Cascade delete of a resource. This includes deletion of all child resources and associated rules.
	rename	resource, new_name	Rename a resource.
Resource/Attribute Assignment/Single	create	attribute, resource, value	Set a value to a currently unset scalar resource attribute.
	delete	attribute, resource, value	Unset a currently set scalar resource attribute.
	modify	attribute, resource, value, new_value	Modify the value of a currently set scalar resource attribute.
Resource/Attribute Assignment/List	create	attribute, resource, value	Set a value to a currently unset vector resource attribute.
	delete	attribute, resource, value	Unset a currently set vector resource attribute.
	modify	attribute, resource, value, new_value	Modify the value of a currently set vector resource attribute
Resource/Metadata/IsApplication	modify	resource, value, new_value	Toggle the “is application” resource metadata.

Table 13-8 Admin Policy Audit Events (Continued)

Policy Element	Action	Type	Event Description
Resource/Metadata/ IsDistributionPoint	modify	resource, value, new_value	Toggle the “is distribution point” resource metadata.
Resource/Metadata/ Logical Name	create	logical_name, resource	Create a logical name for a resource.
	delete	logical_name, resource	Delete the logical name of a resource.
	rename	logical_name, resource, new_name	Rename the logical name of a resource.
Policy/Rule/Grant	create	action, resource, subject_name, constraint	Create a new grant rule. The “action”, “resource”, and “subject_name” attributes are lists.
	delete	action, resource, subject_name, constraint	Delete a grant rule. The “action”, “resource”, and “subject_name” attributes are lists.
	modify	action, resource, subject_name, constraint, new_action, new_resource, new_subject_name, new_constraint	Modify a grant rule. The “action”, “resource”, and “subject_name” attributes are lists.
Policy/Rule/Deny	create	action, resource, subject_name, constraint	Create a new deny rule. The “action”, “resource”, and “subject_name” attributes are lists.
	delete	action, resource, subject_name, constraint	Delete a deny rule. The “action”, “resource”, and “subject_name” attributes are lists.
	modify	action, action_type, resource, subject_name, subject_type, constraint, new_effect, new_action, new_action_type, new_resource, new_subject_name, new_subject_type, new_constraint	Modify a deny rule. The “action”, “resource”, and “subject_name” attributes are lists.

Table 13-8 Admin Policy Audit Events (Continued)

Policy Element	Action	Type	Event Description
Policy/Rule/Delegate	create	action, resource, subject_name, delegator, constraint	Create a new delegate rule. The “action”, “resource”, and “subject_name” attributes are lists.
	delete	action, resource, subject_name, delegator, constraint	Delete a delegate rule. The “action”, “resource”, and “subject_name” attributes are lists.
	modify	action, resource, subject_name, delegator, constraint, new_action, new_resource, new_subject_name, new_delegator, new_constraint	Modify a delegate rule. The “action”, “resource”, and “subject_name” attributes are lists.
Policy/Action/Role/Instance	create	action	Create a new role.
	delete	action	Delete a role.
	rename	action, new_name	Rename a role.
Policy/Action/Privilege/Instance	create	action	Create a privilege.
	delete	action	Delete a privilege.
	rename	action, new_name	Rename a privilege.
Policy/Action/Privilege/Group	create	action_group	Create a privilege group.
	delete	action_group	Delete a privilege group.
	rename	action_group, new_name	Rename a privilege group.
	addMember	action_group, action	Add a privilege to a privilege group.
	remove Member	action_group, action	Remove a privilege from a privilege group.

Table 13-8 Admin Policy Audit Events (Continued)

Policy Element	Action	Type	Event Description
Policy/Analysis/ Inquiry Query	create	title, owner, effect_type, subjects, actions, resources, delegator	Create a new policy query.
	delete	title, owner	Delete a policy query.
	modify	title, owner, effect_type, subjects, actions, resources, delegator	Modify a policy query.
	execute	title, owner, effect_type, subjects, actions, resources, delegator	Execute a policy query. If this is an unsaved query “title” and “owner” is set to an emptystring.
Policy/Analysis/ Verification Query	create	title, owner, actions, resources	Create a new policy verification query.
	delete	title, owner	Delete a policy verification query.
	modify	title, owner, actions, resources	Modify a policy verification query.
	execute	title, owner, actions, resources	Execute a policy verification query. If this is an unsaved query “title” and “owner” is set to an emptystring.
Policy/Repository	deploy Update	resource, directory	Deploy a policy update. The “resource” is the distribution node; all nodes below it may be effected. This check is made for each chosen distribution point
	deploy Structural Change	deleted_directories, deployed_engines, deleted_engines, deleted_bindings, deleted_applications	Deploy a structural change.
Infrastructure/Engines /ARME	create	engine	Create a new SSM.
	delete	engine	Delete an SSM.

Table 13-8 Admin Policy Audit Events (Continued)

Policy Element	Action	Type	Event Description
Infrastructure/Engines /SCM	rename	engine, new_name	Rename an SSM.
	bind	engine, resource	Bind a resource to an SSM.
	unbind	engine, resource	Unbind a resource from an SSM.
	create	engine	Create an SCM.
	delete	engine	Delete an SCM.
	rename	engine, new_name	Rename an SCM.
	bind	engine, resource	Bind an SSM to an SCM. A “resource” contains the name of the SSM.
	unbind	engine, resource	Unbind an SSM from an SCM. A “resource” contains the name of the SSM.
Infrastructure/ Management/Console	login		Login to the WLES administration console.
Infrastructure/ Management/Loader	login		Login to the WLES policy loader.

Additional Audit Event Interfaces

The following sections describe additional audit event interfaces:

- [“Authentication - AuditAtnEvent” on page 13-17](#)
- [“Policy Deployment - AuditPolicyDeployEvent” on page 13-18](#)
- [“Policy Undeployment - AuditPolicyUndeployEvent” on page 13-18](#)
- [“Policy Events - AuditPolicyEvent” on page 13-19](#)
- [“Role Mapping - AuditRoleEvent” on page 13-19](#)
- [“Role Deployment - AuditRoleDeployEvent” on page 13-20](#)

- [“Role Undeployment - AuditRoleUndeployEvent” on page 13-20](#)
- [“Predicate Events - AuditPredicateEvent” on page 13-20](#)
- [“ContextHandler Object” on page 13-20](#)
- [“PolicyAdministrationEvent” on page 13-21](#)

Authentication - AuditAtnEvent

The `AuditAtnEvent` interface provides an interface for audit providers to determine the instance types of the extended authentication event type objects. [Table 13-9](#) describes the event properties.

Table 13-9 Authentication - AuditAtnEvent

Event Property	Description
AUTHENTICATE	Represents the "simple authentication" authentication type.
USERLOCKED	Indicates that a user was locked because of a series of failed login attempts.
USERLOCKOUTEXPIRED	Indicates that a lock on a user has expired.
USERUNLOCKED	Indicates that a lock on a user was cleared.
ASSERTIDENTITY	Represents the identity assertion authentication token type.
IMPERSONATEIDENTITY	Represents the impersonate identity authentication type.

When this event is generated, the following information associated with this `AuditAtnEvent` is available:

- The username associated with this `AuditAtnEvent`; that is, the username of the person who is attempting authentication.
- The event type associated with this `AuditAtnEvent`
- Authorization - `AuditAtzEvent`

There are both pre- and post-authorization access control checks; each of which generates pre- and post-operation audit write events. The `AuditAtzEvent` event interface is used to report events that result when access is allowed on a resource. The Audit Channel provider is called on

both the pre- and post-operation cases. The exceptions reported using this event must derive from the `java.security.GeneralSecurityException`.

When this event is generated, the following information associated with this `AuditAtzEvent` is available:

- The name of the resource
- The name of the subject
- The `ContextHandler` object

The resource container that handles the type of resource requested (for example, in WebLogic Server 8.1, the EJB container receives the request for an EJB resource) constructs a `ContextHandler` object that may be used by an authorization provider `Access Decision` to obtain information associated with the context of the request. This `ContextHandler` object is also available with this `AuditAtzEvent`. For more information about the `ContextHandler` object, see [“ContextHandler Object” on page 13-20](#).

Policy Deployment - `AuditPolicyDeployEvent`

The `AuditPolicyDeployEvent` event interface is used when the Authorization Manager `deployPolicy` method is called. When this event is generated, the following information is available:

- The subject whose action is being audited
- The severity of this event
- The resource that is the target of action being audited
- A string array of role names for this policy
- The exception that occurred (if any) while attempting to carry out this action. Typically, there will only be an exception if the severity is error or failure.

Policy Undeployment - `AuditPolicyUndeployEvent`

The `AuditPolicyUndeployEvent` event interface is used when the Authorization Manager `undeployPolicy` method is called. When this event is generated, the following information is available:

- The subject whose action is being audited
- The severity of this event

- The resource that is the target of action being audited
- A string array of role names for this policy

The exception that occurred (if any) while attempting to carry out this action. Typically, there is only an exception if the severity is error or failure.

Policy Events - AuditPolicyEvent

The `AuditPolicyEvent` event interface determines the instance types of extended Authorization event type objects. [Table 13-10](#) describes the event subtypes.

Table 13-10 Policy Event- AuditPolicyEvent

Event Subtype	Description
DEPLOY	Indicates that a policy deployment event occurred.
UNDEPLOY	Indicates that a policy undeployment event occurred.
UPDATE	Indicates that a policy was updated.

Role Mapping - AuditRoleEvent

The `AuditRoleEvent` event provides an interface for auditing providers to determine the instance types of extended Role Mapping event type objects. [Table 13-11](#) describes the event subtypes.

Table 13-11 Role Mapping - AuditRoleEvent

Event Subtype	Description
DEPLOY	Indicates that a role mapping deployment event occurred.
UNDEPLOY	Indicates that a role mapping undeployment event occurred.
UPDATE	Indicates that a role mapping was updated.

When an `AuditRoleEvent` is generated, the following information is available:

- The Subject that is attempting to access the resource associated with this `AuditRoleEvent`
- The resource attempting to be accessed by the subject associated with this `AuditRoleEvent`

- The `ContextHandler` object

The resource container that handles the type of resource being requested (for example, with WebLogic Server 8.1, the EJB container receives the request for an EJB resource) constructs a `ContextHandler` object that may be used by an Authorization provider Access Decision to obtain information associated with the context of the request. This `ContextHandler` object is also available with this `AuditAtzEvent`. For more information about the `ContextHandler` object, see [“ContextHandler Object” on page 13-20](#).

Role Deployment - `AuditRoleDeployEvent`

The `AuditRoleDeployEvent` event provides a interface used by the role mapping service to determine the instance types of extended Role Mapping deployment event type objects.

Role Undeployment - `AuditRoleUndeployEvent`

The `AuditRoleUndeployEvent` event provides a interface used by the role mapping service to determine the instance types of extended Role Mapping undeployment event type objects.

Predicate Events - `AuditPredicateEvent`

The `AuditPredicateEvent` event provides a interface for auditing providers to determine the instance type of extended predicate event type objects. A predicate event occurs when a policy expression is either registered or unregistered in the Administration Console. [Table 13-12](#) describes the event subtypes.

Table 13-12 Predicate Events - `AuditPredicateEvent`

Event Subtype	Description
REGISTER	Occurs when a policy expression is registered.
UNREGISTER	Occurs when a policy expression is registered.

`ContextHandler` Object

A `ContextHandler` is a class that obtains additional context and container-specific information from the resource container, and provides that information to security providers making access or role mapping decisions. The `ContextHandler` interface provides a way for an application or container to pass additional information to a Security Framework call, so that a security provider can obtain contextual information beyond what is provided by the arguments to a particular method. A `ContextHandler` is essentially a name/value list and as such, it

requires a security provider to know what names to look for. In other words, use of a `ContextHandler` requires close cooperation between the resource container and the security provider. Each name/value pair in a `ContextHandler` is known as a context element, and is represented by a `ContextElement` object.

A context handler is an object that is included with some event types that allows an audit provider to extract other information about the state of the application server at the time of the audit event. The audit provider may log this other contextual information as a way to elaborate on the event and provide other useful information about the causes of the event.

PolicyAdministrationEvent

The `PolicyAdministrationEvent` event is used when WebLogic Enterprise Security policy is modified or deployed using the WebLogic Enterprise Security Administration console or bulk loader. When this event is generated, the following information is available:

- The subject whose action is being audited
- The severity of this event
- The resource that is the target of action being audited
- Detailed information regarding the change being made

The exception that occurred (if any) while attempting to carry out this action. Typically, there will only be an exception if the severity is error or failure.

Using Custom Audit Providers

You can use a custom auditing provider instead of the Log4j Audit Channel provider. For a custom auditing provider to be configurable through the Administration Console, the MBean JAR file for the provider must be installed into the `BEA_HOME.../lib/providers` directory on both the machine on which the Administration Application is installed and on the machine on which the Security Service Module is installed. For complete instructions for configuring a custom security provider, see [“Configuring a Custom Security Provider” on page 9-45](#).

Function Reference

The ASI Authorization and Role Mapper providers use an external server process to evaluate authorization decisions. This process is called the Authorization and Role Mapping Engine (ARME).

This section describes the Application Programming Interface (API) for writing custom extension libraries (plug-ins) for this process to enhance features available through the policy language, such as routines for dynamic computation of an attribute value (credential function) or custom predicate (evaluation function).

Function Pointers

The following plug-in function pointers are described in this API:

[“*CredFunc\(\) - Custom Credential Function Pointer” on page 14-2](#)

[“*EvalFunc\(\) - Custom Evaluation Function Pointer” on page 14-3](#)

[“*ShutdownFunc \(\) - Custom Shutdown Function Pointer” on page 14-4](#)

[“*PluginInitFunc\(\) - Plug-in Initialization Function Pointer” on page 14-5](#)

[“registerCustomCredentialFunction\(\) - Register Credential Function” on page 14-6](#)

[“registerCustomEvaluationFunction\(\) - Register Evaluation Function” on page 14-7](#)

[“registerShutdownFunction\(\) - Register Shutdown Function” on page 14-8](#)

These extension functions (called plug-ins) take two kinds of parameters: required and optional. Parameters also have one of two modes: in or out. Input parameters pass data to an extension. Output parameters receive data from an extension.

*CredFunc() - Custom Credential Function Pointer

Description

Pointer to a function that computes the value of a credential function. Your plug-in function is internally referenced by this pointer. The `registerCustomCredentialFunction()` function assigns the pointer to your function. For example, this function declares a credential function:

```
bool MyCredentialFunction (Session &sess, const char *cvarname)
```

The ARME then connects a function pointer (assign a handle) to your function:

```
returnCode = registerCustomCredentialFunction( "MyCredentialFunctionsName",  
MyCredentialFunction);
```

Syntax

```
bool (*CredFunc)(Session &sess, const char *cvarname);
```

Parameters

***cvarname** is a required input parameter, that points to a null-terminated character string (a character array) and contains the name of a credential function.

&sess is a required output parameter that references the address of a Session object and a session that contains the computed value of the credential function.

Returns

TRUE if the computation is successful

FALSE if the computation is unsuccessful

Example

```
bool GetAccountID(Session &sess, const char *cvarname)  
{  
    ...  
}
```

See Also

`*EvalFunc()`

*EvalFunc() - Custom Evaluation Function Pointer

Points to a function that computes the value of an evaluation function. Your plug-in function is internally referenced by the ARME through this pointer. The `registerCustomEvaluationFunction()` function assigns the pointer to your plug-in function. For example, you could declare a evaluation function like this:

```
TruthValue MyEvalFunction(Session &sess, const char *fname, char **argv);
```

The ARME then connects a function pointer (assign a handle) to your function:

```
returnCode = registerCustomEvaluationFunction("MyEvaluationFunctionsName",  
MyEvalFunction);
```

Syntax

```
TruthValue (*EvalFunc)(Session &sess, const char *fname, char **argv);
```

Parameters

&sess is a required output parameter that references the address of a `Session` object and a session that contains the computed value of the credential variable.

***fname** is a required input parameter that points to a null-terminated character string (a character array) and contains the name of an evaluation function.

****argv** is a required input parameter and is a pointer to a null-terminated array (also a pointer) of null-terminated strings (character arrays), containing a list of arguments for the function.

Returns

`TruthValue` enumerated list, containing one of three constants:

`TV_TRUE` is true

`TV_FALSE` is false

`TV_UNKNOWN` is an error or undefined result

Example

```
TruthValue isValidAccountID(Session &sess, const char* fname, char **argv)  
{...}
```

See Also

`*CredFunc()`

*ShutdownFunc () - Custom Shutdown Function Pointer

Points to a function that is called when the ARME is about to be shutdown. Put all cleanup code inside this function that you want the plug-in to perform before the shutdown; for example, open windows handlers, open file pointers, and database connections. Your plug-in function is internally referenced by the ARME by this pointer. The `registerShutdownFunction()` function assigns the pointer to your plug-in function. For example, to declare a shutdown function, use the following:

```
MyShutdownFunction ()
```

Assign a handle to your function to connect the ARME to your function pointer:

```
returnCode = registerShutdownFunction(MyShutdownFunction);
```

Syntax

```
void (*ShutdownFunc)();
```

Parameters

None

Returns

Nothing

Example

```
void MyShutdownFunction()  
{  
    // Code that needs to run for the plugin to release  
    // any resources or open connections that it created  
}
```

See Also

`*PluginInitFunc()`

*PluginInitFunc() - Plug-in Initialization Function Pointer

Points to a function that initializes an ARME plug-in. The name of this function must appear in the ARME configuration file. For example, for the following statement, include the function name `initMyplugins()` in the ARME configuration file.

```
ARME.Instance1.plugin1 F:/BEA/MyPlugins/ Plugin01.dll (initMyplugins  
'plugin1')
```

Syntax

```
void (*PluginInitFunc)(const char *argp);
```

Parameters

***argp** is a required input parameter and is a pointer to a null-terminated array of null-terminated character strings that contains a list of arguments for the initialization.

Returns

Nothing

Example

```
void initPlugin(const char *pluginName)  
{  
    ...  
}
```

registerCustomCredentialFunction() - Register Credential Function

Registers a credential function with a ARME.

Syntax

```
bool registerCustomCredentialFunction(const char *credname, CredFunc credfunc);
```

Parameters

***credname** is a required input parameter and points to a null-terminated character string that contains the name of a custom credential function.

credFunc is a required input parameter used to register the credential function.

Returns

TRUE if the registration is successful

FALSE if the registration is unsuccessful

Example

```
returnCode = registerCustomCredentialFunction("MyCredentialFunctionsName",  
MyCredentialFunction);
```

See Also

```
registerCustomEvaluationFunction()
```

registerCustomEvaluationFunction() - Register Evaluation Function

Registers a custom evaluation function with a ARME.

Syntax

```
bool registerCustomEvaluationFunction(const char *evalname, CredFunc  
evalfunc);
```

Parameters

***evalname** is a required input parameter and points to a null-terminated character string that contains the name of an evaluation function.

evalFunc is a required input parameter used to register the evaluation function.

Returns

TRUE if the registration is successful

FALSE if the registration is unsuccessful

Example

```
returnCode = registerCustomEvaluationFunction("MyEvaluationFunctionsName",  
MyEvalFunction);
```

See Also

```
registerCustomCredentialFunction()
```

registerShutdownFunction() - Register Shutdown Function

Registers a shutdown function.

Syntax

```
bool registerShutdownFunction(ShutdownFunc shutdownfunc);
```

Parameters

ShutdownFunc is a required input parameter used to register the function.

Returns

TRUE if the registration is successful

FALSE if the registration is unsuccessful

Example

```
returnCode = registerShutdownFunction(MyShutdownFunction);
```

See Also

`registerCustomEvaluationFunction()`

`registerCustomCredentialFunction()`

Session Class

A session object keeps all data related to a single access or role query. All attributes used during evaluation are accessible through the session object.

Subject and resource attributes are always accessible. A dynamic attribute, either computed by an evaluation function or passed in, may be accessible, but this is not guaranteed as they are loaded as needed. A attribute used as an argument for an evaluation function is computed before that function is invoked.

The following session objects are described in this API:

[“Session::SetAttribute\(\) - Append AttributeValue Object” on page 14-10](#)

[“Session::getAttribute\(\) - Get AttributeValue Object from Attribute” on page 14-12](#)

[“Session::getEvalResult\(\) - Get Evaluation Result” on page 14-13](#)

[“Session::appendReturnData\(\) - Return Evaluation Results” on page 14-14](#)

[“Session::getDomainName\(\) - Get Domain Name for the Session” on page 14-16](#)

[“Session::getLocationName\(\) - Get Location Name for Session” on page 14-17](#)

[“Session::getApplicationName\(\) - Get Application Name for Session” on page 14-18](#)

[“Session::getUserID\(\) - Get User Name for Session” on page 14-19](#)

Session::SetAttribute() - Append AttributeValue Object

Sets an attribute value for a credential attribute by appending an `AttributeValue` object to the named credential.

Syntax

```
bool setAttribute(const char *name, AttributeValue *value, bool overwrite)
```

Parameters

***name** is a required input parameter that points to a null-terminated character string (a character array), referencing the name of the credential attribute. This parameter must be passed a `const char *` and not as a `char *` variable. Simply placing the name directly into the function (in quotation marks) does not work.

***value** is a required input parameter that points to an `AttributeValue` object and contains the value of the credential. Allocate this object with a new operator because the session object takes ownership of the object.

Warning: Passing in a pointer to a local object or deleting this pointer inside your plug-in may lead to unexpected results, most likely terminating the ARME process.

overwrite is a required input parameter Boolean value (`TRUE` or `FALSE`), that determines whether to overwrite existing attributes of the same name. During evaluation, values set with this call take precedence over static or dynamic values defined for the same credential attribute. There are no safeguards to prevent modifying the values of these credential attributes. You should set the value of a credential function only for the attribute to which it is registered.

Returns

`TRUE` if operation was successful

`FALSE` if it was not successful

Example

```
AttributeValue* SomeAttributeValue = new AttributeValue(false);  
  
const char *CredName = "SongCount";  
  
const char *Value = "small";  
  
SomeAttributeValue.SetValue(Value);
```



```
retcode = sess.setAttribute(CredName, SomeAttributeValue, true);
```

Warning: Some `AttributeValue` methods may cause an exception and memory leak if you call them before calling `setAttribute()`.

See Also

`getAttribute()`

Session::getAttribute() - Get AttributeValue Object from Attribute

Gets an attribute value for a credential attribute by requiring that you pass it the address of an existing `AttributeValue` object. You can use `getAttribute("my_roles", SomeAttributeValue)` to retrieve a list containing the role for a subject.

Warning: BEA strongly recommends that you not modify the contents of the `AttributeValue` object. Some `AttributeValue` objects are shared between credentials objects and evaluation threads. Manipulating their values can lead to unexpected results.

Syntax

```
bool getAttribute(const char * name, AttributeValue *& value)
```

Parameters

***name** is a required input parameter that points to a null-terminated character string (a character array) and references the name of a credential attribute. You must pass this parameter as a `const char *` and not a `char *` variable. Simply putting the name directly into the function in quotation marks does not work.

***&value** is a required input parameter that points to an `AttributeValue` object and contains the value of the attribute.

Returns

`TRUE` if operation was successful

`FALSE` if it was not successful

Example

```
const char *inputvalue = "LillyLiverCount";  
  
retcode = sess.getAttribute(inputvalue, SomeAttributeValue);
```

See Also

`setAttribute()`

Session::getEvalResult() - Get Evaluation Result

Returns a pointer to a temporary `EvalResult` object containing the data returned, along with the access decision upon the execution of the rule containing the evaluation or credential function. The plug-in does not know in advance if the rule containing the plug-in function executes (other rules that preclude it may execute first), so modifying this object does not guarantee the information is returned. BEA recommends using the `appendReturnData()` method. In many cases, you may find using the `report()` and `report_as()` functions easier and more flexible.

Syntax

```
EvalResult* getEvalResult();
```

Parameters

None

Returns

Pointer to an `EvalResult` object containing the evaluation results (output attributes) that may be returned by the executing rule.

Example

```
EvalResult *MyEvalResultPtr = getEvalResult();
```

See Also

```
getEnumValue()
```

Session::appendReturnData() - Return Evaluation Results

Sets the evaluation results returned through an `EvalResult` object upon successful execution of a rule containing the evaluation or credential function that references the plug-in function. It does so by copying the contents of the `AttributeValue` object that is passed to it.

If the same attribute value is redefined by another plug-in within the same rule, the return value is overwritten.

The evaluation result is returned only if the rule actually executes. It might not if another rule makes evaluating it superfluous. For example, once a user is explicitly denied access with a deny rule, a thousand grant rules cannot undo the one deny. Knowing this, as soon as a single grant rule for an access attempt is found, BEA WebLogic Enterprise Security only looks for deny rules. When a single deny rule is found, it stops looking.

The exception to this is if the `findAllFacts` flag is enabled, which you have to consciously set. When you set the `findAllFacts` flag to true, you are telling BEA WebLogic Enterprise Security to return the rule attributes for all rules affecting the access attempt, whether or not the rules are redundant. Setting the `findAllFacts` flag to true does change the fact that once a deny rule has executed, grant rules are not evaluated. The sole purpose is to evaluate and possibly append return results from all deny rules or all grant rules if no deny rules fired.

Warning: Setting the `findAllFacts` flag to true dramatically slows down each access attempt because all rules are checked every time. Avoid setting the flag to true, unless there is no other way to accomplish your goal.

Each `EvaluationResult` object contains the relevant rule identification number, the name of the object and privilege in that rule, and a collection of return attributes in list form. Single values are represented as single-value lists.

Warning: Unlike the `setAttribute()` method, this method copies the content of the data. If you allocate the `AttributeValue` object with the new operator, you have to delete the object. If you do not, a memory leak may result. You are free to pass in a pointer to a local variable, like that retrieved by the `getAttribute()` method.

Syntax

```
void appendReturnData(const char *name, const AttributeValue *data);
```

Parameters

***name** is a required input parameter that points to a null-terminated character string (a character array) and references the name of the output credential attribute. This parameter must be passed

a `const char *` and not a `char *` variable. Simply putting the name directly into the function in quotation marks does not work.

***data** is a required input parameter that points to an `AttributeValue` object and contains the value of the attribute.

Returns

Nothing

Example

```
AttributeValue LocalValue(true);  
  
const char *OutputAttributeName = "SomeCount";  
  
const char *ValueSample = "shopping";  
  
LocalValue.addValue(ValueSample);  
  
appendReturnData(OutputAttributeName, &LocalValue);
```

See Also

`getEvalResult()`

Session::getDomainName() - Get Domain Name for the Session

Retrieves the name of the domain for the session object to which it belongs.

Syntax

```
const char *getDomainName() const;
```

Parameters

None

Returns

Points to a null-terminated character string (a character array) and references the name of the session domain.

Example

```
const char *CLDomainNameStr = getDomainName();
```

See Also

```
getLocationName()
```

Session::getLocationName() - Get Location Name for Session

Retrieves the name of the location for the session object to which it belongs.

Syntax

```
const char *getLocationName() const;
```

Parameters

None

Returns

Points to a null-terminated character string (a character array) and references the name of the session location.

Example

```
const char *CLLocationNameStr = getLocationName();
```

See Also

```
getDomainName()
```

Session::getApplicationName() - Get Application Name for Session

Retrieves the name of the application for the session object to which it belongs.

Syntax

```
const char *getApplicationName() const;
```

Parameters

None

Returns

Points to a null-terminated character string (a character array) and references the name of the session application.

Example

```
const char *CLAppNameStr = getApplicationName();
```

See Also

```
getUserID()
```


Session::getUserId() - Get User Name for Session

Retrieves the name of the user for the session object to which it belongs.

Syntax

```
const char *getUserId() const;
```

Parameters

None

Returns

Points to a null-terminated character string (a character array) and references the name of the session user.

Example

```
const char *CLUserNameStr = getUserId();
```

See Also

```
getApplicationName()
```

AttributeValue Class

Contains the methods for manipulating `AttributeValue` objects, which is how attribute values are stored for each user credential attribute.

This section describes the following `AttributeValue` objects:

[“`AttributeValue::addValue\(\)` - Add and Set a String List Attribute Value” on page 14-22](#)

[“`AttributeValue::AttributeValue\(\)` - Constructor” on page 14-23](#)

[“`AttributeValue::entries\(\)` - Count Number of List Elements” on page 14-25](#)

[“`AttributeValue::getValue\(\)` - Get Single Attribute Value” on page 14-26](#)

[“`AttributeValue::has\(\)` - Check If Value is Already Present in a List” on page 14-27](#)

[“`AttributeValue::IsList\(\)` - Is Attribute Value an Indexed List?” on page 14-28](#)

[“`AttributeValue::IsSingle\(\)` - Is Attribute Value a Single Value?” on page 14-29](#)

[“`AttributeValue::isUndefined\(\)` - Is Attribute Value an undefined object?” on page 14-30](#)

[“`AttributeValue::setValue\(\)` - Set Single Attribute Value” on page 14-31](#)

[“`AttributeValue::removeAt\(\)` - Remove Indexed List Attribute Value” on page 14-32](#)

[“`AttributeValue::removeValue\(\)` - Remove Named List Attribute Value” on page 14-33](#)

[“`AttributeValue::size\(\)` - Count Number of List Elements” on page 14-34](#)

[“`AttributeValue \[\]` Operator - Returns the Value of an Indexed String List Element” on page 14-35](#)

`AttributeValue` objects store values in one of two ways:

- As a single string value
- As an array-like list of strings

Single Value

Single values are nothing more than one string of characters per `AttributeValue` object and are the most common type of attribute value. The methods that are exclusively for manipulating single values are:

`setValue()`

`getValue()`

Lists of Values

Lists of values are collections of strings for one `AttributeValue` object. They are accessed by their index value (in square brackets), just like a one-dimensional array. Normally, list values are not set with duplicate string values. The methods that are exclusively for manipulating lists of values are:

```
addValue()
removeValue()
removeAt()
entries()
```

There is also an undefined type of `AttributeValue` object created by a default constructor with no arguments. It becomes either a single or a list type after the first call to the `setValue()` method (single), `addValue()` method (list), or `operator=()`.

Methods Common to Both Types

Several methods are available for both single and list attributes. Most of them are for testing what type of attribute you have. These include:

```
size() (returns 1 for single value, or entries() for list value)
operator[0] (same as getValue() for index 0), or operator[] for list
type
isList()
isSingle()
isUndefined()
has()
```

Internal Methods

The header files contain many methods related to this class and ones that are not documented. These are used internally by the ARME and are not recommended for use. BEA cannot guarantee these methods work at all or will work in the future due to deprecation. These include `operator+=()`, `operator-=()`, all those taking non-const `char*`, enumeration-related, and type-checking-related methods.

Warning: BEA does not recommend using any method not documented in this guide.

AttributeValue::addValue() - Add and Set a String List Attribute Value

Adds and sets a string value to a string list `AttributeValue` object. Use `isList()` to determine if the object contains a list of values. If you call this method for an undefined `AttributeValue` object, it becomes a list type.

Syntax

```
void addValue(const char *value, bool check_unique = false)
```

Parameters

***value** is a required input parameter that points to a null-terminated character string (a character array). This parameter adds an element to the end of a string list increasing the list index value by one and must be passed a `const char *` and not a `char *` variable. Simply putting the value directly into the function in quotation marks does not work.

check_unique is an input parameter that defaults to false. If this parameter is set to `TRUE`, duplicate values are not added to the list.

Returns

Nothing

Example

```
const char *FavoriteSongValue = "foreveryours";  
if (FaveSongAvObj.isList())  
    FaveSongAvObj.addValue(FavoriteSongValue);
```

See Also

```
setValue()  
getValue()  
isList()
```

AttributeValue::AttributeValue() - Constructor

The `AttributeValue` class constructor is polymorphic. You can pass it different sets of parameters depending on your goals.

Syntax

For an undefined attribute:

```
AttributeValue();
```

For a single-value attribute:

```
AttributeValue(const char *value);
```

For an empty list of values or a single value:

```
AttributeValue(bool isList);
```

To copy an existing `AttributeValue`:

```
AttributeValue(const AttribtueValue &value);
```

Parameters

***value (String)** is a required input parameter used to create a single-value attribute that points to a null-terminated character string (a character array). This parameter references the name of a credential attribute and must be passed a `const char *` and not a `char *` variable. Simply putting the value directly into the function in quotation marks does not work.

&value (AttributeValue Object) is a required input parameter (for copying an `AttributeValue` object) that points to an `AttributeValue` object.

isList is set to `TRUE` if the attribute and contains an enumerated list of strings; set to `FALSE` for a single value attribute.

Returns

Nothing

Example

For a single-value attribute:

```
const char *MyAttrName = "SomeCount";

AttributeValue *MyAttrValPtr = new
```

```
AttributeValue (MyAttrName) ;
```

For a list of values:

```
AttributeValue *MyAVListPtr = AttributeValue (true) ;
```

To copy an existing AttributeValue object:

```
AttributeValue *MyAttrValPtr = new  
AttributeValue (MyExistingAttrValObj) ;
```

See Also

`setValue()`

AttributeValue::entries() - Count Number of List Elements

Returns a count of the total number of elements in an indexed list.

Syntax

```
int entries() const
```

Parameters

None

Returns

An integer that contains the total count of elements.

Example

```
if (FaveSongAvObj.isList())int elementCount = FaveSongAVObj.entries();
```

See Also

`addValue()`

`AttributeValue::size()`

AttributeValue::getValue() - Get Single Attribute Value

Gets a single string value for an `AttributeValue` object. Use `isSingle()` to determine if the object contains a single value.

Syntax

```
const char *getValue() const
```

Parameters

None

Returns

A pointer to a constant, null-terminated character string (a character array) that contains the string value of the `AttributeValue` object. This value must be a `const char *` and not a `char *` variable.

Example

```
const char *FavoriteSongValue;  
  
if (FaveSongAvObj.isSingle()) FavoriteSongValue =  
FaveSongAVObj.getValue();
```

See Also

`addValue()`

`setValue()`

`isSingle()`

AttributeValue::has() - Check If Value is Already Present in a List

Checks if a value is present in a list or if a single-value attribute is equal to it. `AttributeValue` objects of the list type should not contain redundant values.

Syntax

```
bool has(const char * value) const
```

Parameters

***value**

is a required input parameter

Points to a null-terminated character string (a character array)

Returns

TRUE if the supplied value is present in a list, or is equal to the value of a single type value.

Example

```
Bool A = AttributeValue.has("SomeValue");
```

See Also

`addValue()`

`setValue()`

`AttributeValue::size()`

AttributeValue::IsList() - Is Attribute Value an Indexed List?

Returns `TRUE` if the `AttributeValue` object contains a list of strings or integers. The list attribute values are accessed by their index value.

Syntax

```
bool isList() const;
```

Parameters

None

Returns

`TRUE` if the attribute contains a list of values

Example

```
bool AttrIsList = MyAttrValueObj.isList();
```

See Also

```
isSingle()
```

AttributeValue::IsSingle() - Is Attribute Value a Single Value?

Returns `TRUE` if the `AttributeValue` object contains a single value and not a list of values.

Syntax

```
bool isSingle() const;
```

Parameters

None

Returns

`TRUE` if the attribute is a single-value type

Example

```
bool AttrIsSingle = MyAttrValueObj.isSingle();
```

See Also

```
isList()
```

AttributeValue::isUndefined() - Is Attribute Value an undefined object?

Returns `TRUE` if the `AttributeValue` object is constructed with a default constructor and does not have a list or single type. The type can be set by calling `setValue()`, or `addValue()` methods and becomes a single value or a list correspondingly, or by using an assignment operator. Once set, the type cannot be changed.

Syntax

```
bool isUndefined() const;
```

Parameters

None

Returns

`TRUE` if the attribute is undefined

Example

```
bool AttrIsSingle = MyAttrValueObj.isUndefined();
```

See Also

```
isList()
```

AttributeValue::setValue() - Set Single Attribute Value

Sets a single string value for an `AttributeValue` object. Use `isSingle()` to determine if the object is the correct type for a single value. If you call this method for an undefined `AttributeValue` object, it becomes a single type.

Syntax

```
void setValue(const char *value)
```

Parameters

***value** is a required input parameter that points to a null-terminated character string (a character array) and sets the string value of an `AttributeValue` object. This parameter must be passed a `const char *` and not a `char *` variable. Simply putting the value directly into the function in quotation marks does not work.

Returns

Nothing

Example

```
const char *FavoriteSongValue = "foreveryours";  
FaveSongAVObj.setValue(FavoriteSongValue);
```

See Also

`addValue()`

`getValue()`

AttributeValue::removeAt() - Remove Indexed List Attribute Value

Removes a value identified by index in a string or integer list `AttributeValue` object. Use `isList()` to ensure that it is a list of values.

Syntax

```
char *removeAt(int idx)
```

Parameters

idx is a required input parameter with an integer value that removes a string or integer element in the list by the index value of the element (index starts at 0).

Returns

A pointer to the string representing the value. This is provided as a convenience to use the value so you do not have to retrieve the value with a separate call.

Warning: You must delete the returned value with the `delete[]` operator within the plug-in.

Example

```
//Remove the second value, which is "Apple":  
char * MrValue = someAttrValuePtr->removeValue(1);  
//MrValue now points to "Apple" if the index position  
//contained such value, to null if not..  
//Delete value, even if null:  
delete[] MrValue;
```

See Also

```
addValue()  
removeValue()
```

AttributeValue::removeValue() - Remove Named List Attribute Value

Removes a named element from a string list `AttributeValue` object. Use `isList()` to determine if the object contains a list of values.

Syntax

```
char *removeValue(const char *value)
```

Parameters

***value** is a required input parameter that points to a null-terminated character string (a character array) and identifies an element in the list by the value of the element. This parameter must be passed a `const char *` and not a `char *` variable. Simply putting the value directly into the function in quotation marks does not work.

Returns

A pointer to the sting representing the value. This is provided as a convenience to use the value so you do not have to retrieve the value with a separate call.

Warning: To avoid a memory leak, you must delete the returned value with the `delete[]` operator within the plug-in.

Example

```
char * MrValue = someAttrValuePtr->removeValue("Apple");
//MrValue now points to "Apple" if someAttrValuePtr
//contained such value, to null if not..
//Delete value, even if null:
delete[] MrValue;
```

See Also

`addValue()`

`removeAt()`

`isList()`

AttributeValue::size() - Count Number of List Elements

This method returns a count of the total number of elements in an indexed list and returns the value 1 for single-type attributes. That is, for lists, it is equivalent to `entries()`.

Syntax

```
int entries() const
```

Parameters

None

Returns

An integer value that contains the total count of elements.

Example

```
if (FaveSongAvObj.isList())int elementCount = FaveSongAVObj.size();
```

See Also

`addValue()`

`AttributeValue::size()`

AttributeValue [] Operator - Returns the Value of an Indexed String List Element

Retrieves the value of a specific list element by integer index, use the [] overloaded operator. For example, to retrieve the third element in the Colors string list, use this syntax:

```
const char *FavoriteColor = Colors[2];
```

You can use `isList()` beforehand to determine if the object contains a list of values and `entries()` or `size()` to ensure the index is valid. This method also works with single-type object. In such a case, the index is 0. When used in this way, the method is equivalent to the `getValue()` method.

A

- access decision 1-9
- Active Directory Authentication Provider 9-6
- Active Directory Authentication provider 9-15
- Additional documentation
 - available on Internet 5-9
- Adjudication Provider
 - configuring 9-40
- adjudication provider 1-9
- Admin role 3-3
- Administration Console 4-2
 - About tab 5-2
 - Access Denied error 5-4
 - checking the version 5-2
 - close 5-7
 - closing 5-7
 - configuration 5-4
 - getting help 5-8
 - help
 - displaying 5-7
 - home page 5-7
 - logging out 5-4, 5-7
 - login name 5-6
 - Preferences tab 5-2
 - preferences, setting 5-2
 - refreshing current page 5-7
 - sign in 5-4
 - using 5-5
- Administration policy
 - security role 5-4
- Administration Server 5-3
- administration server
 - host name 5-6
 - IP address 5-6
- algorithm
 - password hashing 12-11
- Anonymous role 3-6
- application node 4-5
- ARME
 - extensions 12-7
- ASI Adjudication Provider 9-5
- ASI Authorization and ASI Role Mapping provider
 - extension 12-1
- ASI Authorization Provider 9-7
- ASI Authorization provider 9-38
- ASI Role Mapping Provider 9-7
- attribute
 - defined 4-15
 - identity 4-12
 - resource 4-6
- attribute converter 12-5, 12-6
- attribute type converter
 - use 12-7
- Audit Channel provider 13-17
- Audit Context extension 13-4
- audit events 13-3, 13-4
- Audit plug-ins 12-2, 13-4
- AuditAtnEvent 13-17
 - interface 13-17
- AuditAtzEvent 13-3, 13-18
- AuditEvent 13-3
- auditing provider 1-9
- Auditing providers 13-1
- AuditPolicyDeployEvent 13-18
- AuditPolicyEvent 13-19
- AuditPolicyUndeployEvent 13-18
- AuditPredicateEvent 13-20
- AuditRoleDeployEvent 13-20
- AuditRoleEvent 13-3, 13-19
- AuditRoleUndeployEvent 13-20
- Authenticating Console
 - users 5-4
- Authentication Provider
 - configuring 9-10
- Authentication provider
 - Open LDAP 9-13
- authentication provider 1-9
- Authentication providers
 - order 9-11
- authorization provider 1-9
- Authorization Providers 9-40

B

Banner

- current user 5-7

binding node 4-5

C

certificate authorities, trusted 7-2

certificate chain

- validating 7-15

certificates

- demo 7-1

command-line arguments

- certificate validation 7-13

configuration 4-3

- Service Control Manager 9-2

- viewing results 11-5

configuration attributes 5-6

configuration ID 9-4

configuration name 9-4

configuring security

- SSL 7-3

constant

- defined 4-15

ContextElements 13-3

Control Flag

- setting 9-12

convertFromASI method 12-7

ConvertResource 12-5

convertToASI method 12-7

credential functions 12-9

credential mapping provider 1-9

D

database authentication

- extension 12-10

Database Authentication Provider 9-6

Database Authentication provider 9-23

Database Credential Mapping provider 9-34

DBMSPlugin 12-11

- interface 12-11

delegate, rule

- rule

- delegate 4-12

Demoidentity.jks 7-8

DemoTrust.jks 7-8

Deployer role 3-5

deployment 4-1

Deployment folder 11-4, 11-5

digital certificates

- definition 7-2

- obtaining, keytool 7-4

- storing 7-4

- troubleshooting 7-16

directory server 4-9

distributed architecture 1-6

distribution results

- viewing 11-5

Domain Name Server 5-3

dynamic attributes

- custom 12-9

dynamic role association 1-9

E

enumerated type 4-15

evaluation function

- understanding 4-15

event

- audit 13-17

- AuditPolicyDeployEvent 13-18

- AuditPolicyEvent 13-19

- AuditPolicyUndeployEvent 13-18

- AuditPredicateEvent 13-20

- AuditRoleDeployEvent 13-20

- AuditRoleEvent 13-19

- AuditRoleUndeployEvent 13-20

- PolicyAdministrationEvent 13-21

Everyone role 3-6

extension

- for ASI Authorization and ASI Role Mapping provider 12-1
- extension
 - Audit Context 13-4
 - building an 12-8
 - database authentication 12-10
 - deploying 12-8
 - using 12-9

F

- filter strings
 - defining 5-2

G

- GeneralSecurityException 13-18
- getASITypeName method 12-7
- GetAttributeValue 12-5
- getAttributeValue method 12-6
- getHandledAttributeNames method 12-6
- GetHandledTypes 12-5
- getType method 12-7
- group
 - membership 4-12
- groups 4-1

H

- header files
 - used by extension 12-8
- Host Name 5-6

I

- identity
 - unique 4-10
- identity asserter 1-9
- identity attribute 4-1, 4-12
 - defined 4-12
- identity keystores
 - configuring 7-8

- creating 7-5
- ImportPrivateKey 7-7
- IP address 5-6
- iPlanet LDAP Authentication provider 9-16

J

- JAAS
 - control flag 9-12

K

- Keystore providers
 - configuring for SSL 7-11
- keystores 7-6
 - configuring 7-8
 - creating 7-5
 - JKS 7-15
 - keytool 7-5
- keytool 7-4
 - common commands 7-6
 - creating keystores 7-5
 - loading, private keys 7-6
 - obtaining, utility 7-4

L

- LDAP 5-4
- Log4j
 - logging libraries 9-5
 - renderers 13-3
- Log4J Audit Channel provider
 - custom extension 13-4
- Log4j Audit Channel Provider 9-5, 9-44, 13-3

M

- Microsoft Windows NT 5-4
- Monitor role 3-6

N

- Netscape iPlanet Authentication Provider 9-6
- Novell Authentication Provider 9-6
- Novell LDAP Authentication provider 9-20

O

- one-way SSL
 - defined 7-3
- Open LDAP 1-9
- Open LDAP Authentication Provider 9-6
- Operator role 3-5
- Oracle
 - database replication 2-7
- organizational node 4-5

P

- page size 5-3
 - default number of elements 5-3
 - limiting information displayed. 5-2
 - restricting 5-2
- page tabs
 - actions performed 5-6
- password
 - hashing algorithm 12-11
- policy
 - authorization 4-13
 - defined 4-1
 - deploy 11-5
 - distributing 11-2
- policy database 5-2
- policy domain 5-1
- policy element folders 5-6
- policy elements 4-2, 5-2
 - restoring defaults 5-3
- Policy Inquiry 3-4
- policy inquiry 4-14
 - understanding 4-14
- policy verification 4-14
 - definition 4-14

- understanding 4-14
- PolicyAdministrationEvent 13-21
- preferences
 - hiding help text 5-1
 - page size 5-1
 - saved in a cookie 5-2
 - setting 5-1
- principal validation provider 1-9
- private key files
 - configuring for SSL 7-11
- private keys 7-2
 - definition 7-2
 - obtaining 7-4
 - storing 7-4
- privilege
 - defined 4-7
- privilege group
 - defined 4-8
- privileges
 - default 3-8
- provider
 - configuring security 9-9
 - extensions 12-1
- providers 4-1
 - configuring 9-1

R

- resource 4-1, 4-12
 - attribute 4-6
 - defined 4-4
 - examples of 4-6
- resource converters 12-4
- Resource Deployment Audit Provider 9-5
- resource node 4-5
- ResourceConverter
 - interface 12-4
- resources
 - grouping of 4-5
- role
 - granting 4-12

- role mapping provider 1-9
- rule design 4-1

S

- SAML Credential Mapping Provider 9-7
- SAML Credential Mapping provider 9-37
- SAML Identity Assertion Provider 9-6
- SAML Identity Assertion provider 9-29
- Secure Socket Layers 5-3
- Secure Sockets Layer
 - production environment 7-1
- Secure Sockets Layer (SSL)
 - configuring two-way SSL 7-11
 - digital certificate 7-2
 - identity 7-2, 7-3
 - one-way SSL 7-3
 - private keys 7-2
 - storing, private keys 7-4
 - storing,digital certificates 7-4
 - storing,trusted CA certificates 7-4
 - two-way SSL 7-3
- security
 - configuration 4-1, 4-2, 9-1
- security configuration 9-1
- Security Configuration folder 5-6
- security provider
 - configuring 9-9
- security providers
 - defining 5-6
 - representation 4-3
- security roles 3-2
- Security Service Module 4-1, 4-2
 - binding 9-8
 - configuring 9-4
 - unbinding 9-8
 - understanding 9-3
- Security Service Modules 1-5
- security, managing 4-1
- server
 - identity 7-2, 7-3

- Service Control Manager 4-1, 4-2, 4-3
 - binding 9-8
 - unbinding 9-8
 - understanding 9-2

- Single 9-30

- Single Pass Negotiate Identity Asserter 9-6

- Single Pass Negotiate Identity Assertion provider 9-30

- SPNEGO
 - protocol 9-30

T

- Transport Layer Security 7-1
- Trust keystores
 - configuring 7-8
 - creating 7-5
- trusted CA certificates
 - storing 7-4
- two-way SSL
 - configuring 7-11
 - defined 7-3

U

- understanding evaluation functions 4-15
- understanding policy rule 4-14
- understanding policy verification 4-14
- URL
 - Administration Console 5-3
- user
 - defined 4-10
- users 4-1

V

- ValidateCertChain
 - description 7-15

W

- WebLogic 9-7

- WebLogic Authentication Provider 9-5, 9-48
- WebLogic Authorization Provider 9-6, 9-48
- WebLogic Credential Mapping Provider 9-7, 9-48
- WebLogic Credential Mapping provider
 - configuring 9-51
- WebLogic Portal 9-7
- WebLogic Role Mapping Provider 9-7, 9-48
- WebLogic Server Security Providers 9-48
- wildcards 5-2
- Windows NT Authentication Provider 9-6
- Windows NT Authentication provider 9-14
- writeEvent 13-1

X

- X.509 Identity Assertion Provider 9-6
- X.509 Identity Assertion provider 9-31