



# BEA AquaLogic Enterprise Security™®

## Integrating ALES with Application Environments





# Copyright

Copyright © 1995-2006 BEA Systems, Inc. All Rights Reserved.

## Restricted Rights Legend

This software is protected by copyright, and may be protected by patent laws. No copying or other use of this software is permitted unless you have entered into a license agreement with BEA authorizing such use. This document is protected by copyright and may not be copied photocopied, reproduced, translated, or reduced to any electronic medium or machine readable form, in whole or in part, without prior consent, in writing, from BEA Systems, Inc.

Information in this document is subject to change without notice and does not represent a commitment on the part of BEA Systems. THE DOCUMENTATION IS 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 DOCUMENT IN TERMS OF CORRECTNESS, ACCURACY, RELIABILITY, OR OTHERWISE.

## Trademarks and Service Marks

Copyright © 1995-2006 BEA Systems, Inc. All Rights Reserved. BEA, BEA JRocket, BEA WebLogic Portal, BEA WebLogic Server, BEA WebLogic Workshop, 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 Interaction, BEA AquaLogic Interaction Analytics, BEA AquaLogic Interaction Collaboration, BEA AquaLogic Interaction Content Services, BEA AquaLogic Interaction Data Services, BEA AquaLogic Interaction Integration Services, BEA AquaLogic Interaction Process, BEA AquaLogic Interaction Publisher, BEA AquaLogic Interaction Studio, BEA AquaLogic Service Bus, BEA AquaLogic Service Registry, BEA Builder, BEA Campaign Manager for WebLogic, BEA eLink, BEA Kodo, BEA Liquid Data for WebLogic, BEA Manager, BEA MessageQ, BEA SALT, BEA Service Architecture Leveraging Tuxedo, BEA WebLogic Commerce Server, BEA WebLogic Communications Platform, 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 Log Central, BEA WebLogic Mobility Server, BEA WebLogic Network Gatekeeper, BEA WebLogic Personalization Server, BEA WebLogic Personal Messaging API, BEA WebLogic Platform, BEA WebLogic Portlets for Groupware Integration, BEA WebLogic Real Time, BEA WebLogic RFID Compliance Express, BEA WebLogic RFID Edge Server, BEA WebLogic RFID Enterprise Server, BEA WebLogic Server Process Edition, BEA WebLogic SIP Server, BEA WebLogic WorkGroup Edition, BEA Workshop for WebLogic Platform, BEA Workshop JSP, BEA Workshop JSP Editor, BEA Workshop Struts, BEA Workshop Studio, Dev2Dev, Liquid Computing, and Think Liquid are trademarks of BEA Systems, Inc. Accelerated Knowledge Transfer, AKT, BEA Mission Critical Support, BEA Mission Critical Support Continuum, and BEA SOA Self Assessment are service marks of BEA Systems, Inc.

All other names and marks are property of their respective owners.

# 1. Introduction

Document Scope and Audience . . . . .	1-1
Guide to this Document . . . . .	1-2
Related Documentation. . . . .	1-3
Contact Us! . . . . .	1-4

# 2. Securing ALES Components

Using the Administration Console . . . . .	2-1
Default Database Objects . . . . .	2-2
Creating a New Admin User. . . . .	2-2
ALES Resources . . . . .	2-3
Administrative Operations. . . . .	2-4
Privileges . . . . .	2-5
Context Attributes . . . . .	2-6
Evaluation Functions . . . . .	2-8
Authorization Queries . . . . .	2-9
Enumerated Types . . . . .	2-16
ALES Identities. . . . .	2-17
Default Role Mapping Policies. . . . .	2-17
Default Authorization Policies . . . . .	2-18
Viewing Authorization Policies . . . . .	2-20

# 3. Setting Up Application Security Administrators

Overview. . . . .	3-1
Establishing a Resource Parent for the Application . . . . .	3-2
Create Administrative Users. . . . .	3-2
Identity Directories . . . . .	3-2
Users and Groups. . . . .	3-2
Policies for Application-Level Administration. . . . .	3-3

# 4. Integrating ALES with Applications

Overview. . . . .	4-1
Security Service Modules . . . . .	4-2
SSM Security Providers . . . . .	4-3
Integrating ALES with Other BEA Applications . . . . .	4-5

## 5. Configuring the Web Server SSM

Understanding the Web Server SSMs .....	5-1
Web Server SSM Overview .....	5-2
Web Server Environmental Binding .....	5-3
Web Server SSM Features .....	5-4
Web Single Sign-on Capabilities .....	5-4
Authentication Service Features .....	5-6
Authorization Service Features .....	5-7
Auditing Service Features .....	5-8
Role Mapping Features .....	5-8
Credential Mapping Features .....	5-8
Administration Features .....	5-9
Session Management Features .....	5-9
Configuration Features .....	5-10
Web Server Constraints and Limitations .....	5-10
Web Server SSM Integration Tasks .....	5-11
Configuring and Deploying Policy for the Web Server SSM .....	5-11
Creating Resources .....	5-12
Creating Policies .....	5-14
Modifying Admin and Everyone Role Mapping Policies .....	5-15
Configuring the Application Deployment Parent .....	5-15
Configuring the ALES Identity Assertion and Credential Mapping Providers .....	5-16
Distributing Policy and Security Configuration .....	5-16
Configuring the Web Server Environmental Binding .....	5-17
Configuring the Environmental Binding for the Microsoft IIS Web Server .....	5-17
Configuring the Microsoft IIS Web Server Binding Plug-In File .....	5-17
Configuring the NamePasswordForm.acc File for the IIS Web Server .....	5-22
Deploying and Testing the IIS Web Server Sample Application .....	5-22
Configuring the Environmental Binding for the Apache Web Server .....	5-23
Downloading and Installing the Apache Web Server .....	5-24
Configuring the ALES Module .....	5-24
Configuring the NamePasswordForm.html File for the Apache Web Server .....	5-25
Deploying and Testing the Apache Web Server Sample Application .....	5-25
Configuring Web Single Sign-on with ALES Identity Assertion .....	5-26
Configuring Web Server SSMs to Web Server SSMs for SSO .....	5-27
Configuring Web Server SSMs to WebLogic Server SSMs for SSO .....	5-27
Configuring Web Server SSM Properties .....	5-28

Session Settings . . . . .	5-28
Authentication Settings . . . . .	5-29
Mapping JAAS Callback Type to Form and Form Fields . . . . .	5-31
Role Mapping Settings . . . . .	5-34
Credential Mapping Settings . . . . .	5-35
Naming Authority Settings . . . . .	5-36
Logging Level Setting . . . . .	5-37
Environment Variables Accessible Using CGI . . . . .	5-37

## 6. Configuring the Web Services SSM

Overview of the Web Services SSM . . . . .	6-1
Web Services Security Service APIs . . . . .	6-2
Authentication Service . . . . .	6-3
Authorization Service . . . . .	6-3
Auditing Service . . . . .	6-3
Role Mapping Service . . . . .	6-4
Credential Service . . . . .	6-4
Configuring and Deploying Policy for the Web Services SSM . . . . .	6-4
Binding the Web Services SSM to a Web Services Client . . . . .	6-4
Configuring SSL in the Web Services SSM . . . . .	6-4
Configuring One-Way SSL . . . . .	6-5
Configuring Two-Way SSL . . . . .	6-6
Configuring a WS-SSM for Two-Way SSL . . . . .	6-6
Configuring a Web Services Client for Two-Way SSL . . . . .	6-7
Adding New Identity Assertion Types . . . . .	6-9

## 7. Configuring the WebLogic Server 8.1 SSM

Location of the WebLogic Server Domain . . . . .	7-1
Modifying the startWebLogic File . . . . .	7-2
Defining Security Properties . . . . .	7-4
Starting and Stopping Processes . . . . .	7-5
Additional Post-Installation Considerations . . . . .	7-5
Setting the Boot Login for WebLogic Server . . . . .	7-5
Creating a WebLogic Boot Policy . . . . .	7-6
Creating the User Identity . . . . .	7-6
Creating Resources for WebLogic Server . . . . .	7-7
Grant Server Resource to Admin Role . . . . .	7-7
Grant Admin Role to WebLogic User/Group . . . . .	7-8

Binding the Resource to the ASI Authorization Provider . . . . .	7-8
Distributing the Policies to the Security Service Module . . . . .	7-9
Creating a WebLogic Console Policy . . . . .	7-9
Protecting Resources . . . . .	7-11
Protecting a Cluster of WebLogic Servers . . . . .	7-11
Security Configuration . . . . .	7-11
Resource Configuration . . . . .	7-13
Policy Configuration . . . . .	7-14

## 8. Configuring the WebLogic Server 9.x SSM

Overview of the WebLogic Server 9.x SSM . . . . .	8-1
Configuring the WebLogic Server 9.x SSM: Main Steps . . . . .	8-2
Configuring Security Providers in the WebLogic Server 9.x SSM . . . . .	8-3
Console Extension for Security Providers in the WLS 9.x Console . . . . .	8-4
Configuring a WLS 9.x Security Realm for ALES . . . . .	8-4
Using the WebLogic Server Console to Configure Security Providers . . . . .	8-4
Modifying the startWebLogic File . . . . .	8-7

## 9. Integrating with WebLogic Portal

Introduction . . . . .	9-1
Integration Features . . . . .	9-3
Supported Use-case Scenario . . . . .	9-3
Constraints and Limitations . . . . .	9-4
Integration Pre-Requisites . . . . .	9-5
Integrating with WebLogic Portal 9.2: Main Steps . . . . .	9-5
Creating the Portal Application Security Configuration . . . . .	9-6
Using the WebLogic Server Console to Configure Security Providers . . . . .	9-6
Modifying the Portal Server startWeblogic File . . . . .	9-6
Integrating with WebLogic Portal 8.1: Main Steps . . . . .	9-8
Creating the Portal Application Security Configuration . . . . .	9-9
Binding the Security Configuration . . . . .	9-10
Distributing the Security Configuration . . . . .	9-10
Creating an Instance of the Security Service Module . . . . .	9-10
Enrolling the Instance of the Security Service Module . . . . .	9-10
Modifying the Portal Server startWeblogic File . . . . .	9-11
Creating the security.properties File . . . . .	9-12
Replacing the Portal p13n_ejb.jar File . . . . .	9-12
Replacing the Portal p13n_system.jar File . . . . .	9-13

Replacing the DefaultAuthorizerInit.Idift File . . . . .	9-14
Configuring Policy for the Portal Application. . . . .	9-14
Creating the Identity Directory and Users. . . . .	9-15
Configuring Resources and Privilege . . . . .	9-16
Creating the Realm Resource. . . . .	9-16
Creating the Shared Resources. . . . .	9-17
Creating the Console Resources. . . . .	9-18
Creating the PortalApp Resources . . . . .	9-19
Creating the Role Mapping Policy . . . . .	9-20
Creating Authorization Policies. . . . .	9-21
Policy for Visitor Entitlements to Portal Resources . . . . .	9-25
Configuring Policy for Desktops . . . . .	9-26
Configuring Policy for Books . . . . .	9-27
Configuring Policy for Pages . . . . .	9-28
Configuring Policy for Portlets . . . . .	9-28
Configuring Policy for Look and Feels . . . . .	9-29
Defining Policy for Portlets using Instance ID. . . . .	9-29
Discovering Portal Application Resources . . . . .	9-30
Distributing Policy and Security Configuration . . . . .	9-30
Starting the WebLogic Portal Server . . . . .	9-30
Configuring Portal Administration to Use the WebLogic Authenticator . . . . .	9-31
Using Portal Administration Tools to Create a Portal Desktop. . . . .	9-31
Accessing the Portal Application. . . . .	9-32

## 10. Integrating with AquaLogic Data Services Platform

Introduction. . . . .	10-1
Integration Features . . . . .	10-3
Supported Use-case Scenario . . . . .	10-3
Constraints and Limitations. . . . .	10-3
Integration Pre-Requisites. . . . .	10-4
Integrating with AquaLogic Data Services Platform: Main Steps . . . . .	10-4
Enabling Elements for Access Control. . . . .	10-5
Creating the WebLogic Server SSM Configuration. . . . .	10-6
Binding the SSM Configuration . . . . .	10-7
Distributing the SSM Configuration . . . . .	10-7
Creating an Instance of the Security Service Module . . . . .	10-7
Enrolling the Instance of the Security Service Module. . . . .	10-7
Creating the WebLogic Server startWebLogicALES File. . . . .	10-7

Creating the security.properties File . . . . .	10-8
Configuring Policy for Data Services. . . . .	10-8
Creating the Identity Directory and Users . . . . .	10-9
Configuring Resources and Privilege . . . . .	10-9
Creating the RTLApp Application Resources . . . . .	10-10
Creating the ALDSP Resources. . . . .	10-10
Creating the Role Mapping Policies . . . . .	10-12
Creating Authorization Policies . . . . .	10-13
Discovering Data Services . . . . .	10-16
Distributing Policy and SSM Configuration . . . . .	10-16
Starting the WebLogic Server . . . . .	10-16
Accessing the ALDSP Application. . . . .	10-17

## 11. Integrating with AquaLogic Service Bus

Introduction . . . . .	11-1
Integration Pre-Requisites . . . . .	11-2
Integrating with AquaLogic Service Bus: Main Steps . . . . .	11-2
Enabling Elements for Access Control. . . . .	11-3
Creating the WebLogic Server SSM Configuration. . . . .	11-3
Creating an Instance of the Security Service Module. . . . .	11-3
Enrolling the Instance of the Security Service Module. . . . .	11-3
Configuring Policy for ALSB Resources. . . . .	11-4
Configuring Resources and Privileges . . . . .	11-4
Creating the ALSB Application Resources . . . . .	11-4
Creating the ALSB Proxy Service Resources . . . . .	11-5
Discovering Services. . . . .	11-7

## 12. Enabling SAML-based Single Sign-On

Overview . . . . .	12-1
Configuring ALES as a SAML Assertion Consumer. . . . .	12-2
Configuring ALES as a SAML Assertion Producer. . . . .	12-3

## 13. Enabling SPNEGO-based Single Sign-on

Configuring Single Sign-On with Microsoft Clients . . . . .	13-1
Requirements. . . . .	13-2
Enabling a Web Service or Web Application . . . . .	13-3
Configuring the SPNEGO Security Provider. . . . .	13-3

Editing the Descriptor File . . . . .	13-3
Configuring Active Directory Authentication . . . . .	13-5
Utility Requirements . . . . .	13-5
Configuring and Verifying Active Directive Authentication. . . . .	13-5
Configure the Active Directory Authentication Provider . . . . .	13-7
Configure the Client .NET Web Service . . . . .	13-7
Configure the Internet Explorer Client Browser . . . . .	13-8
Configure the Sites . . . . .	13-8
Configure Intranet Authentication . . . . .	13-8
Verify the Proxy Settings . . . . .	13-9
Set the Internet Explorer 6.0 Configuration Settings . . . . .	13-9

## 14.Authorization Caching

Understanding Authorization Caching . . . . .	14-1
Configuring Authorization Caching . . . . .	14-2
Authorization Caching Expiration Functions . . . . .	14-3



# Introduction

This section describes the contents and organization of this guide—*Integrating ALES with Application Environments*. It includes the following topics:

- [“Document Scope and Audience”](#) on page 1-1
- [“Guide to this Document”](#) on page 1-2
- [“Related Documentation”](#) on page 1-3
- [“Contact Us!”](#) on page 1-4

## Document Scope and Audience

This document is a resource for system administrators and database administrators who administer and deploy BEA AquaLogic Enterprise Security™. It is primarily intended for Application Security Administrators who are responsible for configuring the ALES components, integrating ALES into application environments, managing interaction between an applications and ALES, and setting up application-level security administrators.

The topics in this document are relevant during the staging, production deployment, and production use phases of a software project. For links to other AquaLogic Enterprise Security documentation and resources, see [“Related Documentation”](#) on page 1-3.

It is assumed that readers understand Web technologies and have a general understanding of the Microsoft Windows or UNIX operating system being used. Prior to using this document, you should have a general understanding of the principal components and architecture of BEA AquaLogic Enterprise Security. Read the [Introduction to BEA AquaLogic Enterprise Security](#) for

conceptual information that is helpful in understanding how the product works. This document provides information for post-installation configuration and operation of AquaLogic Enterprise Security; read *Installing the Administration Server* and *Installing Security Service Modules* for information about installation procedures that you need to perform first.

Additionally, BEA AquaLogic Enterprise Security includes many terms and concepts that you need to understand. These terms and concepts, which you will encounter throughout the documentation, are defined in the *Glossary*.

## Guide to this Document

This document describes tasks associated with configuring and deploying AquaLogic Enterprise Security. After you have installed the ALES Administration Server (as described in *Installing the Administration Server*) and any ALES SSMs (as described in *Installing Security Service Modules*), you need to configure the SSMs to integrate them with the applications they secure. This document is organized as follows:

- [Chapter 2, “Securing ALES Components,”](#) describes how to control access to ALES using the Administration Console.
- [Chapter 3, “Setting Up Application Security Administrators,”](#) describes how to establish application-security administrator users using the Administration Console.
- [Chapter 4, “Integrating ALES with Applications,”](#) describes how to configure SSMs and bind them to application servers.
- [Chapter 5, “Configuring the Web Server SSM,”](#) describes how to configure the Web Server SSM.
- [Chapter 6, “Configuring the Web Services SSM,”](#) describes how to configure the Web Services SSM.
- [Chapter 7, “Configuring the WebLogic Server 8.1 SSM,”](#)
- [Chapter 8, “Configuring the WebLogic Server 9.x SSM,”](#)
- [Chapter 9, “Integrating with WebLogic Portal,”](#)
- [Chapter 10, “Integrating with AquaLogic Data Services Platform,”](#)
- [Chapter 11, “Integrating with AquaLogic Service Bus,”](#)
- [Chapter 12, “Enabling SAML-based Single Sign-On,”](#) describes how to configure SSMs to provide SSO using the SAML 1.1 Browser POST Profile.

- [Chapter 13, “Enabling SPNEGO-based 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 14, “Authorization Caching,”](#) describes how to configure and manage authorization caching to enhance performance when implementing authorization policies.

## Related Documentation

For information about other aspects of AquaLogic Enterprise Security, see the following documents:

- [Introduction to BEA AquaLogic Enterprise Security](#)—This document provides overview, conceptual, and architectural information for AquaLogic Enterprise Security.
- [Installing the Administration Server](#)—This document describes installing and configuring the AquaLogic Enterprise Security Administration Application.
- [Installing Security Service Modules](#)—This document describes installing and configuring Security Service Modules for AquaLogic Enterprise Security.
- [Administration and Deployment Guide](#)—This document provides an architectural overview of the product and includes step-by-step instructions on how to perform various post-installation administrative tasks.
- [Policy Managers Guide](#)—This document defines the policy model used by BEA AquaLogic Enterprise Security, and describes how to generate, import and export policy data.
- [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.
- [Programming Security for Web Services](#)—This document describes how to implement security in web servers. It includes descriptions of the Web Services Application Programming Interfaces.
- [Developing Security Providers for BEA AquaLogic Enterprise Security](#)—This document provides security vendors and security and application developers with the information needed to develop custom security providers.
- [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 AquaLogic Enterprise Security.

- *Wsdl docs for Web Services API*—This document provides reference documentation for the Web Services Application Programming Interfaces that are provided with and supported by this release of BEA AquaLogic 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 AquaLogic Enterprise Security.
- *Javadocs for BLM API*—This document provides reference documentation for the Business Logic Manager (BLM) Application Programming Interfaces that are provided with and supported by this release of BEA AquaLogic Enterprise Security.

## Contact Us!

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

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

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

- Your name, e-mail address, phone number, and fax number
- Your company name and company address
- Your machine type and authorization codes
- The name and version of the product you are using

A description of the problem and the content of pertinent error messages.

# Securing ALES Components

AquaLogic Enterprise Security is itself secured using the same policy model used to secure any other application. This chapter explains the default policies controlling administrative access to ALES.

Information is provided in the following sections.

- “Using the Administration Console” on page 2-1
- “Default Database Objects” on page 2-2
- “ALES Resources” on page 2-3
- “ALES Identities” on page 2-17
- “Default Role Mapping Policies” on page 2-17
- “Default Authorization Policies” on page 2-18
- “Viewing Authorization Policies” on page 2-20

## Using the Administration Console

Many of the tasks described in the document are performed using the ALES Administration Console. For more information about using the Administration Console, see [Using the Administration Console](#) in the *Administration and Deployment Guide* and also consult the ALES Administration Console online help system.

# Default Database Objects

Installing ALES provides a number of database objects that collectively define access to ALES components. This provides rudimentary security at startup; you can use the Administration Console to more completely define administrative access.

The default database objects are listed below and are more fully described in sections that follow.

**Table 2-1 Default Database Objects Defining Access to ALES**

Object Type	Description
Resource	A representation of ALES components is defined in a separate tree under a root resource named ASI. Policies can be assigned to a resource representing an ALES component and thereby define access to that component. For more information, see <a href="#">“ALES Resources” on page 2-3</a> .
Identity	A number of users, groups, and roles that reflect usage of ALES are provided. In particular, a user named <code>system</code> is set up as having complete administrative rights to the database. For more information, see <a href="#">“ALES Identities” on page 2-17</a> .
Role Mapping Policies	A number of role mapping policies are provided that assign some of the default roles to users/groups. For more information, see <a href="#">“Default Role Mapping Policies” on page 2-17</a> .
Authorization Policies	A number of authorization policies are provided that assign privileges to roles/groups/users on specific resources in the ASI resource tree. For more information, see <a href="#">“Default Authorization Policies” on page 2-18</a> .

## Creating a New Admin User

By default, ALES provides a single administrative user identity named `system` having complete administrative rights. In a production environment, you should remove this administrative user and replace it with one or more other user identities. This section describes how to create a new administrative user named `myadmin`, replacing the `system` user:

1. In the ALES Administration Console, navigate to Identities > Users. Add a new user named `myadmin`.
2. Add the `myadmin` user to the Admin role.
3. Set a password for `myadmin` by selecting `myadmin` and clicking Edit > Set password.
4. Remove the `system` user from the Admin role.

5. Distribute policy
6. Stop the Administration Server.
7. Edit `BEA_HOME/ales22-admin/config/WLESWebLogic.conf` so that under “Java Additional Parameters” this line reads:
 

```
wrapper.java.additional.15=-Dwles.user.alias=myadmin
```

 instead of
 

```
wrapper.java.additional.15=-Dwles.user.alias=system
```
8. Set the password for the `myadmin` user, using the `asipassword` utility. Execute:
 

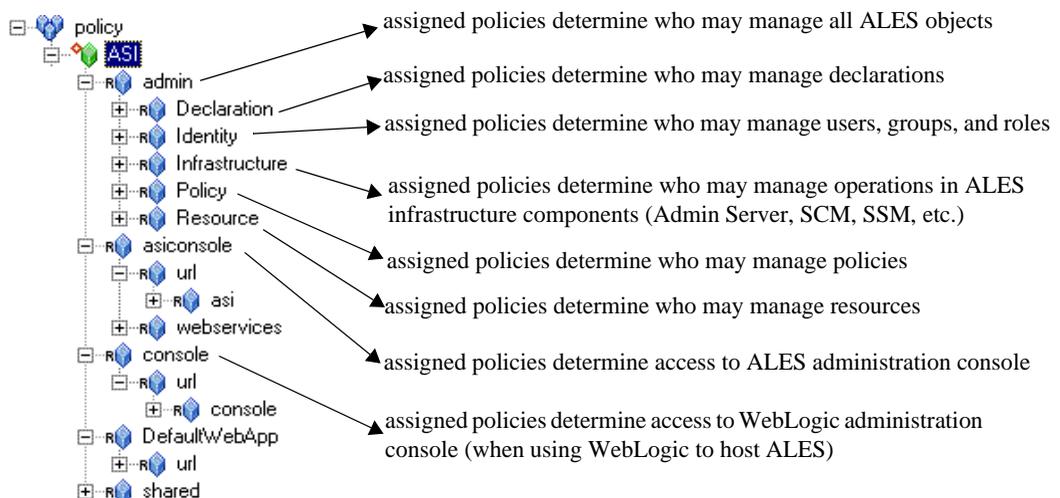
```
BEA_HOME/ales22-admin/bin/asipassword.bat myadmin ../ssl/password.xml  
../ssl/password.key
```

 and supply the password for `myadmin`.
9. Restart the ALES Administration Server with `WLESWebLogic.bat console`. You can now log in as `myadmin` with the password you set.

## ALES Resources

ALES components are represented under the ASI resource tree, as shown in the figure below.

**Figure 2-1 Representation of ALES Components**



# Administrative Operations

Table 2-2 describes resource objects that define the administrative operations that are performed using the Administration Console. By default, these resources are contained within `//app/policy/ASI/admin`.

**Table 2-2 Resources Defining Administrative Operations**

Resource Name	Protects operations on...
admin/Declaration/Attribute	attribute declarations
admin/Declaration/Constant	constant declarations
admin/Declaration/Enumeration	enumeration declarations
admin/Declaration/EvaluationFunction	evaluation function declarations
admin/Identity/Directory/Instance	identity directory instances
admin/Identity/Directory/AttributeMapping/Single	what scalar attributes may be assigned to users within a directory
admin/Identity/Directory/AttributeMapping/List	what vector attributes may be assigned to users within a directory
admin/Identity/Subject/User	users
admin/Identity/Subject/Group	groups
admin/Identity/Subject/Password	user passwords
admin/Identity/Subject/AttributeAssignment/Single	scalar subject attribute values
admin/Identity/Subject/AttributeAssignment/List	vector subject attribute values
admin/Resource/Instance	resources
admin/Resource/AttributeAssignment/Single	scalar resource attribute values
admin/Resource/AttributeAssignment/List	vector resource attribute values
admin/Resource/MetaData/LogicalName	setting the "logical name" resource metadata
admin/Resource/MetaData/IsApplication	setting the "is application" resource metadata
admin/Resource/MetaData/IsDistributionPoint	setting the "is distribution point" metadata
admin/Policy/Grant	grant policies
admin/Policy/Deny	deny policies
admin/Policy/Delegate	delegate policies
admin/Policy/Action/Role/Instance	roles (when used as actions)
admin/Policy/Action/Privilege/Instance	privileges
admin/Policy/Action/Privilege/Group	privilege groups

**Table 2-2 Resources Defining Administrative Operations**

Resource Name	Protects operations on...
admin/Policy/Analysis/InquiryQuery	policy inquiries
admin/Policy/Analysis/VerificationQuery	policy verification
admin/Infrastructure/Engines/ARME	definitions of the Authorization and Role Mapping Engine (ARME)
admin/Infrastructure/Engines/SCM	definitions of the Service Control Manager (SCM)
admin/Infrastructure/Management/BulkManager	the policy loader
admin/Policy/Repository	the policy repository

## Privileges

Table 2-3 lists and describes the default privileges that may be assigned.

**Table 2-3 Privileges**

Privilege	Explanation
create	Create a policy element, including identities (identity 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.

**Table 2-3 Privileges (Continued)**

Privilege	Explanation
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.
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

Context attributes can be used to provide fine-grained protection of policy operations. For example, when creating a privilege, the name of the privilege can be supplied as an attribute and used to control access to a single unique privilege.

[Table 2-4](#) describes the default context attributes.

**Table 2-4 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.
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.

**Table 2-4 Context Attributes (Continued)**

<b>Attribute Name</b>	<b>Data Type</b>	<b>Description</b>
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.
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 policy; this is the portion between the 'if' and ';' exclusive.
new_action	string	Name of new action in a modified policy.
new_action_type	string	New action type in a modified policy.
new_resource	string	New resource in a modified policy.
new_subject_name	string	New subject name.
new_constraint	string	New constraint in a modified policy.
delegator	string	The name of the delegator in a policy.
new_delegator	string	New delegator in a modified policy.
actions	list of strings	A set of actions.

**Table 2-4 Context Attributes (Continued)**

Attribute Name	Data Type	Description
action_groups	list of strings	A list of privilege group names.
action_group	string	The name of a privilege group.
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. <sup>1</sup>
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 role mapping and authorization policy effect.
title	string	The title of a analysis query.

1. The term engine refers to an ASI Authorization provider and ASI Role Mapper provider that are configured to operate in conjunction with one another, also referred to as the ARME. This combination of providers are configured to manage your authorization and role mapping policies.

## Evaluation Functions

The evaluation functions listed in [Table 2-5](#) are provided for writing custom administration policies. They may be used in the constraint portion of policies to limit the applicability of the policy based on contextual information.

**Table 2-5 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.

**Table 2-5 Evaluation Functions**

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

Table 2-6 describes when contextual data is used to define administrative access. This data that may be referenced when writing policies to protect the administration console.

**Table 2-6 Context Attributes and Administrative Access**

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.
	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.

**Table 2-6 Context Attributes and Administrative Access (Continued)**

<b>Admin Resource</b>	<b>Privilege</b>	<b>Context attributes</b>	<b>Description</b>
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.
	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.

**Table 2-6 Context Attributes and Administrative Access (Continued)**

Admin Resource	Privilege	Context attributes	Description
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 policies 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.
	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/AttributeAssignment/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.

**Table 2-6 Context Attributes and Administrative Access (Continued)**

Admin Resource	Privilege	Context attributes	Description
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.
	cascade Delete	resource	Queried when user attempts to cascade delete a resource. This includes deletion of all child resources and associated policies.
	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.
	rename	logical_name, resource, new_name	Queried when user attempts to rename a logical name for a resource.

**Table 2-6 Context Attributes and Administrative Access (Continued)**

<b>Admin Resource</b>	<b>Privilege</b>	<b>Context attributes</b>	<b>Description</b>
Policy/Grant	create	action, resource, subject_name, constraint	Queried when user attempts to create a new grant policy. “action”, “resource”, and “subject_name” attributes are lists.
	delete	action, resource, subject_name, constraint	Queried when user attempts to delete a grant policy. 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 policies “action”, “resource”, and “subject_name” attributes are lists.
Policy/Deny	create	action, resource, subject_name, constraint	Queried when user attempts to create a new deny policy. “action”, “resource”, and “subject_name” attributes are lists.
	delete	action, resource, subject_name, constraint	Queried when user attempts to delete a deny policy. 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 policy. The “action”, “resource”, and “subject_name” attributes are lists.
Policy/Delegate	create	action, resource, subject_name, delegator, constraint	Queried when user attempts to create a new delegate policy. “action”, “resource”, and “subject_name” attributes are lists.
	delete	action, resource, subject_name, delegator, constraint	Queried when user attempts to delete a delegate policy. 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 policy. The “action”, “resource”, and “subject_name” attributes are lists.

**Table 2-6 Context Attributes and Administrative Access (Continued)**

<b>Admin Resource</b>	<b>Privilege</b>	<b>Context attributes</b>	<b>Description</b>
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.
	removeMember	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 empty string.

**Table 2-6 Context Attributes and Administrative Access (Continued)**

<b>Admin Resource</b>	<b>Privilege</b>	<b>Context attributes</b>	<b>Description</b>
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 empty string.
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.
	deploy Structural Change	deleted_directories, deployed_engines, deleted_engines, deleted_bindings, deleted_applications	Queried when user attempts to deploy a structural change.
Infrastructure/Engines/ ARME	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.

**Table 2-6 Context Attributes and Administrative Access (Continued)**

Admin Resource	Privilege	Context attributes	Description
Infrastructure/Engines/SCM	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.
Infrastructure/Management/Console	login		Queried when user attempts to login to the Administration Console.
Infrastructure/Management/BulkManager	login		Queried when user attempts to login to the Policy Import tool.

## Enumerated Types

Table 2-7 lists the name of each enumerated type used in controlling administrative access.

**Table 2-7 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_node, binding_node, resource_node)	Specifies the valid resource types.
effect_type_enum	(grant_effect, deny_effect, delegate_effect)	Specifies the valid role mapping and authorization effect types.

# ALES Identities

[Table 2-8](#) shows the default ALES roles, users, and groups and some of their administrative rights as determined by existing policies.

**Table 2-8 Default ALES Role Privileges and Identities**

Role	Privileges / Resources	User/ Groups
Admin	Has all privileges, including creating and managing resources, identities, configurations, starting/stopping ALES servers, etc.	System (User)
Deployer	Privileges include modifying SCM/SSM configurations, deploying configuration and policy data, and running policy inquiries.	None
Operator	Privileges include managing SCM/SSM configurations, starting /stopping Administration Server, and running policy inquiries.	None
Monitor	This role effectively provides read-only access to the Administration Console. Privileges include monitoring Administration Console activities and viewing SCM/SSM configurations.	None
Everyone	Change password, access the Console login page, access unprotected resources and operations	Allusers(Group)
Anonymous	No privileges. Does not allow access to ASI resources. This role is automatically assigned to all unauthenticated users.	Anonymous(User) Allusers(Group)

## Default Role Mapping Policies

The default role mapping policies are described in [Table 2-9](#) below. There are two ways they can be viewed in the Administration Console:

- To see role mapping policies assigned to a specific ALES resource, navigate to and select the resource in the ASI resource tree. Then click Role Mapping Policy Inquiry in the lower right page.
- To see role mapping policies assigned to a specific ALES role, expand the Identity node and select the Role node. Then select the role in the right page and click Role Mapping Policy Inquiry.
- To see all role mapping policies, expand the Policy node in the navigation tree and select Role Mapping Policies.

Of particular note, one of the role mapping policies assigns the Admin role to the user named System. This is the only administrative user provided when ALES is installed.

**Table 2-9 Default Role Mapping Policies**

Policy	Description
grant(/role/Everyone, //app/policy/ASI, //sgrp/asi/allusers/) if true;	Assigns Everyone role to allusers (group).
grant(/role/Admin, //app/policy/ASI, //user/asi/system/) if true;	Assigns Admin role to system (user).
grant(/role/Anonymous, //app/policy/ASI, //user/asi/anonymous/);	Assigns Anonymous role to anonymous (user)

## Default Authorization Policies

A number of authorization policies are provided that define access to ALES components. Some of the more important default authorization policies are described in [Table 2-10](#) below.

**Table 2-10 Default Authorization Policies**

Default Policy	Description
grant(/priv/delete, //app/policy/ASI/admin, //role/Admin) if true;	Allows Admin role to delete policies.
grant(/priv/cascadeDelete, //app/policy/ASI/admin, //role/Admin) if true;	Allows Admin role to perform cascadeDelete on children of ASI/admin.
grant(/priv/rename, //app/policy/ASI/admin, //role/Admin) if true;	Allows Admin role to rename children of ASI/admin.
grant(/priv/deployStructuralChange, //app/policy/ASI/admin/Policy/Repository, //role/Admin) if true;	Allows Admin role to deploy structural changes.
grant(/priv/login, //app/policy/ASI/admin/Infrastructure/Management/BulkManager, //role/Admin) if true;	Allows Admin role to use the policy loader tool.
grant(/priv/copy, //app/policy/ASI/admin/Identity/Subject/User, //role/Admin) if true;	Allows Admin role to copy users.
grant([/priv/bind,/priv/unbind], //app/policy/ASI/admin/Infrastructure/Engines, //role/Admin) if true;	Allows Admin role to bind/unbind resources, and configure authorization and role mapping provider combinations and SCMs.

**Table 2-10 Default Authorization Policies (Continued)**

Default Policy	Description
grant(/priv/deployUpdate, //app/policy/ASI/admin/Policy/Repository, [/role/Admin,/role/Deployer]) if true;	Allows Admin and Deployer roles to deploy policy updates.
grant(/priv/modify, //app/policy/ASI/admin, [/role/Admin,/role/Deployer]) if true;	Allows Admin and Deployer roles to children of ASI/admin (resources, identities, policies, etc.)
grant(/priv/view, //app/policy/ASI/admin, [/role/Admin,/role/Monitor,/role/Operator,/role/Deployer]) if true;	Allows Admin, Monitor, Operator, and Deployer roles to view children of ASI/admin.
grant(/priv/listAll, //app/policy/ASI/admin, [/role/Admin,/role/Monitor,/role/Operator,/role/Deployer]) if true;	Allows Admin, Monitor, Operator, and Deployer roles to perform the listAll on children of ASI/admin.
grant (/priv/modify, //app/policy/ASI/admin/Identity/Subject/ Password, //role/Everyone) if subject_name = sys_user_q;	Allows Everyone to modify their own password.
grant(/priv/create, [/app/policy/ASI/admin/Declaration, //app/policy/ASI/admin/Identity, //app/policy/ASI/admin/Infrastructure, //app/policy/ASI/admin/Resource], //role/Admin) if true;  grant(/priv/create, [/app/policy/ASI/admin/Policy/Action, //app/policy/ASI/admin/Policy/Analysis, //app/policy/ASI/admin/Policy/Rule/Delegate, //app/policy/ASI/admin/Policy/Rule/Grant], //role/Admin) if true;	Allows Admin role to create policies.
grant([/priv/create,/priv/modify, //priv/view], //app/policy/ASI/admin/Policy/Analysis, [/role/Admin,/role/Monitor, /role/Operator,/role/Deployer]) if owner = sys_user_q;	Allows Admin, Monitor, Operator and Deployer roles to query ALES policies they own.
grant(/priv/execute, //app/policy/ASI/admin/Policy/Analysis, [/role/Admin,/role/Monitor,/role/Operator,/role/Deployer]) if owner = sys_user_q or owner = "";	Allows Admin, Monitor, Operator and Deployer roles to query both policies they own and policies with no owner.
grant([/priv/addMember,/priv/ removeMember], //app/policy/ASI/admin, [/role/Deployer]) if true;	Allows Deployer role to add and remove members to subject and privilege groups.

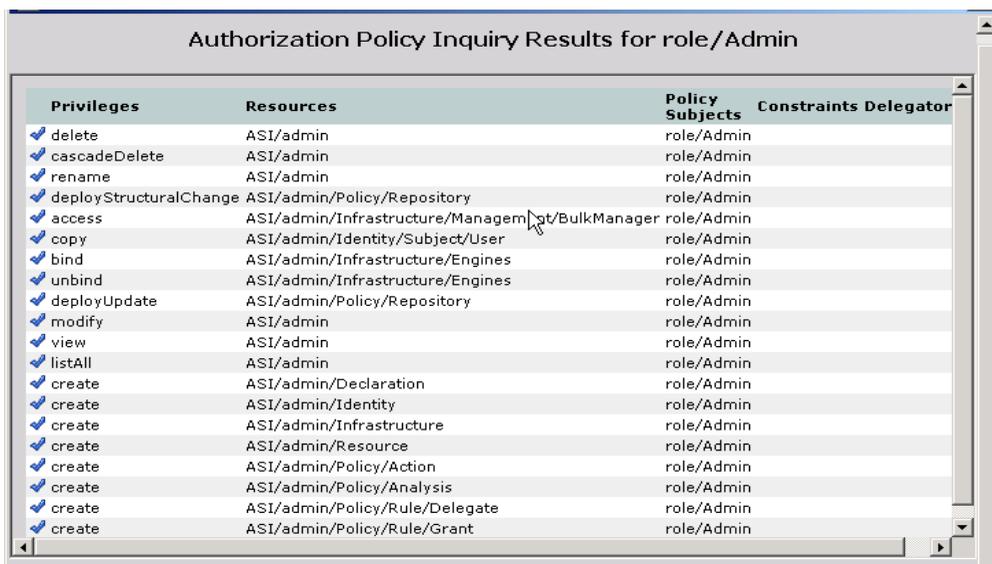
## Viewing Authorization Policies

There several ways to view authorization policies in the Administration Console:

- To see authorization policies set on a specific ALES resource, navigate to and select the resource in the ASI resource tree. Then click Authorization Policy Inquiry in the lower right page.
- To see authorization policies set on a specific ALES role, expand the Identity node and select the Role node. Then click Authorization Policy Inquiry in the lower right page.
- To see all authorization policies, expand the Policy node and select Authorization Policies.

Figure 2-2 below shows the results of an authorization policies query on the Admin role.

Figure 2-2 Authorization Policy Inquiry Results Dialog



Privileges	Resources	Policy Subjects	Constraints	Delegator
delete	ASI/admin	role/Admin		
cascadeDelete	ASI/admin	role/Admin		
rename	ASI/admin	role/Admin		
deployStructuralChange	ASI/admin/Policy/Repository	role/Admin		
access	ASI/admin/Infrastructure/Management/BulkManager	role/Admin		
copy	ASI/admin/Identity/Subject/User	role/Admin		
bind	ASI/admin/Infrastructure/Engines	role/Admin		
unbind	ASI/admin/Infrastructure/Engines	role/Admin		
deployUpdate	ASI/admin/Policy/Repository	role/Admin		
modify	ASI/admin	role/Admin		
view	ASI/admin	role/Admin		
listAll	ASI/admin	role/Admin		
create	ASI/admin/Declaration	role/Admin		
create	ASI/admin/Identity	role/Admin		
create	ASI/admin/Infrastructure	role/Admin		
create	ASI/admin/Resource	role/Admin		
create	ASI/admin/Policy/Action	role/Admin		
create	ASI/admin/Policy/Analysis	role/Admin		
create	ASI/admin/Policy/Rule/Delegate	role/Admin		
create	ASI/admin/Policy/Rule/Grant	role/Admin		

# Setting Up Application Security Administrators

ALES allows you to set up application-level administrators who are responsible for managing the security for a specific application. The application-level administrator will be able to manage the policies protecting resources belonging to that application, but no others. This chapter describes some basic steps for establishing an application-level security administrators and provisioning them with an initial framework for protecting applications. This section provides information on the following topics:

- [“Overview” on page 3-1](#)
- [“Establishing a Resource Parent for the Application” on page 3-2](#)
- [“Create Administrative Users” on page 3-2](#)
- [“Policies for Application-Level Administration” on page 3-3](#)

## Overview

Although the design of the administrative model will vary by use, it is presumed that the task of defining policies to secure an application will be assigned to application-level administrator who has complete rights only for the specific application.

The basic procedure described here for setting up application-level administrators is to create a parent application resource that will contain a representation of the application in the resource tree, create administrator user accounts and groups as needed, and then use policies that will allow the administrators to manage the application’s security.

## Establishing a Resource Parent for the Application

To represent an application in ALES, create a binding application resource to serve as the application parent. Then give the application security administrator the right to build resources under this parent.

To create a binding application resource for an application:

1. Select the Resource node in the navigation tree to display the current resource tree in the right panel of the Administration Console.
2. Right-click the top parent resource that will contain the application and select Add Resource.
3. Enter a resource name and select Binding in the Type field. Then click OK.
4. Right-click the new resource and select Configure Resource.
5. Select Binding Application in the Type field and click OK.

## Create Administrative Users

User accounts are needed for the application security administrators. If you want, you may create application-specific directories containing users and groups for the application.

**Note:** An implicit group named `allusers` is automatically added to all directories.

## Identity Directories

To create a separate directory for an application's users and groups:

1. Select the Identity node in the navigation tree to display the current directories in the right panel of the Administration Console. After ALES is installed, there is one directory named ASI.
2. Click New in the lower right page.
3. On the Create Directory dialog, enter the directory name and click OK.

## Users and Groups

To add a user or group to a directory:

1. Select the Identity node in the navigation tree to display the current directories in the right panel of the Administration Console.

2. Click the directory where you want to add the user or group, then select Edit Users or Edit Groups at the bottom of the page. This displays the directory's groups or users depending on your selection.
3. Select New at the bottom of the page.
4. On the dialog that displays, enter the user or group name and select OK.

## Policies for Application-Level Administration

Once the application parent is defined in the resource tree and the necessary identities have been created, you can use policies to determine administrative access to the application. Here are two examples:

**Note:** A comprehensive understanding of this process can be obtained by examining the policies already in place for ALES components.

- Using policy constraints allows you to limit administrative rights. For example, the following policy assigns the Admin role to Joe only for managing resources for the Petstore application.

```
grant(//role/Admin, //app/policy/ASI/admin/Resource, //user/asi/Joe/)
if resource_is_child(resource, //app/policy/Petstore, no);
```

**Figure 3-1 Using the Resource\_is\_Child Constraint**

Roles	Resources	Policy Subjects	Constraints
Admin	petstore	user/petstore/small	if resource_is_child (resource, //app/policy/petstore)

- Assign Admin role to Bob (user) for the purpose of performing inquiries on policies set on the Petstore application.

```
grant(//role/Admin, //app/policy/ASI/admin, //user/asi/Bob/) if
sys_defined(resource) and resource_is_child(resource,
//app/policy/Petstore, no);
```



# Integrating ALES with Applications

This chapter provides information about ALES built-in support for integration with specific environments.

- [“Overview” on page 4-1](#)
- [“Security Service Modules” on page 4-2](#)
- [“SSM Security Providers” on page 4-3](#)
- [“Integrating ALES with Other BEA Applications” on page 4-5](#)

## Overview

ALES provides a number of built-in solutions for integration with the following environments:

- Microsoft Internet Information Server
- Apache HTTP Server
- Web Services clients
- BEA WebLogic Server
- BEA WebLogic Portal
- AquaLogic Data Services Platform
- AquaLogic Service Bus

Each of these integrations is based on an ALES Security Service Module.

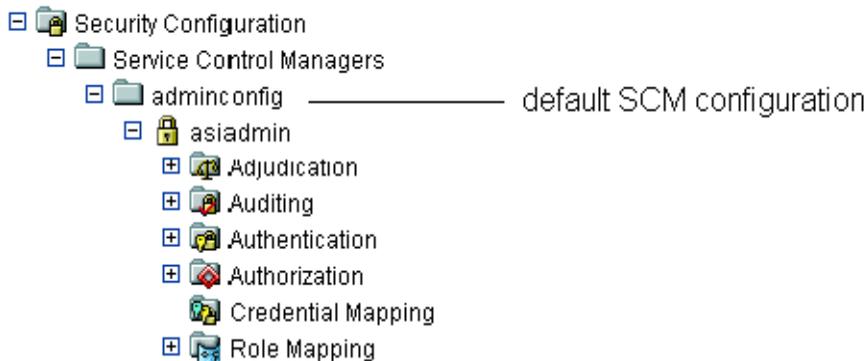
# Security Service Modules

Before a SSM can be integrated with a server, a SSM configuration that specifies the security providers must be created and the configuration must be bound to the SCM running on the same machine.

As shown in [Figure 4-1](#), installation of ALES creates a default SCM configuration named `adminconfig` that contains a SSM configuration and security providers used by the Administration Server itself.

If the SSM instance will be located on the same machine, you can use the SCM and create a SSM configuration under it. If on a separate machine, you must create a new SCM. For step-by-step instructions on managing SCM and SSM configurations, see the Administration Console help.

**Figure 4-1 Default SCM**



To create a SSM configuration:

1. Open the Security Configuration folder.
2. Select Unbound Configurations in the navigation tree and click on `Create a new Security Service Module Configuration` in the right page.
3. On the General tab, complete the following fields and click Create.

**Table 4-1 SSM Configuration ID**

Field	Description
Configuration ID	This entry must match the SSM configuration ID that is specified when the SSM instance is created on the server machine. The configuration ID is the means by which the SSM receives its configuration from the SCM.
Description	(Optional) A brief description of the SSM.

## SSM Security Providers

The security providers needed depend on the requirements of the application. Installing a SSM deploys a JAR file that contains all ALES security providers. However, before any of the security providers can be used, you must use the Administration Console to configure them. You have the option of configuring either the security providers that ship with the product or custom security providers, which you may develop yourself or purchase from third-party security vendors. For more information on how to develop custom security providers, see [Developing Security Providers for BEA AquaLogic Enterprise Security](#). For step-by-step instructions on managing providers, see the Administration Console help.

Note that the process of configuring security providers for the WebLogic Server 9.x SSM is different from that for other SSMs. For more information, see [Chapter 8, “Configuring the WebLogic Server 9.x SSM.”](#)

**Table 4-2 Authentication Providers**

Provider	Description
WebLogic Authenticator	Authenticate users with WebLogic’s embedded LDAP directory.
ALES Identity Asserter	Supports web server authentication and single sign-on between web server SSMs. Use this provider in conjunction with the ALES Credential Mapper.
Database Authenticator	Authenticates users using the ALES relational database provider.
Single Pass Negotiate Identity Asserter	Supports identity assertion using HTTP authentication tokens from the SPNEGO protocol. For more information, see <a href="#">Chapter 13, “Enabling SPNEGO-based Single Sign-on.”</a>
SAML Identity Asserter	Accepts SAML assertions sent using the Browser POST Profile and returns the corresponding user. For more information, see <a href="#">Chapter 12, “Enabling SAML-based Single Sign-On.”</a>

**Table 4-2 Authentication Providers (Continued)**

<b>Provider</b>	<b>Description</b>
Open LDAPAuthenticator	Authenticates users using an Open LDAP directory.
Active Directory Authenticator	Authenticates users using Active Directory.
NTAuthenticator	Authenticates users using Windows NT authentication.
iPlanet Authenticator	Authenticates users using an iPlanet LDAP directory.
Novell Authenticator	Authenticates users using a Novell LDAP directory.
X509 Identity Asserter	Supports identity assertion through an X.509 digital certificate, supporting ASN.1 encoding and decoding

[Table 4-3](#) describes Authorization providers.

**Table 4-3 Authorization Providers**

<b>Provider</b>	<b>Description</b>
WebLogic Authorizer	Authorizes access to resources based on WebLogic security policy.
ASI Authorization Provider	Authorizes access to resources based on ALES security policy.

[Table 4-4](#) describes Credential Mapping providers.

**Table 4-4 Credential Mapping Providers**

<b>Provider</b>	<b>Description</b>
Database Credential Mapper	Returns authentication credentials for a user (username and password) from a database.
SAML Credential Mapper	Returns a SAML assertion for an authenticated user. For more information, see <a href="#">Chapter 12, “Enabling SAML-based Single Sign-On.”</a>
ALES Identity Credential Mapper	Supports web server authentication and single sign-on between web server SSMs. Returns a ALES assertion for an authenticated user.
Weblogic Credential Mapper	Returns authentication credentials for a user (username and password) from the Weblogic LDAP directory.

[Table 4-5](#) describes Role Mapping providers.

**Table 4-5 Role Mapping Providers**

<b>Provider</b>	<b>Description</b>
ASI Role Mapper	Returns a set of roles granted to a user on a protected resource based on ALES security policies.
Weblogic Role Mapper	Returns a set of roles granted to a user on a protected resource based on WebLogic security policies.

## Integrating ALES with Other BEA Applications

ALES includes two SSMs for integrating with WebLogic Server and other BEA applications:

- WebLogic Server 8.1 SSM
- WebLogic Server 9.x SSM

The WebLogic Server SSMs integrate ALES with WebLogic Server and with BEA WebLogic Portal, AquaLogic Data Services Platform, and AquaLogic Service Bus. See the following chapters for more information about configuring ALES to work with those products:

- [Chapter 7, “Configuring the WebLogic Server 8.1 SSM”](#)
- [Chapter 8, “Configuring the WebLogic Server 9.x SSM”](#)
- [Chapter 9, “Integrating with WebLogic Portal”](#)
- [Chapter 10, “Integrating with AquaLogic Data Services Platform”](#)
- [Chapter 11, “Integrating with AquaLogic Service Bus”](#)



# Configuring the Web Server SSM

This section describes the Web Server Security Service Module, including tasks that you must perform after installing and completing the post-installation tasks. See *Installing Security Service Modules* for information about how to install the Web Server Security Service Module for Microsoft IIS or the Apache Web Server. The following topics are covered in this section:

- “Understanding the Web Server SSMs” on page 5-1
- “Configuring and Deploying Policy for the Web Server SSM” on page 5-11
- “Configuring the Web Server Environmental Binding” on page 5-17
- “Configuring Web Single Sign-on with ALES Identity Assertion” on page 5-26

## Understanding the Web Server SSMs

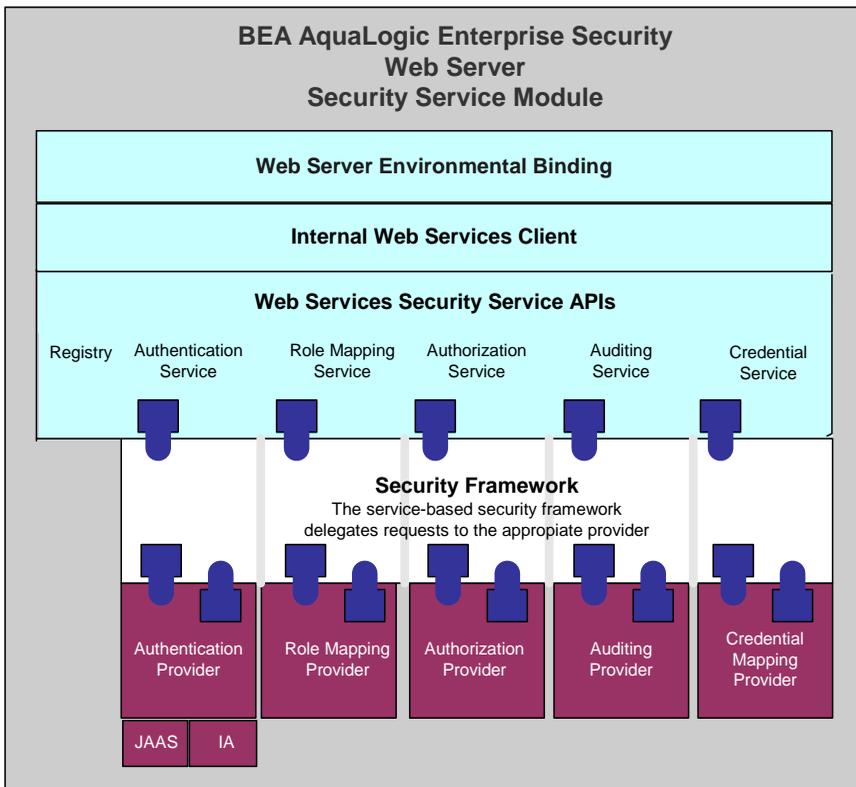
This section includes the following topics describing the Web Server SSMs:

- “Web Server SSM Overview” on page 5-2
- “Web Server Environmental Binding” on page 5-3
- “Web Server SSM Features” on page 5-4
- “Web Server Constraints and Limitations” on page 5-10
- “Web Server SSM Integration Tasks” on page 5-11

## Web Server SSM Overview

An ALES Web Server SSM provides the environmental bindings between the ALES and a web server. It can provide six distinct services: Registry, Authentication, Authorization, Auditing, Role Mapping, and Credential Mapping.

Figure 5-1 Web Server SSM Components



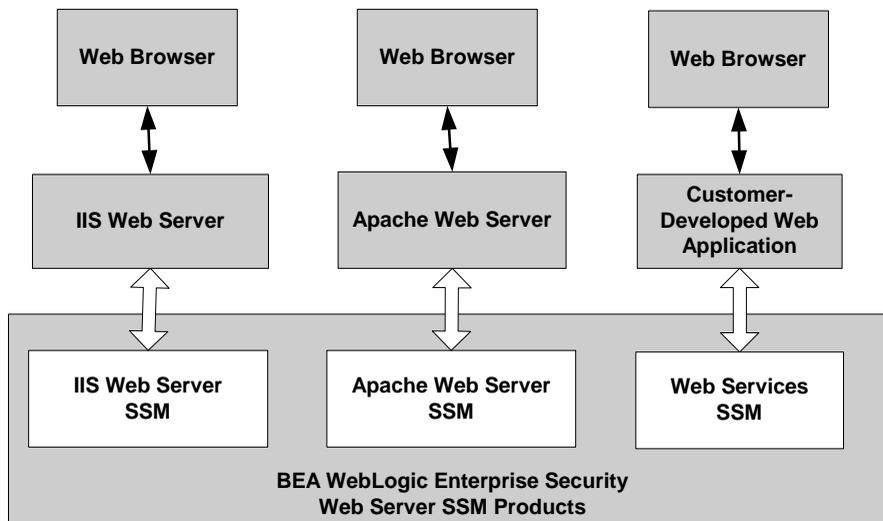
A Web Server SSM makes access decisions for the web server to which it is bound. The security configuration on which the access control decisions are based is defined and deployed by the Administration Server via the Security Control Module.

A Web Server SSM can be tailored to specific needs. Using templates provided as part of the product, security developers can customize the look and feel of authentication pages and

configure parameters that allow fine tuning for a particular installation. Web applications can have information added to the HTTP request by the security framework, such as roles and response attributes.

ALES provides three Web Server SSMs: IIS Web Server SSM (SSM), Apache Web Server SSM, and Web Services SSM (see [Figure 5-2](#)).

**Figure 5-2 Web Server SSM Components**



## Web Server Environmental Binding

The environmental binding is used to bind to and interact with web servers. Binding a Web Server SSM to the server projects the ALES subsystem into the web server environment. The SSM accepts HTTPS requests from the web server and presents them to the ALES security framework.

Bindings are provided for two types of web servers: ASF Apache and Microsoft IIS. The second function is ultimately for enforcing access control and providing a means of implementing the SAML Browser/POST profile. Additionally, the Web Server SSM implements the server-side includes (SSIs) that process SAML Browser/POST profile.

## Web Server SSM Features

This section describes the Web Server SSM features in the following sections

- [“Web Single Sign-on Capabilities” on page 5-4](#)
- [“Authentication Service Features” on page 5-6](#)
- [“Authorization Service Features” on page 5-7](#)
- [“Auditing Service Features” on page 5-8](#)
- [“Role Mapping Features” on page 5-8](#)
- [“Credential Mapping Features” on page 5-8](#)
- [“Administration Features” on page 5-9](#)
- [“Session Management Features” on page 5-9](#)
- [“Configuration Features” on page 5-10](#)

## Web Single Sign-on Capabilities

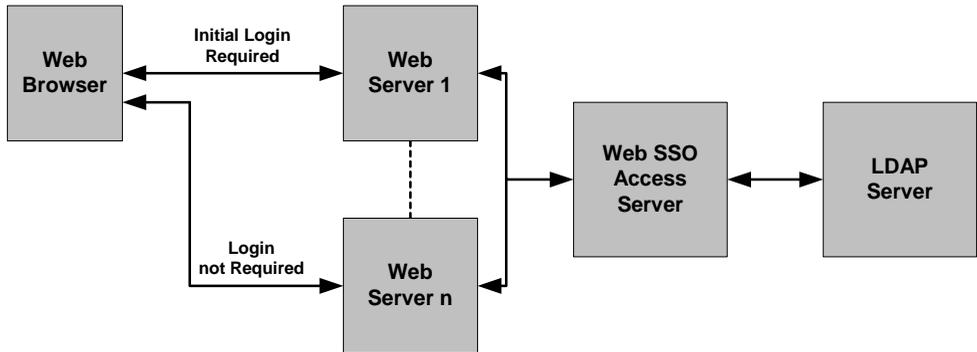
This section covers the following topics:

- [“What is Web Single Sign-On?” on page 5-4](#)
- [“Single Sign-On Use Cases” on page 5-5](#)
- [“Single Sign-On with ALES Identity Assertion” on page 5-6](#)

### What is Web Single Sign-On?

Web single sign-on enables users to log on to one web server and gain access to other web servers in the same domain without supplying login credentials again, even if the other web servers have different authentication schemes or requirements. [Figure 5-3](#) shows the basic components of a web single sign-on service.

Figure 5-3 Web Single Sign-on



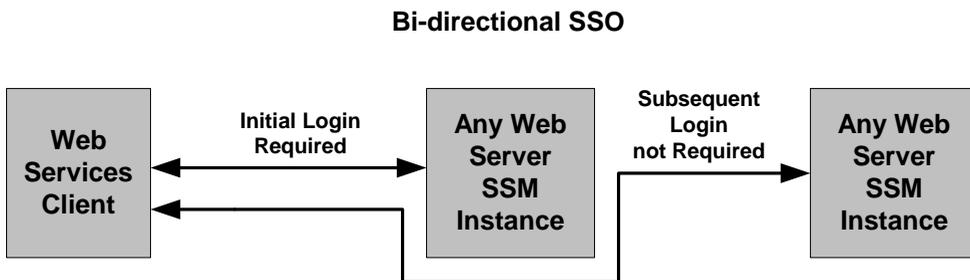
While web single sign-on facilitates access and ease of use, it does not improve security. In fact, security requirements should be considered when implementing a web single sign-on solution.

### Single Sign-On Use Cases

The Web Server SSM supports the following single sign-on (SSO) use cases.

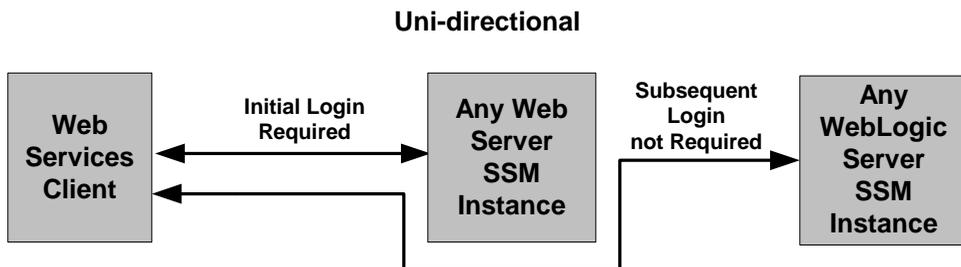
- **Bi-directional SSO among Web Server SSMs**—Once users are authenticated by a Web Server SSM, they are given a identity assertion token that they can use to access other Web Server SSM instances without being required to authenticate again (see [Figure 5-4](#)).

Figure 5-4 Web Server SSM to Web Server SSM Single Sign-on



- **Uni-directional SSO from Web Server SSM to WebLogic Server SSM instances**—Once users are authenticated by a Web Server SSM, they are given an identity assertion token that they can use to access WebLogic Server SSM instances without being required to authenticate again (see [Figure 5-5](#)).

**Figure 5-5 Web Server SSM to WebLogic Server SSM Single Sign-On**



### Single Sign-On with ALES Identity Assertion

The Web Server and WebLogic Server SSMs support single sign-on using the ALES Identity Assertion provider. For instructions on how to implement Single Sign-On, see [Chapter 13](#), “[Enabling SPNEGO-based Single Sign-on.](#)”

### Authentication Service Features

The authentication service supports the following features:

- **Conversion of JAAS callbacks to asynchronous**—Although the authentication service is JAAS based, the Web Server SSM masks the JAAS synchronous callback protocol from the web server. When form-based authentication is configured, the credentials are initially gathered and used for authentication. In most cases, an initial form gathers all credentials necessary for authentication.
- **All standard JAAS callbacks**—Sun Microsystems defines seven types of callbacks: NameCallback, PasswordCallback, ChoiceCallback, ConfirmationCallback, LanguageCallback, TextInputCallback, TextOutputCallback.

**Note:** If a new callback type is encountered during authentication, the Web Server SSM ignores it.

- **Multiple authentication phases**—JAAS may ask several series of questions using callbacks. Therefore, the SSM does not assume that answering one set of callbacks is sufficient.
- **Web farms**—The following features are supported for web farms:
  - **Redirect during credential gathering**—Within a web farm it is possible for a series of questions (on a form) to start on one machine and be transparently redirected to another machine within that web farm.
  - **Single sign-on**—Within a web farm it is possible that a user is authenticated on one machine and transparently redirected to another. However, a mechanism must be available by which the second machine can accept the identity from the first without having to re-authenticate the user. The identity is only shared within the same cookie domain.
- **Single sign-on with other SSMs**—Can share identity with a custom application that is protected by the Java SSM or another client of the Web Server SSM. The Java SSM and the Web Server SSM use the ALES Credential Mapper and ALES Identity Assertion providers. Single sign-on is limited to a cookie domain.
- **SAML Browser/POST profile**—Consumes an identity assertion from a SAML 1.1 Browser/POST transaction and provides an identity transfer service to serve SAML 1.1 identities to remote systems.
- **Custom, form-based authentication**—Allows for the editing and customizing the forms used in form-based authentication.

## Authorization Service Features

The authorization service supports the following features:

- **Resource form**—De-references any use of ". ." and decodes URL encoding. The resource is presented as the path element of a URL and the file or application name. For example, `http://www.example.com/framework.jsp?CNT=index.htm&FP=/products/aqualogic/` is presented as `/framework.jsp`. The query arguments CNT and FP and associated values are made available in the application context.
- **Allows for unprotected URLs**—Always uses the `isAuthenticationRequired()` method to check if a resource is protected by a security system. This feature is important because you may want to leave some web server resources unprotected.
- **Checks for resource authorization**—When checking for resource authorization, the following features are supported:

- **Includes rich web service request context**—Because a web application cannot know what elements may be required for enforcement of security at the time of its authoring, it is important that information about the web services request be available in the context given to the security subsystem. The Web Server SSM provides HTTP headers, cookies, query arguments, and form values to the security subsystem. The SSM also decodes all URL encoded context elements before presenting them to the security subsystem.
- **Works with existing unmodified web applications**—Does not require modification or special code to work with existing web applications running on the web server.
- **Retrieves response attributes during authorization**—Retrieves response attributes during the authorization process and provides them in a form that a layered web application can use.

## Auditing Service Features

The auditing service has the following capabilities:

- **Audits all transactions**—All authentications, identity assertions, authorizations, role mappings, credential mappings and audit failures are automatically made available to the auditing infrastructure by the ALES security framework.
- **Audits session cleanup activity**—The occurrence of idle and absolute time-outs are audited.

## Role Mapping Features

The role mapping service supports hard-coded roles in applications. Generally hard-coding behavior into an application based on roles is not recommended. It is possible, however, that some customers may need to replace an existing system that uses this mechanism or may want to use roles for user interface personalization. Support for this feature requires that a list of mapped roles available from a security provider for a particular request be provided in a usable form by applications running within the web server.

**Note:** It is important to note that roles are not global in ALES but can change depending upon the resource and various elements of the context.

## Credential Mapping Features

ALES defines two types of credential objects: username/password credentials and generic credentials; however, there is no limitation as to the format of objects that can be used. Credentials can be mapped and associated with a resource and identity or an alias.

The credential mapping service has the following features:

- **Provides mapped username/password credentials**—Extracts mapped username/password credentials to the application running within the SSM. This username/password can be used for legacy SSO to log into a database or other system. The Web Server SSM does not use these credentials itself; it will make them available to the web server application.
- **Supports unknown credential types**—Provides a way to inject other credential formats. Since these other formats are unknown to the SSM, they must be converted to a string before being presented to the application.

## Administration Features

Administering the security configuration involves writing policies for users, groups, roles, and the web application resources that the SSM protects. The Web Server SSM has the following features:

- **Presents the full URL**—The full URL (including the protocol, server name, port, full path, and query string) is presented to the ALES security framework as part of the context to allow its use in access control policy. Note that the resource presented to the system is in the canonical form. For example, for a web server with the names `www.bea.com`, `www.beasys.com`, `www.web.internal.bea.com`, and `204.236.43.12`, the canonical name is `www.bea.com`.
- **Uses the HTTP method as the action**—The HTTP method (GET, POST, HEAD, PUT) is presented by the Web Server SSM as the action for authorization. In the administration system the privilege must match the action for a policy so this feature allows for separate security policies to be applied to POSTs, GETs, and other methods.
- **Passes in an application context**—The application context is passed through to the SSM's authorization and role mapping security providers and is associated with any audit records logged. This context contains values relevant to the request environment at the time the security provider processes the call.

## Session Management Features

To manage session behavior, the Web Server SSM supports the following capabilities:

- **Inactivity time-out**—terminates a session after it has been inactive for a configurable period of time

- **Absolute time-out**—terminates a session after a certain (usually large) period of time, thereby preventing a client from staying perpetually connected, which can be a security risk.
- **User logoff**—allows for user initiated logoff.
- **Forced logoff**—forces immediate logoff by terminating the session for a single DNS domain.
- **Session cookie**—uses session cookie, not persistent cookies.

### Configuration Features

The web server is configured to use the filter component of the Web Server SSM. Local configuration of the web server should only be necessary once and should be static. The Web Server SSM has the following configuration capabilities:

- **Logging channel**—to support configuration and debugging.
- **Configurable flag to control logging and debug mode**—to determine what messages are logged.

### Web Server Constraints and Limitations

The Web Server SSM has the following constraints and limitations:

- Does not support cookie-based cross domain single sign-on except through SAML Browser/POST profile.
- Does not support cookie-based cross domain forced logoff.
- To support web farms, local configurations on each web server machine must be manually synchronized.
- Does not support special handling of third-party cookies.
- Requires use of cookies to maintain session state.
- The Web Server SSM must be manually configured into the web server.
- Does not support load-balancing or failover in accessing the Web Services SSM.
- Must be installed on the same machine as the web server to which it is bound.
- Does not support SAML Browser/Artifact profile.

- Does not preserve the original POST data during redirection to an authentication form.
- Does not save and transfer credentials between machines when more than one machine is involved in an attempt to authenticate a user. Within a web farm, it is possible for a series of questions (on a form) to start on one machine and be transparently redirected to another machine within that web farm. The SSM does not save credentials that have already been entered in the same authentication attempt. Therefore, users are forced to re-enter credential information when more than one machine is involved.

## Web Server SSM Integration Tasks

This section provides an overview of integration tasks. Integration tasks center on managing SSM configurations (including the security providers) and configuring the web server to use the web filters. For additional information, see [Installing Security Service Modules](#).

The major tasks performed are:

1. Create a SCM and a SSM configuration using the Administration Console. This includes specifying the security providers.
2. Create a parent resource for the application. This will contain ALES's representation of the application.
3. Create the SSM instance on the web server machine and enroll it in the ALES trust environment. The instance will use the security providers defined in step 1 above.

During the instance creation process, the `default.properties` configuration file is created. This file contains the connection information for the ALES services.

4. Configure the web server environmental binding. This loads the web filter on the server and establishes the connection between the web server and ALES.

## Configuring and Deploying Policy for the Web Server SSM

Developing a policy for a web application typically begins by determining which resources you want to protect. You then create authorization mapping policies to define access privileges and roles for each resource, and under what specific conditions. Next, you create role mapping policies that control which users and groups have membership in the defined roles, and under what conditions.

This section describes how to create resources and define authorization and role mapping policies to protect a sample Web server application. This section also describes how to deploy this policy

to the Web Services SSM that you will use to control access to sample Web server application resources.

AquaLogic Enterprise Security provides three tools for configuring application policy: the Administration Console, the Policy Import Tool, and Business Logic Manager (BLM) API. In this section you are directed to use the Administration Console to configure policy.

For more information on how to use the Administration Console to configure policy, see the *Policy Managers Guide* and Console Help.

For instructions on how to use the Policy Import Tool to import policy files, see the [Importing Policy](#) section in the *Policy Managers Guide*.

For information on the BLM, see the [BLM API Javadocs](#).

To configure and deploy policy for the Web Server SSM, perform the following tasks:

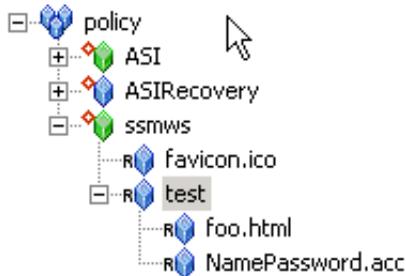
- [“Creating Resources” on page 5-12](#)
- [“Creating Policies” on page 5-14](#)
- [“Modifying Admin and Everyone Role Mapping Policies” on page 5-15](#)
- [“Configuring the Application Deployment Parent” on page 5-15](#)
- [“Distributing Policy and Security Configuration” on page 5-16](#)

## Creating Resources

This section describes how to use the Administration Console to create resources for the sample web server application resource.

[Figure 5-6](#) shows the resources that you must create for the sample IIS Web Server configuration. You create the same resources for the Apache Web Server, except that you assign the `NamePasswordForm` a file extension of `.html`, instead of `.acc`.

### **Figure 5-6 Resources Tree for the IIS Web Server**



To create these resources, perform the following steps:

1. In the Administration Console, open the Resources folder, and click Resources to display the Resources page.
2. Select Policy and click New. The Create Resource dialog box appears.
3. In the Name text box, enter `ssmws`, select Binding from the Type drop-down list box, and click Ok. The `ssmws` resource appears under the Policy node.
4. Select the `ssmws` resource and click Configure. The Configure Resource dialog box appears.
5. From the Type drop-down list box, select Binding Application, check the Distribution Point check box to on, and click Ok.
6. Select the `ssmws` resource and click New. The Create Resource dialog box appears.
7. In the Name text box, enter `favicon.ico`, and click Ok. The resource appears under `ssmws`.
 

**Note:** The `favicon.ico` file is an icon requested by the Internet Explorer and Mozilla browsers for bookmarking a URL.
8. Select the `ssmws` resource and click New. The Create Resource dialog box appears.
9. In the Name text box, enter `test`, and click Ok. The resource appears under `ssmws`.
10. Select the `test` resource and click New. The Create Resource dialog box appears.
11. In the Name text box, enter `foo.html` and click Ok. The `foo.html` resource appears under the `test` resource.
12. Click the `test` resource and click New. The Create Resource dialog box appears.

13. In the Name text box, enter `NamePasswordForm.acc` for IIS (or `NamePasswordForm.html` for Apache), and click Ok. The resource appears under `test`.

## Creating Policies

This section describes how to use the Administration Console to create authorization and role mapping policies to protect the sample Web server application resources. It includes authorization policies for the HTML files and role mapping policies to assign membership to those roles.

[Table 5-1](#) lists and describes the authorization policies that you will create to protect the sample Web server application resources. These authorization policies allow users who are members of the `Everyone` role the `Get` access privilege to the `favicon.ico` and `GET` and `POST` access privileges to `NamePasswordForm.acc` (so users who have membership in the `Everyone` role can access the username/password form when authentication for a protected resource is needed). The policy also restricts access to `foo.html` to users in the `Admin` role.

**Table 5-1 Sample Web Server Application Resources Authorization Policies**

Policy	Description
<code>grant(GET, //app/policy/ssmws/favicon.ico, //role/Everyone) if true;</code>	Allows unauthenticated users to access images used on the application login page.
<code>grant(GET, POST, //app/policy/ssmws/test/NamePasswordForm.acc, //role/Everyone) if true;</code>	On the IIS Web Server, grants <code>GET</code> and <code>POST</code> privileges for those in the <code>Everyone</code> role to access the <code>NamePasswordForm.acc</code> page.  <b>Note:</b> For the Apache Web Server, use <code>NamePasswordForm.html</code> .
<code>grant(GET, //app/policy/ssmws/test/foo.html, //role/Admin) if true;</code>	Grants <code>GET</code> privileges for those in the <code>Admin</code> role to access the <code>foo.html</code> page.

To create the authorization polices listed in [Table 5-1](#), perform the following steps:

1. Open the Policy folder, and click Policy. The Policy page appears.
2. Click Authorization Policies and click New. The Create Authorization Policy dialog box appears.
3. Select the Grant radio button.

4. To add privileges for the first policy listed in [Table 5-1](#), click the Privileges tab, select the `GET` privilege from the Select Privileges from Group list box and add it to the Selected Privileges box.
5. Click the Resources tab, select the `favicon.ico` resource from the Child Resource box and add it to the Selected Resources box.
6. Click the Policy Subjects, select the `Everyone` role from the Roles List box, add it to the Selected Policy Subjects box, and click Ok.
7. Repeat steps 2 to 6 for each of the remaining authorization policies listed in [Table 5-1](#). Notice that the `Admin` role is assigned to the `foo.html` resource.

## Modifying Admin and Everyone Role Mapping Policies

This section describes how to use the Administration Console to modify the role mapping policies that will be used to control access to the sample Web Server application resources.

To modify the `Admin` and `Everyone` role mapping policies, perform the following steps:

1. Open the Policy folder, and click Role Mapping Policies. The Role Mapping Policies page appears.
2. Select the `Admin` role for ASI, and click Edit. The Edit Role Policy dialog appears.
3. Click the Resources tab, add the `ssmws` resource, and click Ok.
4. Repeat steps 2 to 4 for the `Everyone` role to add `ssmws` to the `Everyone` role.

## Configuring the Application Deployment Parent

For the sample Web server application, the Application Deployment Parent setting on the ASI Authorization provider and the ASI Role Mapping provider must be set to `//app/policy/ssmws` and must be bound to the provider.

To configure these providers, perform the following steps:

1. In the Administration Console, click the ASI Authorization provider and click the Details tab.
2. Set the Application Deployment parent to `//app/policy/ssmws`, and click Apply.
3. Click on the Bindings tab and click bind to bind `//app/policy/ssmws` to this provider.
4. Repeat steps 1 to 3 for the ASI Role Mapping provider.

## Configuring the ALES Identity Assertion and Credential Mapping Providers

To configure the ALES Identity Assertion and ALES Credential Mapping providers, perform the following steps:

**Note:** The ALES Identity Assertion provider and the ALES Credential Mapping provider work with one another; therefore, it is important that you ensure that their configuration settings match.

1. In the Administration Console, click the ALES Identity Assertion provider, select the Details tab, set the parameters as listed in [Table 5-2](#), and click Apply.
2. Click the ALES Credential Mapping provider, select the Details tab, set the parameters as listed in [Table 5-2](#), and click Apply.

**Table 5-2 ALES Identity Asserter and Credential Mapper Provider Settings**

Parameter	Setting
Trusted CAKeystore	{HOME}/ssl/demoProviderTrust.jks {HOME} is replaced with the SSM instance directory at runtime.
Trusted CAKeystore Type	JKS
Trust Cert Alias	demo_provider_trust
Trusted Cert Alias Password and Confirmation	password
Trusted Keystore	{HOME}/ssl/demoProviderTrust.jks {HOME} is replaced with the SSM instance directory at runtime.
Trusted Keystore Type	JKS

## Distributing Policy and Security Configuration

Distribute the policy and security configuration to the Web Server SSM.

For information on how to distribute policy and security configuration, see the Console Help. Be sure to verify the results of your distribution.

## Configuring the Web Server Environmental Binding

The Web Server Environmental Binding configuration procedures vary depending on the type of Web Server SSM you are configuring. AquaLogic Enterprise Security supports two Web server SSMs that require configuration of the Web Server Environmental Binding: the Microsoft IIS Web Server SSM and the Apache Web Server SSM. For configuration instructions, see the appropriate topic below:

- [“Configuring the Environmental Binding for the Microsoft IIS Web Server” on page 5-17](#)
- [“Configuring the Environmental Binding for the Apache Web Server” on page 5-23](#)

### Configuring the Environmental Binding for the Microsoft IIS Web Server

To configure the environmental binding for Microsoft IIS Web Server, perform the following tasks:

- [“Configuring the Microsoft IIS Web Server Binding Plug-In File” on page 5-17](#)
- [“Configuring the NamePasswordForm.acc File for the IIS Web Server” on page 5-22](#)
- [“Deploying and Testing the IIS Web Server Sample Application” on page 5-22](#)

#### Configuring the Microsoft IIS Web Server Binding Plug-In File

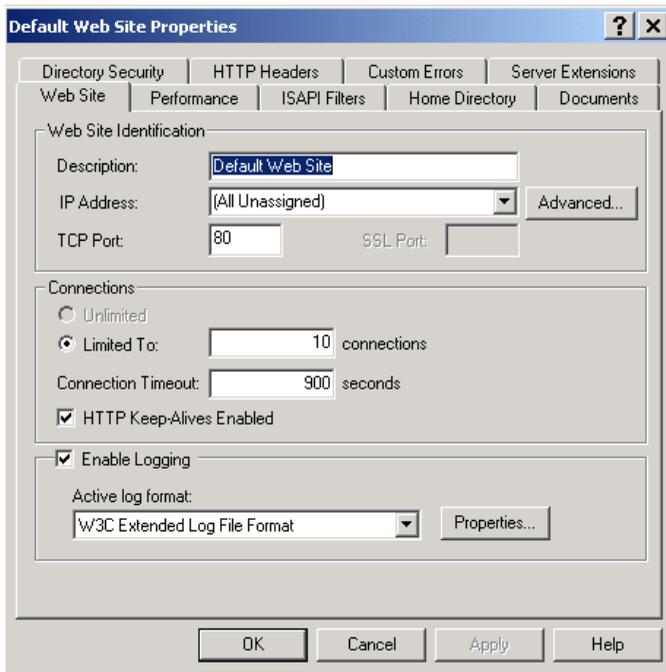
**Note:** This task assumes you have created an instance of the IIS Web Server SSM according to instructions provided in [“Creating an Instance of the Web Server Security Service Module” on page 4-8](#).

The IIS Web Server Binding plug-in file is named `wles_isapi.dll`. This file is located in the `BEA_HOME\ales22-ssm\iis-ssm\lib` directory.

To configure the Microsoft IIS Web Binding plug-in, perform the following steps:

1. To open the Internet Information Services Manager, click `Start>Settings>Control Panel`, select `Administrative Tools`, and double-click `Internet Services Manager`. The Internet Information Services Window appears.
2. In the left-hand pane, expand the machine node, right click `Default Web Site`, and select `Properties`. The Default Web Site Properties dialog box appears (see [Figure 5-7](#)).

Figure 5-7 IIS Web Site Properties Dialog



3. Click the ISAPI Filters tab, click the Add button, assign a name to the ISAPI filter, use the Browse button to add the `wles_isapi.dll` file (which is located in `BEA_HOME\ales22-ssm\iis-ssm\lib` directory), and click OK.
4. Click the Directory Security tab. The Authentication Methods dialog appears (see [Figure 5-8](#)).

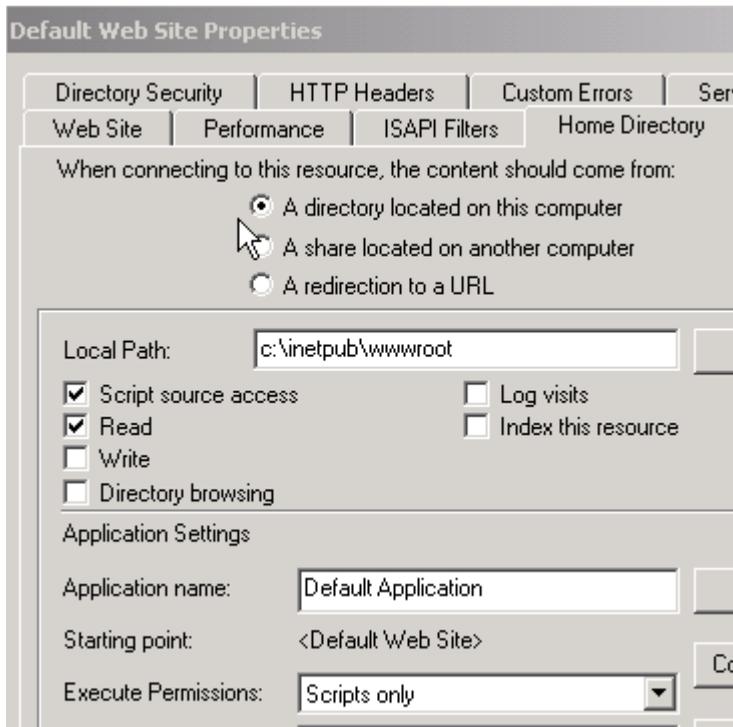
**Figure 5-8 Authentication Methods Dialog**

5. Click the Edit button for Anonymous Access, check the Anonymous username, and, if necessary, change the username and password to ensure that the Anonymous user has Read and Read/Execute permissions on the following directories:

```
BEA_HOME\ales22-ssm\iis-ssm\lib
BEA_HOME\ales22-ssm\iis-ssm\instance\iisssmdemo\ssl
BEA_HOME\ales22-ssm\iis-ssm\instance\iisssmdemo\config
```

6. If you put the NamePasswordForm.acc file in a virtual directory, repeat the previous step for the virtual directory as well.
7. Return to the Default Web Site Properties dialog box (see [Figure 5-7](#)) and click the Home Directory tab. The Home Directory dialog appears (see [Figure 5-9](#)).

**Figure 5-9 IIS Web Site Home Directory Dialog**



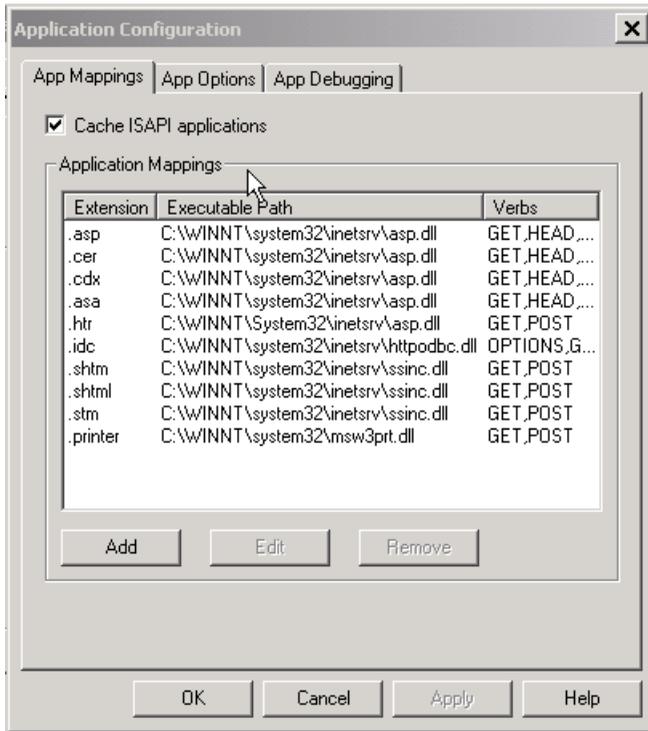
8. Verify that the property settings match the information in [Table 5-3](#) and click Apply and OK.

**Table 5-3 Home Directory Setting**

Property	Setting
Local Path	c:\inetpub\wwwroot
Application name	Default Application
Execute Permissions	Scripts Only

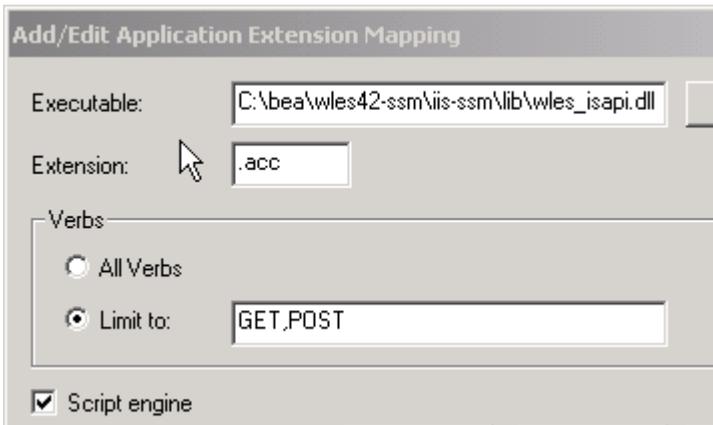
9. Click the Configuration button. The Application Configuration dialog appears (see [Figure 5-10](#)).

**Figure 5-10 IIS Web Site Application Configuration Dialog**



10. Click the Add button. The Add/Edit Application Extension Mapping Dialog appears (see [Figure 5-11](#)).

**Figure 5-11 IIS Web Site Add/Edit Application Extension Mapping Dialog**



11. Use the Browse button to add the `wles_isapi.dll` file to the Executable field, fill in the other fields as shown in [Figure 5-11](#), and click OK.
12. Click OK to close the remaining windows.
13. Right click the Default Web Site again and start the Default Web Site. (Stop the Web Site first if necessary.)
14. Re-open the Default Web Site Properties dialog box and select the ISAPI Filters tab. The IIS Web Server Binding Plug-in status shows a green arrow to indicate that the IIS Web Server Binding Plug-in is loaded. If the green arrow is not displayed, add the `wles_isapi.dll` file again and start the IIS Web Server.

**Note:** Be sure to start the IIS Web server with IIS SSM after you have started the Web Services SSM and ARME.

## Configuring the NamePasswordForm.acc File for the IIS Web Server

Configure the `NamePasswordForm.acc` file for the IIS Web Server as follows:

```
<FORM METHOD=POST ACTION="test/NamePasswordForm.acc">
```

## Deploying and Testing the IIS Web Server Sample Application

To set up the sample web application, perform the following steps:

**Note:** The Web Services SSM must be started before you perform this task because the filter and extension attempt to connect to the Web Services SSM when they are loaded by the Web server.

1. Set up the IIS Server/wwwwroot/test directory as shown in [Figure 5-12](#) and copy the following files to the test directory:

- NamePasswordForm.acc
- foo.html
- atnfailure.html
- atzfailure.html

**Note:** The NamePasswordForm.acc file is provided in the `BEA_HOME\ales22-ssm\iis-ssm\instance\instancename\templates` directory. The `foo.html`, `atnfailure.html` and `atzfailure.html` files are not provided in the product installation kit. You should use your own versions of these files.

**Figure 5-12 Deploying the Sample Application on the IIS Web Server**



2. Start the IIS Web Server, open a browser and go to `http://<machine_name_with_DNS_suffix>:80/test/foo.html`.
3. You are redirected to `NamePasswordForm.acc`.
4. Enter the system username/password (a default system username and password was set when you installed the Administration Application) and click OK. You are granted access to `foo.html`.

## Configuring the Environmental Binding for the Apache Web Server

To configure the Apache Web Server, perform the following tasks:

- [“Downloading and Installing the Apache Web Server” on page 5-24](#)
- [“Configuring the ALES Module” on page 5-24](#)

- “Configuring the NamePasswordForm.html File for the Apache Web Server” on page 5-25
- “Deploying and Testing the Apache Web Server Sample Application” on page 5-25

## Downloading and Installing the Apache Web Server

To download and install the Apache Web Server software, perform the following steps:

1. Go to the Apache download web site at <http://httpd.apache.org/download.cgi> and download and install the software.
2. Verify the following two modules are included in the installation:
  - `ServerRoot/modules/mod_include.so`
  - `ServerRoot/modules/mod_ssl.so`

where `ServerRoot` is the Apache installation directory.

**Note:** The Apache Web Server Security Service Module (SSM) requires that the above two modules be included in the Apache installation; otherwise the Secure Sockets Layer (SSL) and the Security Assertion Markup Language (SAML) server-server include (SSI) related functions will not work.

**Note:** You may build your own 2.0.x version of the Apache Web Server with the above mentioned modules. If the modules are built into Apache, there may be no such files.

## Configuring the ALES Module

**Note:** This task assumes you have created an instance of the Apache Web Server SSM according instructions provided in “Creating an Instance of the Web Server Security Service Module” on page 4-8.

The ALES module contains only one file. For Windows, the file name is `mod_wles.dll`. For Sun Solaris and Linux, the file name is `mod_wles.so`.

To install and configure the ALES module:

1. Open the `ServerRoot/conf/httpd.conf` file and add a `LoadModule` directive. There are several `LoadModule` directives in the `LoadModule` section of the `httpd.conf` file. Add the following line to the end of the `LoadModule` section:

```
LoadModule wles_module <APACHE_SSM_HOME>/lib/mod_wles.so
```

where `<APACHE_SSM_HOME>` is the Apache Web Server SSM installation directory.

For example:

For Windows systems:

```
LoadModule wles_module c:\bea\ales22-ssm\apache-ssm\lib\mod_wles.dll
```

For UNIX systems:

```
LoadModule wles_module
/home/tiger/bea/ales22-ssm/apache-ssm/lib/mod_wles.so
```

2. Add a `WLESConfigDir` directive right after the above `LoadModule` directive as follows:

```
<IfModule mod_wles.cpp>
WLESConfigDir <APACHE_SSM_HOME>/instance/<instance_name>/config
</IfModule>
```

where the `config` directory is the directory that contains the `default.properties` file.

**Note:** In the `IfModule` condition, be sure to specify `mod_wles.cpp`, not `mod_wles.c`.

3. To make sure your server works properly, configure the `ServerName`. For example:

```
ServerName www.yourservername.com:8080
```

4. Change the `Group` directive to have the Apache Web Server running as the `asiusers` group so it can read the `mod_wles` file and other required files:

```
Group asiusers
```

5. Edit the `envvars` file in the `ServerRoot/bin` directory, appending the directory where `mod_wles.so` resides to the default `LD_LIBRARY_PATH`, so that the file looks like this:

```
LD_LIBRARY_PATH="/www/apache/lib:$LD_LIBRARY_PATH:<APACHE_SSM_HOME>/lib"
```

**Note:** This step ensures that the Apache Web Server can load the dependency libraries for the `mod_wles` file.

6. Use the Apache `ctl` script to start or restart Apache Web Server in the `ServerRoot/bin` directory.

## Configuring the NamePasswordForm.html File for the Apache Web Server

Configure the `NamePasswordForm.html` file for the Apache Web Server as follows:

```
<FORM METHOD=POST ACTION="/test/NamePasswordForm.html">
```

## Deploying and Testing the Apache Web Server Sample Application

To set up the sample web application, perform the following steps:

1. Set up the Apache `Server/wwwroot/test` directory as shown in [Figure 5-13](#) and copy the following files to the `test` directory:

## Configuring the Web Server SSM

- NamePasswordForm.html
- foo.html
- atnfailure.html
- atzfailure.html

**Note:** The NamePasswordForm.html file is provided in the `BEA_HOME\ales22-ssm\apache-ssm\instance\<instancename>\templates` directory. The `foo.html`, `atnfailure.html` and `atzfailure.html` files are not provided in the product installation kit. You should use your own versions of these files.

**Figure 5-13 Deploying the Sample Application on the Apache Web Server**



2. Start the Apache Web Server, open a browser, and go to `http://<hostmachine.cookieDomain>:8088/test/foo.html`.
3. You are redirected to `NamePasswordForm.html`
4. Enter the system username/password (a default system username and password was set when you installed the Administration Application) and click OK. You are granted access to `foo.html`.

## Configuring Web Single Sign-on with ALES Identity Assertion

You can configure web single sign-on (SSO) for the following use cases:

- Bi-directional web single sign-on between Web Server Security Service Modules (SSMs)  
With SSO configured, any user that authenticates to one Web Server SSM can access any other Web Server SSM in the cookie domain without having to re-authenticate.
- Uni-directional web single sign-on between Web Server SSMs and WebLogic Server SSMs  
With SSO configured, any user that authenticates to one Web Server SSM can access any other WebLogic Server SSM in the cookie domain without having to re-authenticate.

However, a user that authenticates to a WebLogic Server SSM *cannot* access another WebLogic Server SSM or another Web Server SSM without re-authenticating.

For configuration instructions, see the following topics:

- [“Configuring Web Server SSMs to Web Server SSMs for SSO” on page 5-27](#)
- [“Configuring Web Server SSMs to WebLogic Server SSMs for SSO” on page 5-27](#)

## Configuring Web Server SSMs to Web Server SSMs for SSO

To configure Web Server SSM to Web Server SSM to support web single sign-on, perform the following steps:

1. Using the Administration Console, configure the ALES Identity Assertion and ALES Credential Mapping providers for each Web Server SSM that is to participate in web single sign-on.
2. Configure the ALES Identity Assertion provider and the ALES Credential Mapping provider in each of the Web Server SSMs to use the same Trusted Cert Alias, Trusted Keystore, and Trusted Keystore Type.
3. Deploy the SSM configurations to the SSMs.

For instructions on how to perform the above steps, see the Console Help for the Administration Console.

## Configuring Web Server SSMs to WebLogic Server SSMs for SSO

To configure Web Server SSM to WebLogic Server SSM to support web single sign-on, perform the following steps:

1. Using the Administration Console, configure the ALES Identity Assertion and ALES Credential Mapping providers for each Web Server SSM and WebLogic Server SSM that is to participate in web single sign-on.
2. Configure the ALES Identity Assertion provider and the ALES Credential Mapping provider in each of the SSMs to use the same Trusted Cert Alias, Trusted Keystore, and Trusted Keystore Type.
3. When configuring the ALES Identity Assertion provider for each of the WebLogic Server SSMs, on the Details tab, be sure leave the Base64 Decoding attribute box unchecked, which is the default setting.

4. Deploy the SSM configurations to the SSMs.

For instructions on how to perform the above steps, see the Console Help.

## Configuring Web Server SSM Properties

The Web Server SSM has a configuration file named `default.properties`. All configuration settings for the Web Server SSM instance are defined in this file. This file is pre-configured and placed in the proper location for you.

If you want to edit the `default.properties` file for your particular environment, refer to the parameters descriptions in the following sections:

- [“Session Settings” on page 5-28](#)
- [“Authentication Settings” on page 5-29](#)
- [“Role Mapping Settings” on page 5-34](#)
- [“Credential Mapping Settings” on page 5-35](#)
- [“Naming Authority Settings” on page 5-36](#)
- [“Logging Level Setting” on page 5-37](#)
- [“Environment Variables Accessible Using CGI” on page 5-37](#)

## Session Settings

The AquaLogic Enterprise Security services are stateless services; it is the calling Web Services client that is responsible for determining session related information. In addition, in a web environment, a session does not necessarily end with an explicit logout, so session termination must be inferred from a lack of activity.

[Table 5-4](#) describes the settings used to manage session behavior. You use these settings to configure the Web server session related behavior for the security configuration to which it applies.

**Table 5-4 Session Settings**

Session Setting	Description
<code>session.inactivity.timeout</code>	The number of seconds of inactivity that causes a session to expire. Default value: 600 seconds (10 minutes)
<code>session.absolute.timeout</code>	The number of seconds an active session is allowed to be available before it expires and the user is forced to re-authenticate. If this setting is set to zero, then established active sessions can continue indefinitely. Default value: 3600 seconds (60 minutes)
<code>session.cookie.name</code>	The name of the session cookie. Default value: ALESIdentityAssertion.
<code>session.forcedlogoffURL</code>	The name of the URL that, when accessed, forces the session to logoff.

## Authentication Settings

[Table 5-5](#) describes the settings that you use to configure the Web server authentication behavior for the security configuration to which it applies. Also, for information on mapping JAAS Callbacks, see [“Mapping JAAS Callback Type to Form and Form Fields”](#) on page 5-31.

**Table 5-5 Authentication Settings**

Authentication Setting	Description
<code>authentication.precedence</code>	An ordered, comma-separated list of types of identity creation. If identity information is available from multiple types of identity transfers, this list determines which identity to use. The valid identity type is: <ul style="list-style-type: none"> <li>• FORM—credential information collected from an authentication provider using forms.</li> </ul> Default value: FORM
<code>authentication.initialForm</code>	Specifies the first form presented for form-based authentication.

**Table 5-5 Authentication Settings (Continued)**

Authentication Setting	Description
authentication. <callback type> [<prompt>] = <field>, <form URL>	Given a question, this setting specifies what field on what form will answer that question. Notice that the <prompt> is shown as optional. However, the prompt is required if there are multiple callbacks of the same type, because there is no other way for the SSM to distinguish identical callback types. The prompt is obtained from the callback by calling the <code>getPrompt()</code> method, but it is not used in the display of the form. If the prompt setting is missing, then the Web Server SSM attempts to answer the callbacks in the order of the settings. If the order does not match the order of the providers, then authentication fails. For more information on using this setting, see <a href="#">“Mapping JAAS Callback Type to Form and Form Fields”</a> on page 5-31.
authentication.onatnfailure	If authentication fails, and this setting is set to a URL, then rather than issuing a 401 Authentication Failed, the user will be redirected to the specified URL.
authentication.onatzfailure	If authorization fails and this setting is set to a URL, then rather than issuing a 403 Permission Denied, the user is redirected to the specified URL.

Table 5-6 describes the different types of authentication callbacks that are supported by the Web Server SSM.

**Table 5-6 Authentication Callback Type Descriptions**

Authentication Callback Type	Description
authentication. nameCallback	The form template responsible for collecting a name for a name callback. This form must exist in the same directory as the post handler.
authentication. passwordCallback	The form template is responsible for collecting a password for a password callback. This form must exist in the same directory as the post handler.
authentication. choiceCallback	The form template is responsible for collecting a choice for a choice callback. This form must exist in the same directory as the post handler.

**Table 5-6 Authentication Callback Type Descriptions**

Authentication Callback Type	Description
authentication. confirmationCallback	The form template is responsible for collecting a confirmation for a confirmation callback. This form must exist in the same directory as the POST handler.
authentication. textInputCallback	The form template is responsible for collecting some text input for a text input callback. This form must exist in the same directory as the post handler.

## Mapping JAAS Callback Type to Form and Form Fields

There are two required and one optional configuration setting that specify what form and what field contain the information required to satisfy the authentication callbacks. The credential gathering form must use an HTTP `POST` method to specify this information. [Listing 5-1](#) shows an example of how to use the `POST` method in the credential gathering form.

**Listing 5-1 Example of Using the POST Method in the Credential Gathering Form**

```
<FORM METHOD=POST ACTION="LoginNamePwdTextIn.html">
<!--#AUTHSTATE -->
<TABLE BGCOLOR="#C0C0C0"><TR><TD>
<TABLE BGCOLOR="#FFFFFF">
<TR><TD COLSPAN="2" BGCOLOR="#C0C0C0">Please Login</TD></TR>
<TR><TD COLSPAN="2">User Name&nbsp;   </TD><TR>
<TR><TD><!--#PROMPT --></TD><TD><INPUT NAME="username"></TD></TR>
<TR><TD COLSPAN="2">Password&nbsp;  </TD><TR>
<TR><TD><!--#PROMPT.1--></TD><TD><INPUT TYPE=
    PASSWORD NAME="password"></TD></TR>
<TR><TD COLSPAN="2">Input Text&nbsp;  </TD><TR>
<TR><TD><!--#PROMPT --></TD><TD><INPUT NAME="textinput"></TD></TR>
<TR><TD COLSPAN="2">&nbsp;  </TD><TR>
<TR><TD COLSPAN="2" ALIGN="CENTER"><INPUT TYPE="SUBMIT"
VALUE="OK"></TD><TR>
</TABLE>
</TD></TR></TABLE>
</FORM>
```

The form field defines the HTTP POST data name that results from a submitted form.

The settings have the following format:

```
authentication.<callback type>[<prompt>] = <field>:<form URL>
```

Given a question, this setting specifies what field on what form will answer that question. Notice that the `<prompt>` is shown as optional. However, if there are multiple callbacks of the same type, the `<prompt>` is required because there is no other way for the Web Server SSM to distinguish identical callback types. The `<prompt>` is obtained from the callback by calling the `getPrompt()` method, but it is not used in the display of the form. If the `<prompt>` setting is missing, then the Web Server SSM attempts to answer the callbacks in the order of the settings. If the order does not match the order of the authentication providers, then authentication fails.

The supported callback types are: `nameCallback`, `passwordCallback`, `textInputCallback`, `textOutputCallback`.

[Table 5-7](#) provides examples of callback usage and more information on each supported callback type.

**Table 5-7 Authentication Callback Usage Examples**

<b>Authentication Callback Types</b>	<b>Example/Discussion</b>
Name and password callbacks	<pre> authentication.nameCallback[]=username: /ales/NamePasswordForm.htm  authentication.passwordCallback[]=password: /ales/NamePasswordForm.htm </pre>
Name, password, and textInput callbacks	<pre> authentication.initialForm=/test/NamePasswordForm.html # username/password authentication.nameCallback[]=username:/test/ NamePasswordForm.html authentication.passwordCallback[]=password:/test/ NamePasswordForm.html  # username/password/textInput authentication.nameCallback[]=username:/test/ LoginNamePwdTextIn.html authentication.passwordCallback[]=password:/test/ LoginNamePwdTextIn.html authentication.textInputCallback[]=textInput:/test/ LoginNamePwdTextIn.html </pre> <p data-bbox="489 1069 1228 1199">In this example the user will be prompted for username/password. The authentication provider then prompts for the user's mother's maiden name. The Web Server SSM redirects to <code>QuestionForm.htm</code> and knows from what field to get the information.</p>

**Table 5-7 Authentication Callback Usage Examples (Continued)**

Authentication Callback Types	Example/Discussion
<p>Name, password, and textInput callbacks</p>	<pre>authentication.nameCallback[]=username: /ales/NamePasswordForm.htm  authentication.passwordCallback []= password: /ales/NamePasswordForm.htm  authentication.textInputCallback ["maiden name"]=maiden_name: /ales/ QuestionForm.htm  authentication.textInputCallback ["social security number"]=maiden_name: /ales/ QuestionForm.htm</pre> <p>In this example two providers require username/password callbacks, a third provider requires a textInputCallback for mother's maiden name, and a fourth provider requires a textInputCallback for a Social Security number: The prompts distinguish between the two textInputCallbacks.</p> <p><b>Note:</b> The textInputCallback prompt requires quotes only if it contains embedded strings.</p>
<p>TextOutput Callback</p>	<p>The textOutputCallback is used to display a message to the user. Because the Web Server SSM does not create or update forms, if it gets a textOutputCallback, it redirects it to the form URL and adds the field as a query string argument and the message value. The application that processes the URL is responsible for parsing the query string and displaying the message.</p>
<p>Language callback</p>	<p>Language callbacks are handled internally by the Web server; the user is never prompted, so no configuration is needed. The user's browser Accept-Language header is checked for the preferred language it supports and that locale is returned to the authentication provider. If the user's browser has no Accept-Language header, the system default locale is used.</p>

## Role Mapping Settings

Table 5-8 describes the settings that you use to configure the Web server role mapping behavior for the policy domain to which it applies.

**Table 5-8 Role Mapping Settings**

Role Mapping Setting	Description
<code>rolemapping.enable</code>	If set to <code>true</code> , then roles are injected into the request stream as a comma separated list.
<code>rolemapping.name</code>	The name of the variable in which to put the roles. The default is: <code>ALES_ROLES</code> .

## Credential Mapping Settings

[Table 5-9](#) describes the settings that you use to configure the Web server credential mapping behavior for the policy domain to which it applies.

**Table 5-9 Credential Mapping Settings**

Credential Mapping Setting	Description
<code>credentialmapping.enable</code>	If set to <code>true</code> , then credentials for each request are injected into the request stream.
<code>credentialmapping.credtypes</code>	<p>List of credential types to ask for in this policy domain. Only credentials that are mapped and that are supported by configured Credential Mapping provider are returned for a specific request. Therefore, asking for a credential does not guarantee that it is there.</p> <p>For example, to configure credential mapping to support the password for the database server, perform the following steps:</p> <ul style="list-style-type: none"> <li>• Set <code>credentialmapping.credtypes</code> to: "<code>credentialmapping.credtypes=DBPASSWORD</code>"</li> <li>• On the Details tab of the Database Credential Mapping provider in the Web Services SSM, set the <code>Allowed Types</code> parameter to <code>DBPASSWORD</code>.</li> </ul> <p><b>Note:</b> The Database Credential Mapper provider provides identity credentials. An identity credential is the same as a <code>PasswordCredential</code> in Java. Others credentials, such as SAML assertions, ALES Identity Assertions IA, and so on, are identity assertions. They are the same as a <code>GenericCredential</code> in Java. The Web Services SSM can have only one identity credential defined, but many identity assertions.</p>
<code>credentialmapping.prefix</code>	Prefix to prepend to credential names, for example <code>CRED</code> .

## Naming Authority Settings

[Table 5-10](#) describes the settings that you use to configure the Web Server SSM naming authority.

**Table 5-10 Naming Authority Settings**

Setting	Description
<code>namingauthority.resource</code>	Specifies the naming authority for the resource. The naming authority is configured in the Web Services SSM.
<code>namingauthority.action</code>	Specifies the action naming authority.
<code>namingauthority.audit</code>	Specifies the audit naming authority.
<code>webservice.registry.url</code>	Specifies the URL of the Web Services Registry Service. For example: <code>http://localhost:8000/ServiceRegistry</code>

## Logging Level Setting

[Table 5-11](#) describes the settings that you use to configure the Web Server SSM naming authority.

**Table 5-11 Logging Level Setting**

Setting	Description
<code>log.level</code>	Specifies the logging level for the log4j Auditing provider.

## Environment Variables Accessible Using CGI

The Web Server Security Service Module (SSM) tool kit enables you to access user environment variables using Common Gateway Interface (CGI).

Although security is embedded within the web server itself, requiring no special programming (if the user does not have access, your code will never run), a security administrator may want to use CGI to access and modify environment variables passed in by the Web Server Security Service Module. In order to customize the application according to the details of the security being enforced, a web application may access several environmental values in order to provide a more integrated user experience.

You can use CGI to access the following environment variables:

- **ALES\_IDENTITY**—An authentication environment variable. It is available to a CGI programmer after a user successfully authenticates. This variable contains the username of the user, if available. It specifies the name of the HTTP header that will be added. The value of the variable is a list of the identity principals, including username and groups.
- **ALES\_DECISIONTIME**—An authorization environment variable. It is available to a CGI programmer after a user is authorized to access a secure resource. It contains the date and time this authorization decision was rendered and has this format: “Monday June 23 15:14:21 EDT 2003”
- **ALES\_ROLES**—A role environment variable that stores a list of roles calculated for the user.
- **Credential Environment Variable**—[Table 5-12](#) describes the credential that is injected into the request stream when the user is authenticated. A CGI application can use this variable to access an LDAP store or database with an appropriate credential, rather than hard coding usernames and passwords. The prefix to this credential variable is configurable, although **CRED** is the default. Different credential types are handled differently, but the general format of the variable is: `CRED_{NAME}={VALUE}`

**Table 5-12 Credential Environment Variables**

Environment Variable	Description
Password Credentials	<p>Password credentials conform to the format <code>javax.resource.spi.security.PasswordCredential</code>. The <code>ManagedConnectionFactory</code> element of this class is ignored. This credential type is rendered in the CGI environment as:</p> <pre>{PREFIX}_PASSWORD={NAME} : {PASSWORD}</pre> <p>where <b>PREFIX</b> is the configured prefix, <b>NAME</b> is the username, and <b>PASSWORD</b> is the password as a string. This name must match the requested credential type from <code>credentialmapping.credtypes</code>.</p> <p>For example:</p> <pre>CRED_PASSWORD=system:weblogic</pre>

# Configuring the Web Services SSM

This section describes how to configure and use a Web Services Security Service Module (WS-SSM), after you have completed the installation and post-installation procedures described in *Installing Security Service Modules*. It includes the following sections:

- “Overview of the Web Services SSM” on page 6-1
- “Configuring and Deploying Policy for the Web Services SSM” on page 6-4
- “Binding the Web Services SSM to a Web Services Client” on page 6-4
- “Configuring SSL in the Web Services SSM” on page 6-4
- “Adding New Identity Assertion Types” on page 6-9

## Overview of the Web Services SSM

The Web Services SSM provides an application programming interface (API) that allows security developers to write custom application clients that invoke AquaLogic Enterprise Security services through SOAP. These interfaces support the most commonly required security functions, such as authentication, authorization, and auditing, and are organized into services that are logically grouped by functionality. A Web Services client can use the Web Services SSM (which incorporates the Security Services APIs, the Security Framework, and the configured security providers) to make access control decisions for the web server to which it is connected. Then you can use the AquaLogic Enterprise Security Administration Server to configure and deploy a security configuration to protect the web server application resources. Thus, the Web

Services SSM enables security administrators and web developers to perform security tasks for applications running on a web server.

The Web Services SSM includes a set of examples that illustrate Web Services client development in different environments. The examples are located in

`BEA_HOME/a1es22-ssm/examples`:

### **ssmWorkshop**

Demonstrates how to access the ALES Web Services SSM through its published WSDL in a WebLogic Workshop 8.1 or 9.x environment.

### **ssmNET**

Demonstrates how to access the ALES Web Services SSM through its published WSDL in the .NET 1.1 or 2.0 environment.

### **javaWebServiceClient**

Demonstrates a simple Java client that accesses the ALES Web Services SSM for authorization.

### **XACMLClient**

Demonstrates how to access the ALES Web Services SSM using the XACML protocol.

**Note:** For a Web Services client developed on Axis, use Axis 1.2 or later. For more information, see the Apache Axis site: <http://ws.apache.org/axis/>.

For more information about developing security services for Web Services client applications, see *Programming Security for Web Services*.

## **Web Services Security Service APIs**

The Web Services Security Service APIs enable access to the ALES security framework. These APIs provide the following security services:

**Note:** The following topics provide a very brief description of these APIs. For more information, see *Programming Security for Web Services*.

- “Authentication Service” on page 6-3
- “Authorization Service” on page 6-3
- “Auditing Service” on page 6-3

- “Role Mapping Service” on page 6-4
- “Credential Service” on page 6-4

## Authentication Service

There are two variations of authentication, JAAS-based and identity assertion. JAAS-based authentication collects evidence (credentials) from a user in order to establish user identity.

**Note:** For more information on JAAS, see Sun’s documentation at <http://java.sun.com/products/jaas/>.

Identity assertion authentication consumes a trusted token object to establish identity. The Web Services SSM supports both types of authentication.

- JAAS authentication is highly variable and is dependent upon the configured authentication provider. An authentication provider can use several different types of questions (modeled as callbacks) to collect information from the user.

**Note:** The Web Services SSM does not support custom callback types.

- Identity assertion authentication is linked to a specific protocol. For example, X.509 certificate assertion is only valid within the context of a 2-way SSL handshake, and SAML identity assertion is only valid in the context of an Oasis SAML profile. The Web Services SSM implements the Oasis SAML Browser/POST profile and consumes or produces a SAML Identity Assertion.

## Authorization Service

In addition to providing a simple permit or denied decision on a URL, the authorization service also has the ability to return attributes into the request as determined by the access control policy implemented. Because the inclusion of coding in the application to handle these attributes creates an undue coupling between the application and security infrastructure, the SSM inserts these returned attributes into the HTTP request header. Depending upon the technology used (ASP, CGI, ISAPI), these headers can be extracted and used by the application.

## Auditing Service

The auditing service audits all transactions through the security subsystem. Every URL accessed is sent through the auditing infrastructure.

## Role Mapping Service

Although roles are primarily used in authorization, some applications may wish to have access to the roles to which a user is mapped for the purposes of role-based personalization. In order to provide this information to the running applications, the Web Services SSM adds a list of roles to the HTTP request header. Depending upon the technology used (ASP, CGI, ISAPI), the application can extract this list of roles from the header and use it.

## Credential Service

The credential service returns sensitive credentials to an application so that the application can use systems that require a secondary (or tertiary) layer of authentication. The Web Services SSM extracts mapped credentials from the security system and makes them available in the HTTP header for use by the application. Depending upon the technology used (ASP, CGI, ISAPI), the application can extract the credential headers and use them to authenticate to other back-end systems.

# Configuring and Deploying Policy for the Web Services SSM

You configure and deploy policy on a Web Services SSM in the same way as you would on an IIS Web Server SSM or an Apache Web Server SSM. For information about the Web Server SSM procedures, see [“Configuring and Deploying Policy for the Web Server SSM” on page 5-11](#).

## Binding the Web Services SSM to a Web Services Client

The Web Services SSM can be used to protect application resources on customer designed and implemented Web Services clients. The Web Services Application Programming Interface (API) is provided for this purpose. For a description of the Web Services API, see [Programming Security for Web Services](#).

## Configuring SSL in the Web Services SSM

When you create a new WS-SSM instance, the WS-SSM is accessible via the HTTP protocol. The protocol is best for development and for debugging purposes, but should not be used in a production environment. For a production environment, BEA recommends that you use either one-way SSL or two-way SSL (SSL with client authentication).

This section describes how to enable one-way and two-way SSL communication between a Web Services SSM and its client. It is assumed that the reader has basic knowledge of the SSL protocol, Certificate Authorities (CA), X.509 certificates and Java Key Stores (JKS).

In this section, `%SSM_INST_HOME%` represents the installation folder of the WS-SSM instance in the file system, for example, `c:\bea\ales22-ssm\web-service-ssm\instance\wssm`

## Configuring One-Way SSL

When you configure a WS-SSM to use one-way SSL communication, the WS-SSM sends its identity certificate, and the CA that signed the identity certificate must be trusted by the client. The WS-SSM does not authenticate the client; thus, the client does not have to have its own certificate.

To configure a WS-SSM to use one-way SSL:

1. Stop the WS-SSM if it is running
2. Delete the contents of the `%SSM_INST_HOME%\apps` directory.
3. Run the following command to regenerate the content of the `apps` directory:

```
%SSM_INST_HOME%\adm\ssmwsInstance.bat -h
```

4. Restart the WS-SSM.

The JKS of the certificates (identities) and the alias that uniquely identifies the identity within the store is defined by the `<identity>` XML element in the

`%SSM_INST_HOME%\apps\ssmws-asi\SAR-INF\config.xml` file. By default, the key store file is `%SSM_INST_HOME%\ssl\identity.jks` and the alias is `wles-ssm`.

**WARNING:** Do not delete the `identity.jks` file, as it is used by other ALES servers to authenticate the WS-SSM. However, you can add any number of additional certificates to that file and reference them by unique aliases.

To access the WS-SSM using SSL, the client must use the HTTPS protocol, rather than HTTP. The client must also be configured to trust the server's certificate; otherwise the connection will be closed. This requires that the CA that signed the server's certificate must be in the client's trusted JKS.

The client's trusted JKS is defined by the system property `javax.net.ssl.trustStore` and the JKS password is defined by the system property `javax.net.ssl.trustStorePassword` property. (These properties are defined in the [Java Secure Socket Extension \(JSSE\)](#))

documentation.) You can specify these system properties by running the WS-SSM Java client with command line arguments such as:

```
-Djavax.net.ssl.trustStore=C:\jks\trust.jks
```

```
-Djavax.net.ssl.trustStorePassword="secretword"
```

For testing purposes, the client can use the `%SSM_INST_HOME%\ssl\trust.jks` JKS, which contains the CA that was used to sign the default server's identity.

## Configuring Two-Way SSL

In two-way SSL (SSL with client authentication), the WS-SSM and the client must each present the other with a trusted certificate. Configuring two-way SSL is similar to configuring one-way SSL, but additional steps have to be taken to allow the WS-SSM server to trust the client.

### Configuring a WS-SSM for Two-Way SSL

To configure the WS-SSM for two-way SSL communication:

1. Add the CA that signed the client's certificate to the trusted CA list of the WS-SSM. By default this list is the `%SSM_INST_HOME%/ssl/trust.jks` file.

The WS-SSM's JKS for trusted CAs is defined by the `<trust>` element in the `%SSM_INST_HOME%\apps\ssmws-asi\SAR-INF\config.xml` configuration file. You can specify the filter for the alias prefixes inside the trusted JKS using the `aliasPrefix` XML attribute. If you specify an `aliasPrefix` attribute, then only the aliases that start with the given prefix will be used. By default the prefix of the alias must start with `domain(<YOUR_ALES_DOMAIN>)`. You can use one of these two approaches when adding a new trusted CA certificate:

- Give the certificate an alias that starts with the prefix `domain(<YOUR_ALES_DOMAIN>)`.
  - As an alternative, give the certificate any other alias and modify the `config.xml` file to remove any `aliasPrefix` XML attribute. If there is no `aliasPrefix` attribute, then the WS-SSM trusts all CAs stored in the `trust.jks`.
2. Add the client's identity to the WS-SSM's trusted peer list. By default this list is the `%SSM_INST_HOME%/ssl/peer.jks` file. The JKS location is determined by the `<peer>` XML element inside the `%SSM_INST_HOME%\apps\ssmws-asi\SAR-INF\config.xml` file.
  3. Stop the WS-SSM if it is running
  4. Delete the contents of the `%SSM_INST_HOME%\apps` directory.

5. Run the following command to regenerate the content of the `apps` directory:

```
%SSM_INST_HOME%\adm\ssmwsInstance.bat -s
```

6. Restart the WS-SSM.

**WARNING:** Do not delete the `trust.jks` or `peer.jks` files, as they are used for communication between the WS-SSM and other ALES servers. The best practice is to add new certificates to the same files and reference them by unique aliases.

## Configuring a Web Services Client for Two-Way SSL

In contrast to one-way SSL, in the case of two-way SSL, the client has to supply its certificate (identity) upon the server's request. The JKS location that stores the identity is defined by the `javax.net.ssl.keyStore` system property. The password for decrypting the identity is defined by the `javax.net.ssl.keyStorePassword` system property. For more information, see the [Java Secure Socket Extension \(JSSE\)](#) documentation.

For development and testing purposes, you can use the `%SSM_INST_HOME%\ssl\identity.jks` key store, as it contains the certificate under the `wles-ssm` alias that it trusted by the server in its default configuration; moreover, the CA that signed the certificate is also trusted. You should never use this approach in production.

The steps necessary to create and use a Web Services client with two-way SSL can vary; the final goal is to create a JKS file that contains the right certificate chain and the private key. This section describes an example of how set up a Web Services client for two-way SSL. In the example, we use the `keytool` utility is used that is shipped with the standard Java Development Kit (JDK). For complete information about this utility, consult the Sun Microsystems [keytool documentation](#).

To create the client's identity store.

1. Create your private/public key pair, self-signed certificate and a new JKS to store it.

```
keytool -genkey -dname "cn=WS-SSM Client" -alias "WS-SSM Client"
-keystore clientkeystore.jks -validity 365
```

2. Create Certificate Signing Request:

```
keytool -certreq -alias "WS-SSM Client" -file WS-SSM-Client.csr -keystore
clientkeystore.jks
```

3. Send the `WS-SSM-Client.csr` file to a CA for signing or sign it by your own CA.
4. Import the CA's certificate (`CA.crt`):

## Configuring the Web Services SSM

```
keytool -import -file CA.crt -alias Trusted-CA -keystore
clientkeystore.jks
```

### 5. Import the signed Certificate Reply from the CA (WS-SSM-Client.crt):

```
keytool -import -file WS-SSM-Client.crt -keystore clientkeystore.jks
-alias "WS-SSM Client"
```

### 6. Delete the CA's certificate, which is no longer needed:

```
keytool -delete -alias Trusted-CA -keystore clientkeystore.jks
```

You can test the key store by executing the following command:

```
keytool -list -keystore clientkeystore.jks -v
```

The output should be similar to the output in [Listing 6-1](#):

### Listing 6-1 Example Keytool List Output

---

```
Keystore type: jks
Keystore provider: SUN
Your keystore contains 1 entry
Alias name: ws-ssm client
Creation date: Apr 14, 2006
Entry type: keyEntry
Certificate chain length: 2
Certificate[1]:
Owner: CN=WS-SSM Client
Issuer: CN=exampleCN, OU=exampleOU, O="exampleO", L=San Exemplo,
ST=California, C=US
Serial number: 1
Valid from: Fri Apr 14 12:45:11 EDT 2006 until: Sat Apr 14 12:45:11 EDT 2007
Certificate fingerprints:
    MD5: BD:2F:C4:9E:27:CB:3A:8F:D4:8E:FB:EA:4E:86:6E:9C
    SHA1: 11:56:D9:94:A0:E2:9B:BC:AD:EF:FD:83:0A:39:5F:0C:0A:B0:9D:22
Certificate[2]:
Owner: CN=exampleCN, OU=exampleOU, O="exampleO", L=San Exemplo,
ST=California, C=US
Issuer: CN=exampleCN, OU=exampleOU, O="exampleO", L=San Exemplo,
ST=California, C=US
Serial number: -7abad5cc20db248d7901d4359b06dbb5
```

```
Valid from: Thu Mar 09 18:42:33 EST 2006 until: Sat Mar 10 18:42:33 EST 2018
Certificate fingerprints:
    MD5:  CF:F8:EA:64:84:02:6F:AD:C4:2E:2B:4B:AC:20:7B:76
    SHA1: 7C:5A:C1:9F:E5:03:26:7E:D7:50:8A:72:20:24:8A:7E:0F:D8:22:CF
*****
```

At this point, the client can use the `clientkeystore.jks` file as the client's identity key store.

To make the WS-SSM trust the client identity:

1. Import the CA's certificate to the server's trust key store (the `trust.jks` file):

```
keytool -import -file CA.crt -alias Trusted-CA -keystore trust.jks
```

2. Import the client's identity certificate to the server's key store of trusted peers (`peer.jks`)

```
keytool -import -file WS-SSM-Client.crt -alias Trusted-WS-Client
-keystore peer.jks
```

3. In the `%SSM_INST_HOME%\apps\ssmws-asi\SAR-INF\config.xml` configuration file, remove the `aliasPrefix XML` attribute under the `<trust>` element.

Now you should be able to establish two-way SSL communication using the client's key store created in the example.

## Adding New Identity Assertion Types

To add support for new assertion types to the Web Services SSM:

1. Create a new Java class as a holder for the identity assertion. Note that the new holder class must belong to the `com.bea.security.ssmws.credentials` namespace. In this procedure, we use a class named `com.bea.security.ssmws.credentials.TestCredHolderImpl` and a custom identity assertion type named `TestIA` as an example. See [Listing 6-2](#) for an example holder class.
2. Add the JAR file containing the holder class to the Web Services SSM's classpath. To do this, modify the `WLESws.wrapper.conf` configuration file, which is located in `BEA_HOME/ales22-ssm/webservice-ssm/instance-name/config`. For example, if the holder class is contained in a file named `ssmwsCustomAssertion.jar`, add a line like this to `WLESws.wrapper.conf`:

```
wrapper.java.classpath.40=C:/bea/ales22-ssm/webservice-ssm/lib/ssmwsCustomAssertion.jar
```

**Note:** The `wrapper.java.classpath` lines must increment sequentially.

3. Modify the mapping file for incoming messages. Mapping for incoming messages is controlled by the `castor.xml` file in the `BEA_HOME/ales22-ssm/webservice-ssm/lib/com/bea/security/ssmws/soap` directory. Add an entry like the following inside the `<mapping>` XML element:

```
<class name="com.bea.security.ssmws.credentials.TestCredHolderImpl">
  <map-to cst:xml="TestIA" />
  <field name="cookie" type="java.lang.String" >
    <bind-xml node="text"/>
  </field>
</class>
```

4. Modify the mapping file for outgoing messages. Mapping for incoming messages is controlled by the `castor.xml` file in the `BEA_HOME/ales22-ssm/webservice-ssm/lib/com/bea/security/ssmws/credentials` directory. Add an entry like the following inside the `<mapping>` XML element

```
<class name="com.bea.security.ssmws.credentials.TestCredHolderImpl">
  <map-to cst:xml="TestIA"
  cst:ns-uri="http://security.bea.com/ssmws/ssm-soap-types-1.0.xsd" />
  <field name="cookie" type="java.lang.String" >
    <bind-xml node="text"/>
  </field>
</class>
```

5. If you want to log SOAP messages received and sent by the Web Services SSM, modify the `log4j.properties` file in the `BEA_HOME/ales22-ssm/webservice-ssm/instance-name/config` directory. Change this line:

```
log4j.appender.A1.Threshold=ERROR
```

to read instead:

```
log4j.appender.A1.Threshold=DEBUG
```

and add the following entry:

```
log4j.logger.com.bea.security.ssmws.server=DEBUG
```

Now, when you restart the Web Services SSM, it will use the new holder implementation and the mapping entries to convert back and forth between the token's XML and Java representations.

## Listing 6-2 Sample Identity Assertion Holder Class

---

```
public class TestCredHolderImpl implements CredentialHolder
{
```

```
private String m_cookie;
public static final String m_Type = "TestIA";

public void setCookie(String cookie)
{
    m_cookie = cookie;
}

public String getCookie()
{
    return m_cookie;
}

public Object getObject()
{
    return getCookie();
}

public void setObject(Object cred)
{
    setCookie((String)cred);
}

public String getType()
{
    return TestCredentialHolderImpl.m_Type;
}

public String getAsString()
{
    return m_cookie;
}
}
```

## Configuring the Web Services SSM

# Configuring the WebLogic Server 8.1 SSM

This section covers tasks that you must perform after installing and completing the post-installation tasks for the WebLogic Server 8.1 Security Service Module. Note that the WebLogic Server 9.x Security Service Module uses a different security framework from the one used in the WLS 8.1 SSM and therefore has configuration procedures. See [Chapter 8](#), “Configuring the WebLogic Server 9.x SSM” for more information.

The following topics are covered in this section:

- “Location of the WebLogic Server Domain” on page 7-1
- “Modifying the startWebLogic File” on page 7-2
- “Defining Security Properties” on page 7-4
- “Starting and Stopping Processes” on page 7-5
- “Additional Post-Installation Considerations” on page 7-5
- “Protecting a Cluster of WebLogic Servers” on page 7-11

## Location of the WebLogic Server Domain

For the purposes of the example presented here, this document assumes that the WebLogic Server domain is in the following location:

```
BEA_HOME/user_projects/domains/mydomain
```

However, your domain can be in any location you desire. If you want to create a domain, you can use the WebLogic Server Configuration Wizard to create a domain or create it manually. The domain includes a `startWebLogic` file, which you are instructed to modify in [“Modifying the startWebLogic File” on page 7-2](#).

## Modifying the startWebLogic File

The WebLogic startup script does the following:

- Sets environment variables.
- Invokes the `java weblogic.Server` command, which starts a JVM that is configured to run a WebLogic Server instance.

Before you can start a WebLogic Server that uses BEA AquaLogic Enterprise Security, you must edit the `startWebLogic` file that is located in the WebLogic Server domain directory. For example:

```
BEA_HOME/user_projects/domains/mydomain
```

where:

- `user_projects` is the directory where your WebLogic Server user projects are located.
- `domains` is the directory where your WebLogic Server domain instances are located.
- `mydomain` is the name of the WebLogic Server domain instance you are using.

See [Listing 7-1](#) for an example of a modified `startWebLogic` file. To edit the `startWebLogic` file, do the following:

1. Before the `CLASSPATH` is set, add a call to the `set-wls-env` script file in your the `bin` directory for your instance. The `set-wls-env` script sets environment variables that are used in the next steps: `WLES_PRE_CLASSPATH`, `WLES_POST_CLASSPATH` and `WLES_JAVA_OPTIONS`. For example:

```
BEA_HOME/aes22-ssm/wls-ssm/instance/wls-ssm/bin
```

Where:

`aes22-ssm` is the directory where you installed the Security Service Module.

`instance` is the directory where all instances are stored.

`wls-ssm` is the name of the Security Service Module instance you created earlier.

For example, if you created an instance called `myInstance`, the call looks like this:

On Windows:

```
call
"C:\bea\ales22-ssm\wls-ssm\instance\myInstance\bin\set-wls-env.bat"
```

On UNIX:

```
. "/bea/ales22-ssm/wls-ssm/instance/myInstance/bin/set-wls-env.sh"
```

2. Add the following line to the CLASSPATH:

On Windows:

```
%WLES_PRE_CLASSPATH% and %WLES_POST_CLASSPATH%
```

On UNIX:

```
${WLES_PRE_CLASSPATH} and ${WLES_POST_CLASSPATH}
```

3. On Windows, add quotes to %JAVA\_HOME%\bin\java in the weblogic.Server command.

```
"%JAVA_HOME%\bin\java"
```

4. Add the following to the java command that starts WebLogic Server with the weblogic.Server class:

On Windows:

```
%WLES_JAVA_OPTIONS%
```

On UNIX:

```
${WLES_JAVA_OPTIONS}
```

### Listing 7-1 Modifying the startWebLogic.cmd File for Windows

---

...

```
set SERVER_NAME=myserver
```

```
call "C:\BEA_HOME\ales22-ssm\wls-ssm\instance\myInstance\bin\set-wls-env.bat"
```

```
set CLASSPATH=%WLES_PRE_CLASSPATH%;%WEBLOGIC_CLASSPATH%;
```

```
%POINTBASE_CLASSPATH%;%JAVA_HOME%\jre\lib\rt.jar;
```

```
%WL_HOME%\server\lib\webservices.jar;%CLASSPATH%;
```

```
%WLES_POST_CLASSPATH%
```

```
@REM Call WebLogic Server
```

```
echo .
```

```
echo CLASSPATH=%CLASSPATH%
```

```
echo .
```

## Configuring the WebLogic Server 8.1 SSM

```
echo PATH=%PATH%
echo .

echo *****
echo * To start WebLogic Server, use a username and *
echo * password assigned to an admin-level user. For *
echo * server administration, use the WebLogic Server *
echo * console at http:\\[hostname]:[port]\\console *
echo *****

"%JAVA_HOME%\bin\java" %JAVA_VM% %MEM_ARGS% %JAVA_OPTIONS% %WLES_JAVA_OPTIONS%
-Dweblogic.Name=%SERVER_NAME%
-Dweblogic.ProductionModeEnabled=%PRODUCTION_MODE%
-Djava.security.policy="%WL_HOME%\server\lib\weblogic.policy" weblogic.Server

ENDLOCAL
```

---

## Defining Security Properties

You can use the `security.properties` file to set the necessary security properties. To set the security properties, create a `security.properties` file and put it in the WebLogic Server domain directory; for example:

```
BEA_HOME/user_projects/domains/mydomain
```

Include the information shown in [Listing 7-2](#) in the `security.properties` file, where:

- `wles.realm` is set to the value of the Configuration ID entered for the Security Service Module using the ALES Administration Console (see the Console Help).
- `wles.default.realm` must be set to the same value as `wles.realm`.

You may also copy this file from the

```
BEA_HOME/ales22-ssm/wls-ssm/instance/myInstance/config folder.
```

**Note:** The `security.properties` file is not required if you add these parameters to Java Options.

### Listing 7-2 Security.properties File

---

```
wles.realm=ConfigurationID
wles.default.realm=ConfigurationID
```

---

## Starting and Stopping 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 and stop the required processes, see [Starting and Stopping Processes for Security Service Modules](#) in the *Administration and Deployment Guide*.

## Additional Post-Installation Considerations

When using the Database Authentication provider, ASI Authorization provider and ASI Role Mapping provider, refer to the following sections for important information:

- [“Setting the Boot Login for WebLogic Server” on page 7-5](#)
- [“Creating a WebLogic Boot Policy” on page 7-6](#)
- [“Creating a WebLogic Console Policy” on page 7-9](#)
- [“Protecting Resources” on page 7-11](#)

## Setting the Boot Login for WebLogic Server

The WebLogic Server uses the login information contained in the `boot.properties` file to start the server. This file contains a `username` and `password` that must match a username and password in the configured authentication policy. The `boot.properties` file is located in the WebLogic Server domain directory on the machine on which the Security Service Module is installed, for example:

```
BEA_HOME/user_projects/domains/mydomain
```

If you used a username of `system` and a password of `weblogic`, then modify WebLogic Server `boot.properties` in the domain as follows:

```
user = system
password = weblogic
```

The next time you start the WebLogic Server, the username and password you specified are encrypted.

## Creating a WebLogic Boot Policy

Before you can use the ASI Authorization provider with the WebLogic Server, you need to configure a boot policy, and then distribute it to the WebLogic Server Security Service Module. The boot policy allows the user named `system` to start the WebLogic Server instance. If you need instructions on how to perform any of the following tasks, see the Console Help for details. You may also want to refer to the *Policy Managers Guide* for information on how the policy language is constructed and how it appears in the console.

To configure and distribute a boot policy, perform the following tasks:

- [“Creating the User Identity” on page 7-6](#)
- [“Creating Resources for WebLogic Server” on page 7-7](#)
- [“Grant Server Resource to Admin Role” on page 7-7](#)
- [“Grant Admin Role to WebLogic User/Group” on page 7-8](#)
- [“Binding the Resource to the ASI Authorization Provider” on page 7-8](#)
- [“Distributing the Policies to the Security Service Module” on page 7-9](#)

## Creating the User Identity

To create the user identity named `alesusers`, perform these steps:

1. Using the ALES Administration Console, create an Identity directory called `alesusers`.
  - a. Open the Identity folder and click Identity.
  - b. Click New. In the Name text box, enter `alesusers`, and then click OK.
2. Within this directory, create a user named `system` and set the password for `system` to `weblogic`. Replace `system` and `weblogic` with the values used in `boot.properties` file.
  - a. Click Users, click New, enter `system`, and click OK.
  - b. Click Edit, click Set Password, enter `weblogic`, and click OK.
  - c. Click OK.

## Creating Resources for WebLogic Server

Create the following resources below the resource called `policy` for the defined user, `alesusers`:

- `wlserver` as a bound application node.
- `wlserver/shared` as virtual
- `wlserver/shared/svr`

To create these resources using the ALES Administration Console:

1. Click Resources and then click New.
2. In the Name box, type `wlserver`, select Binding from the Type drop-down menu, and then click OK.
3. Select `wlserver` and click Configure.
4. From the Type drop-down menu, select Binding Application, check Distribution Point, and then click OK.
5. Select `wlserver`, click New, enter `shared` in the name box, and then click OK.
6. Select `shared`, click Configure, check Allow Virtual Resources, and then click OK.
7. Select `shared`, click New, enter `svr` in the name box, and then click OK.

## Grant Server Resource to Admin Role

Create the following policy:

```
grant(any, //app/policy/wlserver/shared/svr, //role/Admin) if true;
```

1. Expand the Policy node in the left pane, and click Authorization Policies.
2. In the Authorization Policies page, click New.
3. In the Create Authorization Policy dialog page, select the Privileges tab, select `any` in the Select Privileges from Group list box, and then click Add.
4. Select the Resources tab, expand the `wlserver` and `shared` nodes in the Child Resources list box, select `svr`, and then click Add.
5. Select the Policy Subjects tab, select `Admin` from the Roles List list box, click Add, and click OK.

## Grant Admin Role to WebLogic User/Group

Create the following role mapping policy:

```
grant(//role/Admin, //app/policy/wlserver, //user/alesusers/system/)
    if true;
```

1. Click Role Mapping Policies.
2. In the Role Mapping Policies page, click New.
3. In the Create Role Mapping Policy dialog page, select the Roles tab, select Admin from the Available Roles list box, and click Add.
4. Select the Resources tab, select `wlserver` in the Child Resources list box, and click Add.
5. Select the Policy Subjects tab, select Users from the Select Policy Subjects From: drop-down menu, change the directory to `alesusers`, select `system` from the list box, click Add, and click OK.

## Binding the Resource to the ASI Authorization Provider

To bind the resource `//app/policy/wlserver` to the ASI Authorization provider for this Security Service Module, perform the following steps:

1. Open the Security Configuration and Security Control Manager folders.
2. Open the Security Service Module folder and click Authorization.
3. The Authorization page appears.
4. Click Create a new ASI Authorization Provider.
5. The Edit ASI Authorization Provider page appears.
6. Enter a name for the provider in the Name text box, and then click Create.
7. Click the Details tab, set the Identity Directory to `alesusers`, set the Application Directory Parent to `//app/policy/wlserver`.
8. Click Apply.
9. Click the Bindings tab and select the resource you want to bind to the provider from the Bind drop-down menu, and then click Bind.

## Distributing the Policies to the Security Service Module

Distribute the policies to the WebLogic Server Security Service Module.

For information on how to distribute policies, see the Administration Console help system. Be sure to verify the results of the distribution.

## Creating a WebLogic Console Policy

Before you can login into the WebLogic Server Administration Console, you need to configure a console policy and then distribute it to the WebLogic Server Security Service Module. This is needed if you want to access the WebLogic Server Administration Console.

To configure and distribute a WebLogic Server Administration Console policy, do the following on the AquaLogic Enterprise Security Administration Console:

1. Create the following resource:

```
//app/policy/wlserver/console
```

- a. Click Resources. The Resources page appears.
  - b. Select `wlserver`, click New, enter `console` in the name box, and then click OK.
  - c. Select `console`, click Configure, check Allow Virtual Resources, and then click OK.
2. Create the following resource:

```
//app/policy/wlserver/console/url/console/login/bea_logo.gif
```

The resource represents the BEA logo image at the top-right corner on the login page of the Server Administration Console. To create this resource:

- a. Click Resources. The Resources page appears.
- b. Right-click on resource `//app/policy/wlserver/console` and select Add Resource in the context menu..
- c. In the Create Resource dialog window, give the name `url` to the new resource.
- d. Right-click on the resource `//app/policy/wlserver/console/url` and select Add Resource in the context menu..
- e. In the Create Resource dialog window, give the name `console` to the new resource.
- f. Right-click on the resource `//app/policy/wlserver/console/url/console` and select Add Resource in the context menu..

- g. In the Create Resource dialog window, give the name `login` to the new resource.
  - h. Right-click on the resource `//app/policy/wlserver/console/url/console/login` and select **Add Resource** in the context menu..
  - i. In the Create Resource dialog window, give the name `bea_logo.gif` to the new resource.
3. Create the following authorization policy, which allows a user with role `Admin` to access all the resources associated with the `console` application:

```
grant(any, //app/policy/wlserver/console, //role/Admin) if true;
```

- a. Click **Authorization Policies**.
  - b. In the **Authorization Policies** page, click **New**.
  - c. In the **Create Authorization Policy** dialog page, select the **Privileges** tab, click `any` in the **Select Privileges from Group** list box, and then click **Add**.
  - d. Select the **Resources** tab, expand `wlserver` in the **Child Resources** list box, select `console`, and then click **Add**.
  - e. Select the **Policy Subjects** tab, select `Admin` from the **Roles List** list box, click **Add**, and then click **OK**.
4. Create the following authorization policy, which allows any user to see the BEA logo image at the top-right corner on the login page of the Server Administration Console:

```
grant( //priv/GET,  
//app/policy/wlserver/console/url/console/login/bea_logo.gif,  
//sgrp/alesusers/allusers/) if true;
```

- a. Click **Authorization Policies**.
- b. In the **Authorization Policies** page, click **New**.
- c. In the **Create Authorization Policy** dialog page, select the **Privileges** tab, click `GET` in the **Select Privileges from Group** list box, and then click **Add**.
- d. Select the **Resources** tab, expand `wlserver/console/url/console/login` in the **Child Resources** list box, select `bea_logo.gif`, and then click **Add**.
- e. Select the **Policy Subjects** tab, select `allusers` from the **Groups List** list box, click **Add**, and then click **OK**. Be sure that the selected identity store is `alesusers`.

5. Distribute the policies to the WebLogic Server Security Service Module. For information on how to distribute policy, see the Administration Console's help system. Be sure to verify the results of the distribution.

## Protecting Resources

When you secure an EJB using a WebLogic Server Security Service Module, you must follow these steps if you want to use the AquaLogic Enterprise Security providers instead of the default WebLogic providers.

1. Modify the EJB deployment descriptor (`ejb-jar.xml`) so that the assembly-descriptor does not have any method-permissions set to unchecked or excluded.

If either of these settings is present in the deployment descriptor, then the EJB container enforces them rather than calling into the security subsystem.

2. Set the following system property to true, indicating that the EJB container delegates other security checks to the security subsystem, by adding this line to the `WLES_JAVA_OPTIONS` in the `set-wls-env` script:

```
weblogic.security.fullyDelegateAuthorization=true
```

## Protecting a Cluster of WebLogic Servers

If you want to protect a cluster of WebLogic Servers using AquaLogic Enterprise Security, you must make some addition changes to the security configuration and resource configuration. For information on how to protect cluster of WebLogic Servers, see the following topics:

- [“Security Configuration” on page 7-11](#)
- [“Resource Configuration” on page 7-13](#)
- [“Policy Configuration” on page 7-14](#)

## Security Configuration

[Figure 7-1](#) shows a Security Service Module configuration named `myrealm`, located under a Service Control Manager named `adminconfig` in the AquaLogic Enterprise Security Administration Console. Your actual Security Service Module configuration will vary from this example based on the needs of your WebLogic domain.

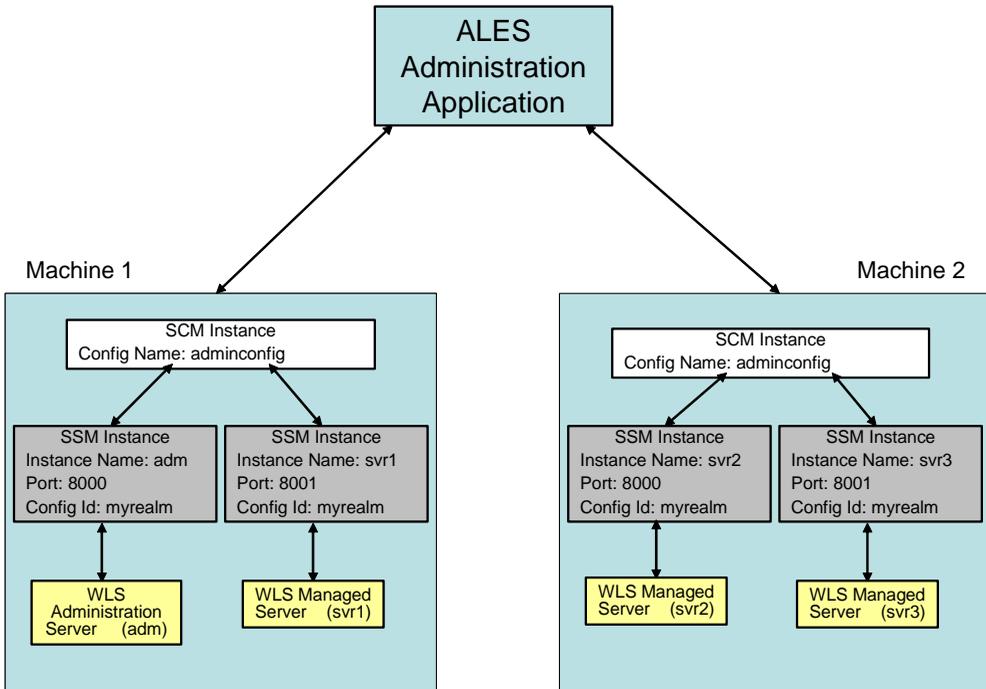
**Figure 7-1 Service Control Manager Configuration**

## Configuring the WebLogic Server 8.1 SSM



Figure 7-2 shows a configuration for a cluster of four WebLogic Servers: one administration server (*adm*) and three managed servers (*svr1*, *svr2*, *svr3*), with one Security Service Module instance for each server. The Service Control Manager on both machines must use the same Configuration Name (*adminconfig*). Each Security Service Module must have a unique Instance Name and Port number per machine, but always shares a common Configuration ID (*myrealm*) across all machines. Thus, each server uses the same security provider configuration and receives the same policy.

**Figure 7-2 WebLogic Server Clusters**



## Resource Configuration

You must also create the following three resources shown in [Figure 7-3](#), setting them each as virtual resources.

The `myrealm/wl_management_internal1` resource is accessed on the cluster's administration server by the WebLogic Admin Console to view WebLogic Server related log files.

The `myrealm/wl_management_internal2` resource is accessed on the cluster's administration server by a managed server during bootstrap and file distribution operations.

The `myrealm/bea_wls_internal` resource is accessed when one managed server is synchronizing with another managed server.

The `myrealm/wl_management_internal1`, `myrealm/wl_management_internal2` and `myrealm/bea_wls_internal` resources must be configured to allow virtual resources.

**Figure 7-3 Resources for Managing WebLogic Server Clusters**



## Policy Configuration

You must create the policy listed in [Table 7-1](#).

**Table 7-1 Policy Configuration**

Privileges	Resources	Policy Subjects	Conditions
any	myrealm/bea_wls_internal	//sgrp/alesusers/allusers/	none
any	myrealm/wl_management_internal1, myrealm/wl_management_internal2	//sgrp/alesusers/allusers/	none

To create this policy in the ALES Administration Console:

1. Click Authorization Policies.
2. On the Authorization Policies page, click New.
3. In the Create Authorization Policy dialog, select the Privileges tab, click any in the Select Privileges from Group list box, and then click Add.
4. Select the Resources tab, expand `myrealm` in the Child Resources list box, select `bea_wls_internal`, and then click Add.

5. Select the Policy Subjects tab, select `allusers` from the Groups List list box, click Add, and then click OK. Be sure that the selected identity store is `alesusers`.
6. On the Authorization Policies page, click New.
7. In the Create Authorization Policy dialog, select the Privileges tab, click any in the Select Privileges from Group list box, and then click Add.
8. Select the Resources tab, expand `myrealm` in the Child Resources list box, select `wl_management_internal1`, and then click Add.
9. Select `wl_management_internal2` also and click Add.
10. Select the Policy Subjects tab, select `allusers` from the Groups List list box, click Add, and then click OK. Be sure that the selected identity store is `alesusers`.

Configuring the WebLogic Server 8.1 SSM

# Configuring the WebLogic Server 9.x SSM

This section covers tasks that you must perform after completing the post-installation tasks for the WebLogic Server 9.x Security Service Module. The following topics are covered in this section:

- [“Overview of the WebLogic Server 9.x SSM” on page 8-1](#)
- [“Configuring the WebLogic Server 9.x SSM: Main Steps” on page 8-2](#)
- [“Configuring Security Providers in the WebLogic Server 9.x SSM” on page 8-3](#)
- [“Modifying the startWebLogic File” on page 8-7](#)

## Overview of the WebLogic Server 9.x SSM

The WebLogic Server 9.x Security Service Module integrates AquaLogic Enterprise Security with BEA WebLogic Server versions 9.1 and 9.2. It uses a different security framework from the one used in the WLS 8.1 SSM and the other ALES SSMs. When you install the WLS 9.x SSM, ALES uses the WLS 9.x security framework. As a consequence, when you use the WLS 9.x SSM, you configure security providers and other aspects of the SSM in the WebLogic Administration Console, rather than the ALES Administration Console. You still use the ALES Administration Console to configure SSMs other than the WLS 9.x SSM and to write security policies for any SSM. You must also use the ALES Administration Console to configure the ASI Authorizer and ASI Role Mapper providers.

## Configuring the WebLogic Server 9.x SSM: Main Steps

To configure the ALES WebLogic Server 9.x SSM:

1. Install WebLogic Server 9.x and create a WebLogic domain.
2. Install the ALES Administration Server and policy and configuration database. See [Installing the Administration Server](#).
3. Install the WebLogic Server 9.x Security Service Module, as described in [Installing Security Service Modules](#).
4. Copy the WLS 9.x console extension for the ALES security providers into the `console-ext` directory of your WebLogic Server domain. See “[Console Extension for Security Providers in the WLS 9.x Console](#)” on page 8-4.
5. Using the WebLogic Server Administration Console, create a new security realm in WebLogic Server. See [Configure new security realms](#) in the WebLogic Server Console Help.
6. Configure security providers in the new WebLogic Server security realm. See “[Configuring Security Providers in the WebLogic Server 9.x SSM](#)” on page 8-3.
7. Make the new security realm the active security realm for WebLogic Server. See [Change the default security realm](#) in the WebLogic Server Console Help.
8. In the ALES Administration Console, create an SSM configuration using the same name as you used for the WLS security realm.
9. In the ALES Administration Console, create an instance of the ASI Authorizer and ASI Role Mapper providers. Set the Identity Directory attribute of the ASI Authorizer and ASI Role Mapper to the same value in the ALES Administration Console and the WebLogic Server Administration Console.
10. In the ALES Administration Console, create the Resource tree.
11. In the ALES Administration Console, create users, groups, attributes and policy.
12. Distribute policy and configuration. The WLS 9 instance must be started after the configuration has been deployed. Policy changes can be deployed while the WLS 9 instance is running.

# Configuring Security Providers in the WebLogic Server 9.x SSM

The WebLogic Server 9.x security framework includes a full set of security providers that are available out of the box. The WLS 9.x security providers are described in the WLS documentation, in the following chapters of *Securing WebLogic Server*:

- [Configuring WebLogic Security Providers](#)
- [Configuring Authentication Providers](#)

In addition, you can use the following ALES security providers by adding them to your WebLogic Server security realm:

- ASI Authorizer
- ASI Role Mapper
- ASI Adjudicator
- ALES Identity Asserter
- Log4J Auditor
- PerfDB Auditor
- Database Authenticator

**Note:** While you can use the WebLogic Server Administration Console to add these ALES security providers to a WebLogic Server security realm and to configure those security providers, the WLS console does not provide online help for the ALES security providers.

See the following topics in this section for detailed information about configuring the WebLogic 9.x SSM:

- [“Console Extension for Security Providers in the WLS 9.x Console”](#) on page 8-4
- [“Configuring a WLS 9.x Security Realm for ALES”](#) on page 8-4
- [“Using the WebLogic Server Console to Configure Security Providers”](#) on page 8-4

## Console Extension for Security Providers in the WLS 9.x Console

ALES includes an extension to the WebLogic Server 9.x Administration Console. If you are using the WLS 9.x SSM for WLS, you must install the console extension in order for the ALES security providers to be visible in the WebLogic Server 9.x Administration Console.

To install the ALES security provider console extension, copy

`ales_security_provider_ext.jar` from `BEA_HOME/ales22-ssm/wls9-ssm/lib` to the `BEA_HOME/WLS_HOME/domains/DOMAIN_NAME/console-ext` directory, where `DOMAIN_NAME` is the name of your WebLogic Server 9.x domain.

## Configuring a WLS 9.x Security Realm for ALES

When you configure a WebLogic 9.x security realm for ALES, you must include at a minimum the following ALES security providers:

- **ASI Authorizer**—In order to take advantage of the ALES Authorization and Role Mapping Engine (ARME), your WLS security realm must include an instance of the ASI Authorizer.
- **ASI Role Mapper**—Your WLS security realm must also include an instance of the ASI Role Mapper in order to take advantage of the ALES Authorization and Role Mapping Engine.
- **Log4J Auditor**—The ALES security providers use a different logging system than the system used by WLS security providers. In order to support logging from any ALES security providers that are present, your WLS security realm must include an instance of the Log4J Auditor.
- **ASI Adjudicator**—If there are multiple authorization providers configured and unanimous permit is false and all authorization providers return ABSTAIN, then the ASI Adjudicator returns false, denying access. The default WLS Adjudicator returns true in the same scenario. Therefore, it is recommended that you use the ASI Adjudicator in order to obtain an appropriate adjudication result.

## Using the WebLogic Server Console to Configure Security Providers

To configure security providers for the WebLogic Server 9.x Security Service Module, you use the WebLogic Server Administration Console, not the ALES Administration Console. In order to create and configure ALES security provider instances using the WebLogic Server

Administration Console, you must first install an extension to the console. See [“Console Extension for Security Providers in the WLS 9.x Console” on page 8-4](#).

To configure security providers for ALES and WebLogic Server 9.x:

1. Log into the WebLogic Server Administration Console.
2. In the Change Center in the upper left corner, click Lock & Edit.
3. In the left panel of the WebLogic Server Administration Console, under Domain Structure, select Security Realms.
4. On the Summary of Security Realms page, select the `myrealm` security realm.
5. On the Configuration: General page:
  - a. Set Security Model Default to Advanced.
  - b. Uncheck Combined Role Mapping Enabled.
  - c. Click Save.
  - d. Click Advanced.
  - e. Set Check Role and Policies to All Web applications and EJBs.
  - f. Click Save.
6. Select the Providers tab. You will configure new authentication, authorization, adjudication, role mapping, and auditing providers for the `myrealm` security realm.
7. On the Providers: Authentication page, configure a new Database Authenticator security provider. To do this:
  - a. Click New.
  - b. Give the new Database Authenticator a name.
  - c. Select Database Authenticator as the Type.
  - d. Click OK.
  - e. Select the new Database Authenticator. On its Configuration: Common page, set the Control Flag to REQUIRED and click Save.
  - f. On the new Database Authenticator’s Configuration: Provider Specific page, set the database login, password, JDBC class name and driver. Click Save.

8. Select the Providers: Authorization page and configure a new ASI Authorization provider:
  - a. Click New.
  - b. Give the new ASI Authorization provider a name.
  - c. Select ASIAuthorizationProvider as the Type.
9. Select the Providers: Adjudication page and configure a new ASI Adjudication provider:
  - a. Click Replace.
  - b. Give the new ASI Adjudication provider a name.
  - c. Select ASIAdjudicator as the Type.
  - d. On the ASIAdjudicator's Configuration: Provider Specific page, uncheck Require Unanimous Permit and click Save.
10. Select the Providers: Role Mapping page and configure a new ASI Role Mapper provider:
  - a. Click New.
  - b. Give the new ASI Role Mapper provider a name.
  - c. Select ASIRoleMapperProvider as the Type.
11. Select the Providers: Auditing page and configure a new Log4j Auditing provider:
  - a. Click New.
  - b. Give the new Log4j Auditing provider a name.
  - c. Select Log4jAuditor as the Type.
12. In the Change Center in the upper left corner, click Activate Changes.
13. Shut down your WebLogic Server instance.

## Modifying the startWebLogic File

The WebLogic Server startup script does the following:

- Sets environment variables.
- Invokes the `java weblogic.Server` command, which starts a JVM that is configured to run a WebLogic Server instance.

Before you can start a WebLogic Server that uses BEA AquaLogic Enterprise Security, you must edit the `startWebLogic` file that is located in the WebLogic Server domain directory. For example:

```
BEA_HOME/user_projects/domains/mydomain
```

where:

- `user_projects` is the directory where your WebLogic Server user projects are located.
- `domains` is the directory where your WebLogic Server domain instances are located.
- `mydomain` is the name of the WebLogic Server domain instance you are using.

See [Listing 8-1](#) for an example of a modified `startWebLogic` file. To edit the `startWebLogic` file, do the following:

1. Before the `CLASSPATH` is set, add a call to the `set-wls-env` script file in your the `bin` directory for your instance. The `set-wls-env` script sets environment variables that are used in the next steps: `WLES_POST_CLASSPATH` and `WLES_JAVA_OPTIONS`. For example:

```
BEA_HOME/ales22-ssm/wls9-ssm/instance/wls-ssm/bin
```

where:

`ales22-ssm` is the directory where you installed the Security Service Module.

`instance` is the directory where all instances are stored.

`wls-ssm` is the name of the Security Service Module instance you created earlier.

For example, if you created a WLS SSM instance called `myInstance`, the call looks like this:

On Windows:

```
call
"C:\bea\ales22-ssm\wls9-ssm\instance\myInstance\bin\set-wls-env.bat"
```

On UNIX:

## Configuring the WebLogic Server 9.x SSM

- ```
. "/bea/ales22-ssm/wls9-ssm/instance/myInstance/bin/set-wls-env.sh"
```
2. Add the following to the CLASSPATH:  
On Windows:  
`%WLES_POST_CLASSPATH%`  
On UNIX:  
`${WLES_POST_CLASSPATH}`
  3. On Windows, add quotes to `%JAVA_HOME%\bin\java` in the `weblogic.Server` command.  
`"%JAVA_HOME%\bin\java"`
  4. Add the following to the `java` command that starts WebLogic Server with the `weblogic.Server` class:  
On Windows:  
`%WLES_JAVA_OPTIONS%`  
On UNIX:  
`${WLES_JAVA_OPTIONS}`

### Listing 8-1 Modifying the startWebLogic.cmd File for Windows

---

```
...  
set SERVER_NAME=myserver  
  
call "C:\BEA_HOME\ales22-ssm\wls9-ssm\instance\myInstance\bin\set-wls-env.bat"  
  
set CLASSPATH=%WEBLOGIC_CLASSPATH%;  
%POINTBASE_CLASSPATH%;%JAVA_HOME%\jre\lib\rt.jar;  
%WL_HOME%\server\lib\webservices.jar;%CLASSPATH%;  
%WLES_POST_CLASSPATH%  
  
@REM Call WebLogic Server  
echo .  
echo CLASSPATH=%CLASSPATH%  
echo .  
echo PATH=%PATH%  
echo .  
  
echo *****  
echo * To start WebLogic Server, use a username and *  
echo * password assigned to an admin-level user. For *
```

```
echo * server administration, use the WebLogic Server *
echo * console at http:\\[hostname]:[port]\\console *
echo *****

"%JAVA_HOME%\bin\java" %JAVA_VM% %MEM_ARGS% %JAVA_OPTIONS% %WLES_JAVA_OPTIONS%
-Dweblogic.Name=%SERVER_NAME%
-Dweblogic.ProductionModeEnabled=%PRODUCTION_MODE%
-Djava.security.policy="%WL_HOME%\server\lib\weblogic.policy" weblogic.Server

ENDLOCAL
```

---

Configuring the WebLogic Server 9.x SSM

# Integrating with WebLogic Portal

This section covers the following topics:

- [“Introduction” on page 9-1](#)
- [“Integration Pre-Requisites” on page 9-5](#)
- [“Integrating with WebLogic Portal 9.2: Main Steps” on page 9-5](#)
- [“Integrating with WebLogic Portal 8.1: Main Steps” on page 9-8](#)
- [“Configuring Policy for the Portal Application” on page 9-14](#)

## Introduction

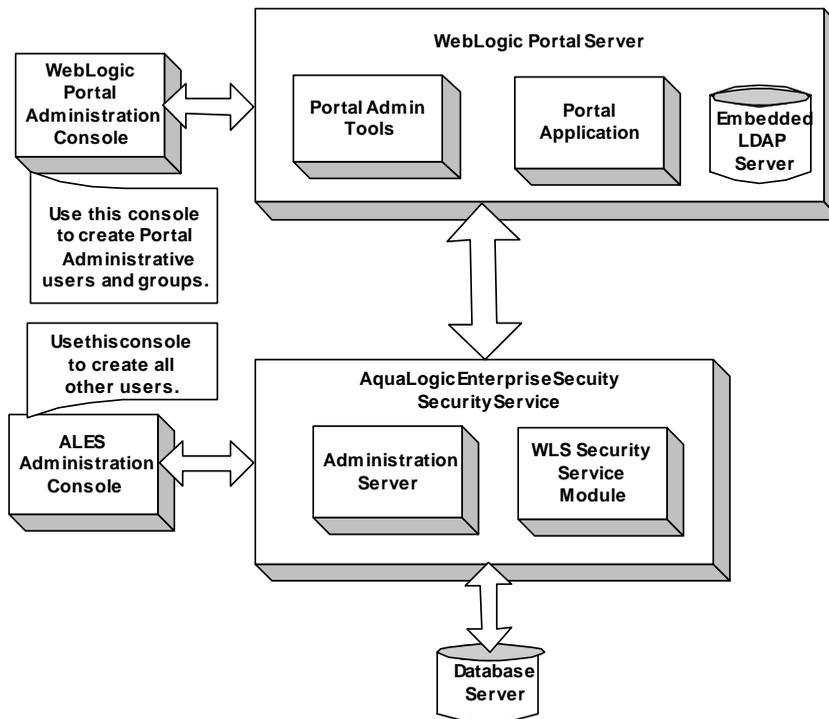
Integrating AquaLogic Enterprise Security with WebLogic Portal server and portal application results in an enhanced set of security services for use in protecting WebLogic Portal (see [Figure 9-1](#)). AquaLogic Enterprise Security participates in the authoring and management of policy for WebLogic Portal resources. Once AquaLogic Enterprise Security is integrated with WebLogic Portal, you use AquaLogic Enterprise Security Administration Server to manage resources related to portal desktops, books, pages, and portlets.

Therefore, the intent is that you use AquaLogic Enterprise Security for authorization of the resources associated with a portal application as well as standard WebLogic Server J2EE resources. The benefit of using AquaLogic Enterprise Security to manage visitor entitlements is that it offers fine-grained, dynamic role-based authorization. Additionally, AquaLogic Enterprise Security allows you to have common security policies for a heterogeneous environment. For

example, you may have a single security infrastructure that supports WebLogic Portal, WebLogic Server, and custom applications.

The AquaLogic Enterprise Security security service does not replace all of the management functionality provided by the Portal Administration Tools. For example, as shown in [Figure 9-1](#), AquaLogic Enterprise Security is not used to manage administrative users and resources associated with Portal Delegated Administration and Portal Content Management; use the Portal Administration Tools for those management tasks.

**Figure 9-1 Portal Integration Overview**



AquaLogic Enterprise Security enables you to write, deploy, and manage fine-grained policy for controlling access to WebLogic Portal application resources. You can use AquaLogic Enterprise Security to protect portal desktops, books, pages, portlets, and application look and feels.

For more information, see the following topics:

- [“Integration Features” on page 9-3](#)
- [“Supported Use-case Scenario” on page 9-3](#)
- [“Constraints and Limitations” on page 9-4](#)

## Integration Features

AquaLogic Enterprise Security and WebLogic Portal can be used with WebLogic Server 9.2 or WebLogic Server 8.1 Service Pack 4 or Service Pack 5. While several different security providers can be used with WebLogic Portal, the following security providers include enhanced WebLogic portal support:

- The ASI Authorization provider—Enables you to use AquaLogic Enterprise Security to write, deploy, and manage fine-grained authorization of WebLogic Portal application resources related to desktops, books, pages, portlets, and application look and feels.
- The Database Authentication provider—Enables user and group controls. This provider is integrated into the Portal Administration Tools environment so as to handle user and group queries.  
**Note:** Use of the Database Authentication provider with WebLogic Portal is not mandatory. You may use other authentication providers as well.
- ASI Role Mapper—Enables dynamic mapping of principals (users and groups) to security roles at runtime. Role Mapping determines which security roles to apply to the principals stored in a subject when the subject is attempting to perform an operation on a portal application resource. Because this operation usually involves gaining access to the portal application resource, the ASI Role Mapper is used in conjunction with the ASI Authorization provider.

## Supported Use-case Scenario

The following use-case scenario is supported when you integrate AquaLogic Enterprise Security with WebLogic Portal:

- The AquaLogic Enterprise Security Administration Server assumes responsibility for management and policy of resources related to Portal visitor entitlements.

- The AquaLogic Enterprise Security Administration Server is responsible for management of J2EE application resources associated with Portal applications and portal administration tools.
- The Portal Administration Tools continue to be responsible for the rest of Portal Management and Administration, including the creation and management of portal administrative users and groups.

**Note:** To implement this use case scenario, you must define the security configuration as specified in [“Creating the Portal Application Security Configuration” on page 9-9.](#)

## Constraints and Limitations

When integrated with AquaLogic Enterprise Security, WebLogic Portal has the following constraints and limitations:

- Portal application administrators will not use the WebLogic Portal Administration Tools to create and manage visitor entitlements.

Use of AquaLogic Enterprise Security with a Portal application implies that an administrator will not use the Portal Administration Tools to create “Visitor Entitlements” on portal desktops, books, pages, and portlets. Managing visitor entitlements from the Portal Administration Tools is not a supported use case.

- Users will not use application deployment descriptors to deploy policy.

Use of Deployment descriptors to deploy policy is not supported in AquaLogic Enterprise Security.

- Migration of existing portal application policy is not supported.

AquaLogic Enterprise Security does not support the migration of visitor entitlements policy for existing portal applications. There are no facilities for migrating any information from the WebLogic Server embedded LDAP store.

- AquaLogic Enterprise Security does not replace or in any way interfere with the use of the Portal Administration Tools for the management of resource structures associated with Portal Delegated Administration and Portal Content Management.

You cannot use AquaLogic Enterprise Security to manage the resources associated with Portal Delegated Administration and Portal Content Management. AquaLogic Enterprise Security does not support Portal Unified User Profiles.

# Integration Pre-Requisites

Before you begin, you must ensure that the following pre-requisites are satisfied:

- The WebLogic Platform/Portal 9.2 or 8.1, with Service Pack 4 or Service Pack 5, must be installed on the local machine.
- The AquaLogic Enterprise Security 2.2 WebLogic Server Security Service Module must be installed on the local machine.
- You must have access to an Administration Console that is running on the AquaLogic Enterprise Security 2.2 Administration Server on either the local machine or a remote machine.
- When using Weblogic Platform/Portal 9.2, you must have access to the WebLogic Server Administration Console to set the security configuration. For Weblogic Platform/Portal 8.1, the security configuration is set using the ALES Administration Console.
- You have created a WebLogic Portal domain on the local machine and installed a portal application in that domain.

## Integrating with WebLogic Portal 9.2: Main Steps

This section describes how to integrate AquaLogic Enterprise Security with WebLogic Portal 9.2. The procedure for integrating ALES with WebLogic Portal 8.1 is different; for information about that, see [“Integrating with WebLogic Portal 8.1: Main Steps” on page 9-8](#). Once integrated, you can use the AquaLogic Enterprise Security Administration Console to write and deploy a set of authorization and role mapping policies to protect WebLogic Portal application resources.

To integrate AquaLogic Enterprise Security with WebLogic Portal 9.2, perform the following tasks:

- [“Creating the Portal Application Security Configuration” on page 9-9](#)
- [“Using the WebLogic Server Console to Configure Security Providers” on page 9-6](#)
- [“Modifying the Portal Server startWeblogic File” on page 9-6](#)

## Creating the Portal Application Security Configuration

This section describes how to create a new security configuration named `myrealm`.

1. Using the AquaLogic Enterprise Security Administration Console, create a security configuration with the Configuration ID `myrealm`.
2. Create instances of the ASI Authorization and ASI Role Mapper providers.
3. Bind the security configuration to a Service Control Manager.
4. Distribute the security configuration.
5. Create an instance of a WebLogic Server 9.x Security Service Module, as described in [“Creating an Instance of a Security Service Module” on page 4-6](#). Set the instance name to `portalinstance` and the Configuration ID to `myrealm`.
6. Enroll the WebLogic Server 9.x Security Service Module instance, as described in [“Enrolling the Instance of the Security Service Module” on page 4-7](#).
7. Use the WebLogic Server Administration Console to finish the security configuration for your WebLogic 9.x SSM. See [“Using the WebLogic Server Console to Configure Security Providers” on page 9-6](#).

## Using the WebLogic Server Console to Configure Security Providers

To configure security providers for the WebLogic Server 9.x Security Service Module, you use the WebLogic Server Administration Console, not the ALES Administration Console. In order to create and configure ALES security provider instances using the WebLogic Server Administration Console, you must first install an extension to the console. See [“Console Extension for Security Providers in the WLS 9.x Console” on page 8-4](#).

For information about how to configure security providers for ALES and WebLogic Portal 9.2, see [“Configuring Security Providers in the WebLogic Server 9.x SSM” on page 8-3](#).

## Modifying the Portal Server `startWeblogic` File

Before you can start a WebLogic Portal server that uses BEA AquaLogic Enterprise Security, you must modify the `startWeblogic` file that is located in the WebLogic Portal domain that you are using for your WebLogic Portal server.

The `startWeblogic` file for the WebLogic Portal sample domain named `portalDomain` is located at: `BEA_HOME\weblogic92\samples\domains\portal`

To edit the `startWeblogic` file, perform the steps:

**Note:** This procedure assumes a Windows installation of WebLogic Portal in the directory `c:\bea` with a WebLogic Server Security Service Module instance named `portalInstance`.

1. Before you modify the script, make sure to make a backup copy. For example, for Microsoft Windows, copy `startWeblogic.cmd` to `startWeblogic.cmd.original`.
2. Add a line to call the environment batch file `set-wls-env.bat`. For example, add it below the line: `set SAVE_JAVA_OPTIONS=`

```
call
"c:\bea\ales22-ssm\wls9-ssm\instance\portalInstance\bin\set-wls-env.bat
"
```
3. Add the AquaLogic Enterprise Security `classpath` variables to the `classpath`. For example, add the following text before the line: `echo CLASSPATH=%CLASSPATH%`

```
set CLASSPATH=%CLASSPATH%;%WLES_POST_CLASSPATH%
```
4. Add `%WLES_JAVA_OPTIONS%` to the server start command after `%JAVA_OPTIONS%`. [Listing 9-2](#) shows, in bold text, where to make this change.

### Listing 9-1 Adding WLES\_JAVA\_OPTIONS to the startWebLogic File

---

```
if "%WLS_REDIRECT_LOG%"==" " (
    echo Starting WLS with line:
    echo %JAVA_HOME%\bin\java %JAVA_VM% %MEM_ARGS% %JAVA_OPTIONS%
%WLES_JAVA_OPTIONS% -Dweblogic.Name=%SERVER_NAME%
-Djava.security.policy=%WL_HOME%\server\lib\weblogic.policy
%PROXY_SETTINGS% %SERVER_CLASS%
    %JAVA_HOME%\bin\java %JAVA_VM% %MEM_ARGS% %JAVA_OPTIONS%
%WLES_JAVA_OPTIONS% -Dweblogic.Name=%SERVER_NAME%
-Djava.security.policy=%WL_HOME%\server\lib\weblogic.policy
%PROXY_SETTINGS% %SERVER_CLASS%
) else (
    echo Redirecting output from WLS window to %WLS_REDIRECT_LOG%
    %JAVA_HOME%\bin\java %JAVA_VM% %MEM_ARGS% %JAVA_OPTIONS%
%WLES_JAVA_OPTIONS% -Dweblogic.Name=%SERVER_NAME%
```

```
-Djava.security.policy=%WL_HOME%\server\lib\weblogic.policy
%PROXY_SETTINGS% %SERVER_CLASS% > "%WLS_REDIRECT_LOG%" 2>&1
)
```

---

## Integrating with WebLogic Portal 8.1: Main Steps

This section describes how to integrate AquaLogic Enterprise Security with WebLogic Portal 8.1. Once integrated, you can use the AquaLogic Enterprise Security Administration Console to write and deploy a set of authorization and role mapping policies to protect WebLogic Portal application resources.

**Note:** While the instructions provided in this section use a WebLogic Portal server and the sample portal application that ships with the WebLogic Platform software distribution, you can use this procedure to integrate AquaLogic Enterprise Security with your WebLogic Portal server and portal application.

To integrate AquaLogic Enterprise Security with WebLogic Portal, perform the following tasks:

- [“Creating the Portal Application Security Configuration” on page 9-9](#)
- [“Binding the Security Configuration” on page 9-10](#)
- [“Distributing the Security Configuration” on page 9-10](#)
- [“Creating an Instance of the Security Service Module” on page 9-10](#)
- [“Enrolling the Instance of the Security Service Module” on page 9-10](#)
- [“Modifying the Portal Server startWeblogic File” on page 9-11](#)
- [“Creating the security.properties File” on page 9-12](#)
- [“Replacing the Portal p13n\\_ejb.jar File” on page 9-12](#)
- [“Replacing the Portal p13n\\_system.jar File” on page 9-13](#)
- [“Replacing the DefaultAuthorizerInit.ldift File” on page 9-14](#)

## Creating the Portal Application Security Configuration

This section describes how to create a new security configuration named `myrealm`. A security configuration defines the set of security providers to use for adjudication, authentication, auditing, authorization, role mapping, and credential mapping services. The security configuration named `myrealm` matches the default security configuration for the WebLogic Portal sample portal application.

**Note:** To implement the use-case scenario described in [“Supported Use-case Scenario” on page 9-3](#), you are required to define the security configuration as described in this section. This security configuration is a requirement; it is not optional.

Refer to [Table 9-1](#) and use the AquaLogic Enterprise Security Administration Server to configure the security providers listed there. Set the Configuration ID to `myrealm`. For instructions on creating a security configuration, see the administration console’s help system.

**Table 9-1 Portal Security Configuration**

| Security Provider                | Configuration Settings                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|----------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ASI Adjudication Provider        | Uncheck the Require Unanimous Permit check box, and click Create.                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Log4j Auditor                    | Accept the default settings, and click Create.                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Database Authentication Provider | <p>Set the Control Flag to <code>SUFFICIENT</code>, and click Create. For the Details tab settings, except for the Identity Scope, the parameters are populated automatically. Set the Identity Scope to <code>myusers</code>, and click Apply.</p> <p><b>Note:</b> Even though you set the Identity Scope to <code>myusers</code>, you do not actually create the <code>myusers</code> identity until you perform the steps in <a href="#">“Creating the Realm Resource” on page 9-16</a>.</p> |
| WebLogic Authentication Provider | <p>Set the Control Flag to <code>SUFFICIENT</code>, and click Create.</p> <p><b>Note:</b> Make sure the authentication providers are configured in the following order: 1) Database Authenticator and 2) WebLogic Authenticator.</p> <p><b>Note:</b> The WebLogic Authentication provider can be replaced with another authentication provider that supports write access to users and groups.</p>                                                                                              |
| ASI Authorization Provider       | <p>On the General tab, accept the default settings, and click Create.</p> <p>On the Details tab, set the Identity Scope to <code>myusers</code>.</p>                                                                                                                                                                                                                                                                                                                                            |

**Table 9-1 Portal Security Configuration (Continued)**

| <b>Security Provider</b>            | <b>Configuration Settings</b>                                                                                                              |
|-------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------|
| WebLogic Authorization Provider     | Uncheck the Policy Deployment Enabled check box, and click Create.                                                                         |
| WebLogic Credential Mapper Provider | Uncheck the Credential Mapping Deployment Enabled check box, and click Create.                                                             |
| ASI Role Mapping Provider           | On the General tab, accept the default settings, and click Create.<br>On the Details tab, set the Identity Scope to <code>myusers</code> . |
| WebLogic Role Mapper Provider       | Uncheck the Role Deployment Enabled check box, and click Create.                                                                           |

## Binding the Security Configuration

The security configuration must be bound to a Service Control Manager.

To bind the `myrealm` security configuration, see the Console Help

## Distributing the Security Configuration

The `myrealm` security configuration must be distributed.

To distribute the `myrealm` security configuration, see the Console Help.

## Creating an Instance of the Security Service Module

Before starting a WebLogic Server Security Service Module, you must first create an instance of the WebLogic Server Security Service Module using the Create New Instance Wizard.

To create an instance of a WebLogic Server Security Service Module, see [“Creating an Instance of a Security Service Module” on page 4-6](#).

## Enrolling the Instance of the Security Service Module

You must have the Administration Server running prior to enrolling the Security Service Module.

**Note:** While you can use the demonstration digital certificate in a development environment, you should never use it in a production environment.

To enroll a security service module, see [“Enrolling the Instance of the Security Service Module” on page 4-7](#).

## Modifying the Portal Server startWeblogic File

Before you can start a WebLogic Portal server that uses BEA AquaLogic Enterprise Security, you must modify the `startWeblogic` file that is located in the WebLogic Portal domain that you are using for you WebLogic Portal server.

The `startWeblogic` file for the WebLogic Portal domain named `portalDomain` is located at: `BEA_HOME\user_projects\domains\portalDomain`

To edit the `startWeblogic` file, perform the steps:

**Note:** This procedure assumes a Windows installation of WebLogic Portal in the directory `c:\bea` with an WebLogic Server Security Service Module instance named `portalInstance`.

1. Before you modify the script, make sure to make a backup copy. For example, for Microsoft Windows, copy `startWeblogic.cmd` to `startWeblogic.cmd.original`.
2. Add a line to call the environment batch file `set-wls-env.bat`. For example, add it below the line: `set SAVE_JAVA_OPTIONS=`

```
call
"c:\bea\ales22-ssm\wls-ssm\instance\portalInstance\bin\set-wls-env.bat"
```
3. Add the AquaLogic Enterprise Security `classpath` variables to the `classpath`. For example, add the following text before the line: `echo CLASSPATH=%CLASSPATH%`

```
set CLASSPATH=%WLES_PRE_CLASSPATH%;%CLASSPATH%;%WLES_POST_CLASSPATH%
```
4. Add `%WLES_JAVA_OPTIONS%` to the server start command after `%JAVA_OPTIONS%`. [Listing 9-2](#) shows, in bold text, where to make this change.

### Listing 9-2 Adding WLES\_JAVA\_OPTIONS to the startWebLogic File

---

```
if "%WLS_REDIRECT_LOG%"==" " (
    echo Starting WLS with line:
    echo %JAVA_HOME%\bin\java %JAVA_VM% %MEM_ARGS% %JAVA_OPTIONS%
%WLES_JAVA_OPTIONS% -Dweblogic.Name=%SERVER_NAME%
-Djava.security.policy=%WL_HOME%\server\lib\weblogic.policy
%PROXY_SETTINGS% %SERVER_CLASS%
    %JAVA_HOME%\bin\java %JAVA_VM% %MEM_ARGS% %JAVA_OPTIONS%
%WLES_JAVA_OPTIONS% -Dweblogic.Name=%SERVER_NAME%
-Djava.security.policy=%WL_HOME%\server\lib\weblogic.policy
```

```
%PROXY_SETTINGS% %SERVER_CLASS%
) else (
    echo Redirecting output from WLS window to %WLS_REDIRECT_LOG%
    %JAVA_HOME%\bin\java %JAVA_VM% %MEM_ARGS% %JAVA_OPTIONS%
%WLES_JAVA_OPTIONS% -Dweblogic.Name=%SERVER_NAME%
-Djava.security.policy=%WL_HOME%\server\lib\weblogic.policy
%PROXY_SETTINGS% %SERVER_CLASS% > "%WLS_REDIRECT_LOG%" 2>&1
)
```

---

## Creating the security.properties File

Create a text file named `security.properties` and place it in the portal domain directory. You use this file to define the AquaLogic Enterprise Security realm and the default realm. [Listing 9-3](#) shows the content of this file for the realm `myrealm`.

### Listing 9-3 security.properties File

---

```
# AquaLogic Enterprise Security Configuration File
#
# This file contains AquaLogic Enterprise Security configuration
# properties. By default, the AquaLogic Enterprise Security runtime
# looks for a property file called 'security.properties' in the
# working directory.
wles.realm=myrealm
wles.default.realm=myrealm
```

---

## Replacing the Portal p13n\_ejb.jar File

To integrate AquaLogic Enterprise Security with WebLogic Portal, you must replace the `p13n_ejb.jar` file in the top-level portal application directory with the version of that file that is provided in the AquaLogic Enterprise Security software distribution. The AquaLogic Enterprise Security version of `p13n_ejb.jar` is located in `BEA_HOME/ales22-ssm/wls-ssm/lib` directory.

**Note:** BEA AquaLogic Enterprise Security 2.2 includes two versions of `p13n_ejb.jar`, the WebLogic Server 8.1 SP4 version: `p13n_ejb_81SP4.jar`, and the SP5 version: `p13n_ejb_81SP5.jar`. Be sure to use the correct version.

**Note:** Because these instructions assume that you are using the sample portal application that ships with WebLogic Portal, this procedure instructs you to replace the `p13n_ejb.jar` in the sample portal application. To use a different portal application, replace `p13n_ejb.jar` in that application as well.

To replace `p13n_ejb.jar`, perform the following steps:

1. Rename the portal version of the `p13n_ejb.jar`. For example, rename it to `p13n_ejb.jar.original`. The portal application version of this file is located in `BEA_HOME/weblogic81/samples/portal/portalApp`.
2. Depending on which version of the WebLogic Server 8.1 you are using (SP4 or SP5), copy either `p13n_ejb_81SP4.jar` or `p13n_ejb_81SP5.jar` from `BEA_HOME/ales22-ssm/wls-ssm/lib/` to `BEA_HOME/weblogic81/samples/portal/portalApp` and rename it to `p13n_ejb.jar`.

## Replacing the Portal `p13n_system.jar` File

To integrate AquaLogic Enterprise Security with WebLogic Portal, you must replace the `p13n_system.jar` file in the `BEA_HOME/weblogic81/p13n/lib` directory with the version of that file that is provided in the AquaLogic Enterprise Security software distribution. The AquaLogic Enterprise Security version of this file is located in `BEA_HOME/ales22-ssm/wls-ssm/lib` directory.

**Note:** BEA AquaLogic Enterprise Security 2.2 includes two versions of `p13n_system.jar`, the WebLogic Server 8.1 SP4 version: `p13n_system_81SP4.jar`, and the SP5 version: `p13n_system_81SP5.jar`. Be sure to use the correct version.

**Note:** Once you replace `p13n_system.jar` in the `/lib` directory of the WebLogic Platform installation, all portal domains configured for that installation must be AquaLogic Enterprise Security enabled.

To replace `p13n_system.jar`, perform the following steps:

1. Rename the portal version of the `p13n_system.jar`. For example, rename it to `p13n_system.jar.original`. The portal version of this file is located in `BEA_HOME/weblogic81/p13n/lib`.
2. Depending on which version of the WebLogic Server 8.1 you are using (SP4 or SP5), copy either `p13n_system_81SP4.jar` or `p13n_system_81SP5.jar` from `BEA_HOME/ales22-ssm/wls-ssm/lib/` to `BEA_HOME/weblogic81/p13n/lib` and rename it to `p13n_system.jar`.

## Replacing the DefaultAuthorizerInit.ldif File

WebLogic Server uses the `DefaultAuthorizerInit.ldif` file to establish access controls for J2EE resources. By default, WebLogic Server allows access to all J2EE resources to users in the `Everyone` role. To protect these resources, WebLogic Server provides the Administration Console and other tools to define security policies.

When using AquaLogic Enterprise Security, there is a need to supersede the WebLogic Server J2EE access controls. The `DefaultAuthorizerInit.ldif` file, provided in the AquaLogic Enterprise Security 2.2 for the WebLogic Server Security Service Module, is used for this purpose.

To enable the AquaLogic Enterprise Security `DefaultAuthorizerInit.ldif` file to supersede WebLogic Server access controls for J2EE resources in the sample portal application, perform the following steps:

1. Copy the `DefaultAuthorizer.ldif` file from  
`BEA_HOME/ales22-ssm\wls-ssm\template\config` to  
`BEA_HOME/user_projects/domains/mydomain`.
2. If the `/ldap` directory exists at the following location, delete it:  
`BEA_HOME/user_projects/domains/portalDomain/portalServer`

**Note:** Because these instructions assume that you are using the sample portal domain that ships with WebLogic Portal, this procedure instructs you to delete the `ldap` directory in the `portalDomain/portalServer` directory. Repeat the above steps for all AquaLogic Enterprise Security enabled portal domains.

## Configuring Policy for the Portal Application

Developing a set of policies typically begins by determining which resources you need to protect and your access control requirements. You then create the identity directory, resources, groups, users, and roles that you will use to write policies to protect those resources. Next you write a set of authorization and role mapping policies to define access control on those resources. Finally, you deploy the set of policies to the WebLogic Server Security Service Module that you use to control access to your portal application resources.

AquaLogic Enterprise Security provides two means for writing portal application policy, the Administration Console and the Policy Import Tool. In this section you are directed to use the Administration Console to write policy. For more information on how to use the Administration Console to write policy, see the *Policy Managers Guide* and the Console Help.

In addition, the ALES Administration Server installation includes a set of sample policies for BEA WebLogic Portal, located at

`BEA_HOME/ales22-admin/examples/policy/portal_sample_policy`. You can import these sample policies and use them as a starting point for developing a full set of policies for your applications. For information about how to import the sample policies, see the README files in each of the sample directories and see also [Importing Policy Data](#) in the *Policy Managers Guide*.

This section covers the following topics:

- [“Creating the Identity Directory and Users” on page 9-15](#)
- [“Configuring Resources and Privilege” on page 9-16](#)
- [“Creating the Role Mapping Policy” on page 9-20](#)
- [“Creating Authorization Policies” on page 9-21](#)
- [“Policy for Visitor Entitlements to Portal Resources” on page 9-25](#)

## Creating the Identity Directory and Users

This section describes how to use the Administration Console to create an identity directory, groups, and users for a portal application.

**Note:** This procedure uses `myusers` as the name of the Identity directory; however, you can use a different name.

To create the Identity directory and users:

1. In the left pane, click Identity. The Identity page displays the name of each directory available.
2. Click New. The Create Directory dialog box appears.
3. In the Name text box, type `myusers` and click OK. The `myusers` directory appears in the list of Identity directories.
4. In the left pane, click Users. The `myusers>Users` page displays.
5. Click New. The Create User dialog box appears.
6. Create the users that will visit your portal application.

If you are using the WebLogic 9.x SSM, create a user with the Admin role to access the WebLogic Server Administration Console. The default WebLogic Server Admin user and password is `weblogic/weblogic`.

# Configuring Resources and Privilege

This section describes how to use the Administration Console to define the portal application resources that you will protect using AquaLogic Enterprise Security.

To configure resources, perform the following tasks:

- [“Creating the Realm Resource” on page 9-16](#)
- [“Creating the Shared Resources” on page 9-17](#)
- [“Creating the Console Resources” on page 9-18](#)
- [“Creating the PortalApp Resources” on page 9-19](#)

## Creating the Realm Resource

**Note:** `myrealm` is used in this procedure as the realm name because the WebLogic Portal 8.1 sample portal application exists in the `myrealm` realm. You can choose any realm name for your portal application; however, if you are integrating WebLogic Portal 9.2 using the WebLogic Server 9.x SSM, the Configuration ID must match the name of the WebLogic Server security realm.

To create a realm resources, perform the following steps:

1. Expand the Resources folder, and click Resources. The Resource page displays.
2. In the Resources page, select the Policy node, and click New. The Create Resource dialog box appears.
3. In the Name text box, enter `myrealm`, select `Binding` from the Type drop-down list box, and click Ok. The `myrealm` resource appears under the Policy node.
4. Select the `myrealm` resource and click Configure. The Configure Resource dialog box appears.
5. From the Type drop-down list box, select `Binding Application`, check the `Distribution Point` and `Allow Virtual Resources` check boxes, and click Ok.
6. Refer to [Table 9-2](#) and modify the configuration of the ASI Authorization provider as described there.

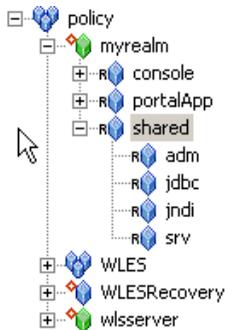
**Table 9-2 Portal Security Configuration Modifications**

| Security Provider          | Configuration Settings                                                                                                                                                                                                        |
|----------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ASI Authorization Provider | On the Details tab, set the Application Deployment Parent to <code>//app/policy/myrealm</code> and click Apply. On the Bindings tab, from the Bind drop-down menu, select <code>//app/policy/myrealm</code> , and click Bind. |

## Creating the Shared Resources

Figure 9-2 shows the shared resources that you must create.

**Figure 9-2 Shared Resources**



To create the `shared` resources, perform the following steps:

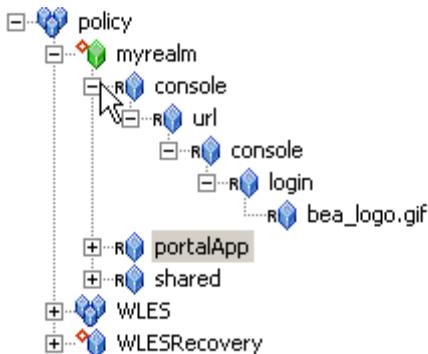
1. Select the `myrealm` resource and click New. The Create Resource dialog box appears.
2. In the Name text box, enter `shared`, and click Ok. The `shared` resource appears under `myrealm`.
3. Select the `shared` resource and click Configure. The Configure Resource dialog box appears.
4. Check the Allow Virtual Resources and click Ok.
5. Click the `shared` resource and click New. The Create Resource dialog box appears.

- In the Name text box, enter `adm` and click Ok. The `adm` resource appears under the `shared` resource.
- To configure the `jdbc`, `jndi`, and `svr` resources as shown in [Figure 9-2](#), repeat steps 5 and 6 for each resource.

## Creating the Console Resources

[Figure 9-3](#) shows the console resources that you must create.

**Figure 9-3 Console Resources**



To create the `console` resources, perform the following steps:

- Select the `myrealm` resource and click New. The Create Resource dialog box appears.
- For WebLogic Portal 9.2, in the Name text box, enter `consoleapp`, and click Ok. The `consoleapp` resource appears under `myrealm`.  
For WebLogic Portal 8.1, in the Name text box, enter `console`, and click Ok. The `console` resource appears under `myrealm`.
- To create the `url`, `console`, `login`, and `bea_logo.gif` resources as shown in [Figure 9-3](#), repeat steps 1 and 2 for each resource.
- Select the `console` or `consoleapp` resource directly under `myrealm` and click Configure. The Configure Resource dialog box appears.
- Check the Allow Virtual Resources and click Ok.

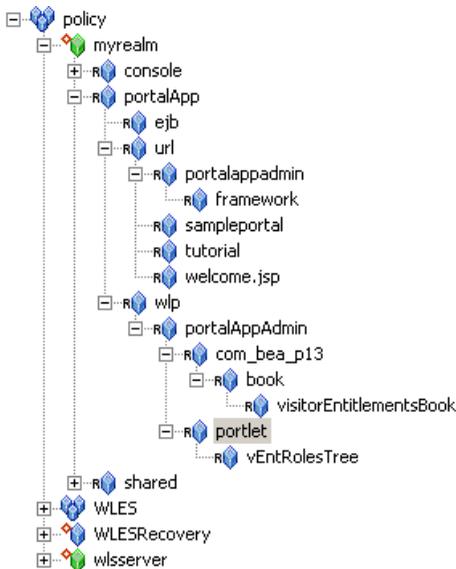
## Creating the PortalApp Resources

**Note:** This procedure uses `portalApp` as the name of the portal application resource because it is the name of the WebLogic Portal sample portal application. However, you should use the name of your portal application when creating the portal application resource.

Figure 9-4 shows the `portalApp` resources that you must create. In addition, if you are integrating WebLogic Portal 9.2 using the WebLogic Server 9.x SSM, you must create these virtual resources under `myrealm`:

- `bea_wls_internal`
- `wl_management_internal1`
- `wl_management_internal2`
- `pf-proliferation-jms`

**Figure 9-4** PortalApp Resources



To create the `portalApp` resources, perform the following steps:

1. Select the `myrealm` resource and click New. The Create Resource dialog box appears.

2. In the Name text box, enter `portalApp`, and click Ok. The `portalApp` resource appears under `myrealm`.
3. Select the `portalApp` resource and click New. The Create Resource dialog box appears.
4. In the Name text box, enter `ejb` and click Ok. The `ejb` resource appears under the `portalApp` resource.
5. Select the `ejb` resource and click Configure. The Configure Resource dialog box appears.
6. Check the Allow Virtual Resources and click Ok.
7. To configure the `url` resource, repeat steps 5 and 6.
8. Select the `portalApp` resource and click New. The Create Resource dialog box appears.
9. In the Name text box, enter `wlp` and click Ok. The `wlp` resource appears under the `portalApp` resource. Do **not** configure the `wlp` resource to allow virtual resources.
10. Select the `url` resource and click New. The Create Resource dialog box appears.
11. In the Name text box, enter `portalappadmin` and click Ok. The `portalappadmin` resource appears under the `url`.
12. Repeat steps 10 and 11 to create the remaining resources shown under the `portalApp` resource in [Figure 9-4](#). Do **not** configure any of the remaining resources to allow virtual resources.

## Creating the Role Mapping Policy

This section describes how to use the Administration Console to create role mapping policy that will be used to control access to portal application resources.

[Table 9-3](#) lists and describes the role mapping policy that you have to create for the WebLogic Portal domain.

**Table 9-3 Portal Application Role Mapping Policy**

| Role Mapping Policy                                                                        | Description                                                                                                                                                                                                                                                                                                            |
|--------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <pre>grant(//role/Everyone, //app/policy/myrealm, //sgrp/myusers/allusers/) if true;</pre> | <p>Creates the role mapping policy necessary for the Everyone role to be used in the myrealm Identity directory.</p> <p><b>Note:</b> If you do not create the Everyone role mapping policy correctly, none of the policy rules defined in <a href="#">Table 9-4</a> that use the Everyone role will work properly.</p> |

To create the role mapping policy, refer to [Table 9-3](#) and perform the following steps:

**Caution:** If you do not create the Everyone role mapping policy correctly, none of the authorization policies defined in [Table 9-4](#) and [Table 9-5](#) that use the Everyone role will work properly.

1. Expand the Policy folder in the left pane, and click Role Mapping Policies. The Role Mapping Policies page appears.
2. Click New. The Create Role Mapping Policy dialog box appears.
3. Select the Grant radio button.
4. Select the Roles tab, select Everyone in the Available Roles list box, and click Add.
5. Select the Resources tab, select myrealm, and click Add.
6. Select the Policy Subjects tab, select the allusers in the list box, click Add, and click Ok.

## Creating Authorization Policies

This section describes how to use the Administration Console to create authorization policies to protect portal application resources.

[Table 9-4](#) lists and describes the authorization policies that you have to create for the WebLogic Portal domain to protect the sample portal application resources. In addition, [Table 9-5](#) lists the authorization policies required for WebLogic Portal 9.2.

**Table 9-4 Portal Application Authorization Policies**

| Authorization Policies                                                                                                                                                                                                                                                   | Description                                                                                                                                                                                                                                            |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <pre>grant(any, //app/policy/myrealm/shared/svr, //role/Admin) if true; grant(any, //app/policy/myrealm/shared/adm, //role/Admin) if true;</pre>                                                                                                                         | <p>Authorization policies for booting the WebLogic Portal server and performing administrative tasks.</p>                                                                                                                                              |
| <pre>grant(any, //app/policy/myrealm/portalApp/url/ sampleportal, //role/Everyone) if true; grant(any, //app/policy/myrealm/portalApp/url/tutorial, //role/Everyone) if true; grant(any, //app/policy/myrealm/portalApp/url/welcome.jsp, //role/Everyone) if true;</pre> | <p>Grants permission to those in the role Everyone (includes the anonymous user) to access all of the tutorial and sample portal url resources. This authorization policy creates Portal open by default orientation for these two sample portals.</p> |
| <pre>grant(GET, //app/policy/myrealm/portalApp/url/ portalappadmin/framework, //role/Everyone) if true;</pre>                                                                                                                                                            | <p>Allows unauthenticated users to access images used on the Administration Portal login page.</p>                                                                                                                                                     |
| <pre>grant(any, //app/policy/myrealm/portalApp/url/ portalappadmin, //role/ PortalSystemAdministrator) if true;</pre>                                                                                                                                                    | <p>Grants permission for those is the role PortalSystemAdministrator to access the WebLogic Portal Administration url Portal resources</p>                                                                                                             |
| <pre>grant(lookup, //app/policy/myrealm/shared/jndi, //role/Everyone) if true;</pre>                                                                                                                                                                                     | <p>Grants permission for those is the Everyone role to lookup JNDI resources.</p>                                                                                                                                                                      |
| <pre>grant(reserve, //app/policy/myrealm/shared/jdbc, //role/Everyone) if true;</pre>                                                                                                                                                                                    | <p>Grants permission for those is the Everyone role to reserve JDBC resources.</p>                                                                                                                                                                     |
| <pre>grant(any, //app/policy/myrealm/console, //role/Admin) if true;</pre>                                                                                                                                                                                               | <p>Grants permission for those in the Admin role to access the url resources of the WebLogic Server console.</p>                                                                                                                                       |

**Table 9-4 Portal Application Authorization Policies (Continued)**

| Authorization Policies                                                                                         | Description                                                                                                                      |
|----------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------|
| <pre>grant (GET, //app/policy/myrealm/console/url/console/login/ bea_logo.gif, //role/Everyone) if true;</pre> | Grants permission for those in the Everyone role to get access to the bea_logo.gif image resource in the WebLogic Server console |
| <pre>grant (any, //app/policy/myrealm/portalApp/ejb, //role/Everyone)</pre>                                    | Initially allows access to all EJB methods.                                                                                      |

**Table 9-5 WebLogic Portal 9.2 Authorization Policies**

| <b>9.2 Authorization Policy</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    | <b>Description</b>                                                    |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------|
| <pre>grant( any, //app/policy/myrealm/consoleapp, //role/Admin) if true;</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     | Authorization policy for accessing the WLS 9.x Administration Console |
| <pre>grant( any, //app/policy/myrealm/pf-proliferation-jms, //role/Everyone) if true; grant( any, //app/policy/myrealm/bea_wls_internal, //role/Everyone) if true; grant( any, //app/policy/myrealm/wl_management_internall, //role/Everyone) if true; grant( any, //app/policy/myrealm/wl_management_internal2, //role/Everyone) if true;  grant( any, //app/policy/myrealm/shared/eis, //role/Everyone) if true; grant( any, //app/policy/myrealm/shared/ejb, //role/Everyone) if true; grant( any, //app/policy/myrealm/shared/jdbc, //role/Everyone) if true; grant( any, //app/policy/myrealm/shared/jms, //role/Everyone) if true; grant( any, //app/policy/myrealm/shared/jndi, //role/Everyone) if true; grant( any, //app/policy/myrealm/shared/svr, //role/Everyone) if true; grant( any, //app/policy/myrealm/shared/url, //role/Everyone) if true; grant( any, //app/policy/myrealm/shared/webservices, //role/Everyone) if true; grant( any, //app/policy/myrealm/shared/workcontext, //role/Everyone) if true;</pre> | Authorization policies required by WebLogic Portal 9.2                |

Perform the following steps create the authorization policies listed in [Table 9-4](#) (and, for WebLogic Portal 9.2, [Table 9-5](#)).

1. Expand the Policy folder in the left pane, and click Authorization Policies. The Authorization Policies page appears.
2. Click New. The Create Authorization Policy dialog box appears.
3. Select the Grant radio button.
4. To create the first authorization policy listed in [Table 9-4](#), click the Privileges tab, select the any privilege from the Select Privileges from Group list box, and click Add.
5. Click the Resources tab, select the `svr` resource from the Child Resource box and click Add.  
**Note:** If you want to assign multiple resources to a single privilege and role, you may define all of the resources in one authorization policy.
6. Click the Policy Subjects, select the `Admin` role from the Roles List box, click Add, and click Ok.
7. Repeat steps 4 to 6 for each of the remaining authorization policies.

## Policy for Visitor Entitlements to Portal Resources

Visitor entitlements is a mechanism used by WebLogic Portal for determining who may access the resources in a portal application and what they may do with those resources. AquaLogic Enterprise Security provides a means of defining robust role-based policy for portal resources. The resources that can be entitled within a portal application include:

- desktops
- books
- pages
- portlets
- look and feels

[Table 9-6](#) shows the capabilities of each of these resources:

**Table 9-6 Capabilities According to Resource Type**

| Resource Type | View | Minimize | Maximize | Edit | Remove |
|---------------|------|----------|----------|------|--------|
| Desktop       | x    |          |          |      |        |
| Book          | x    | x        | x        |      |        |
| Page          | x    |          |          |      |        |
| Portlet       | x    | x        | x        | x    | x      |
| Look & Feel   | x    |          |          |      |        |

The capabilities listed in [Table 9-6](#) are defined as follows:

- **View**—Determines whether or not the user can see the resource.
- **Minimize/Maximize**—Determines whether or not the user is able to minimize or maximize the portlet or book. This applies to books within a page, not to the primary book.
- **Edit**—Determines whether or not the user can edit the resource properties.
- **Remove**—Determines whether or not the user can remove the portlet from a page.

The following topics provide information on how to use AquaLogic Enterprise Security to configure portal resources:

- [“Configuring Policy for Desktops” on page 9-26](#)
- [“Configuring Policy for Books” on page 9-27](#)
- [“Configuring Policy for Pages” on page 9-28](#)
- [“Configuring Policy for Portlets” on page 9-28](#)
- [“Configuring Policy for Look and Feels” on page 9-29](#)
- [“Defining Policy for Portlets using Instance ID” on page 9-29](#)

## Configuring Policy for Desktops

A desktop is a view of the portal that the visitor accesses. There can be one or more desktops per portal, so the portal is effectively a container for the desktops. A Desktop is referenced as a resource in AquaLogic Enterprise Security in the following manner:

```
//app/policy/myrealm/portalapp/wlp/sampleportal/com_bea_p13n/Desktop/samplePortal
```

where:

- *myrealm* is the realm in which the portal application is installed
- *portalapp* is the portal application directory
- *sampleportal* is the name of the sample portal application
- *samplePortal* is the label definition of the desktop.

If you define an authorization policy at the *samplePortal* level, you can control access at the *samplePortal* desktop level.

[Table 9-7](#) shows an authorization policy that grants the *view* privilege to the *samplePortal* desktop for visitors in the *SampleVisitor* role.

**Table 9-7 SamplePortal Authorization Policy**

| Effect | Privilege | Resource                                                              | Policy Subject     |
|--------|-----------|-----------------------------------------------------------------------|--------------------|
| Grant  | view      | /myrealm/portalapp/wlp/sampleportal/com_bea_p13n/Desktop/samplePortal | SampleVisitor role |

## Configuring Policy for Books

A book is a collection of pages. A book is referenced as a resource in AquaLogic Enterprise Security in the following manner:

```
//app/policy/myrealm/portalapp/wlp/sampleportal/com_bea_p13n/Book/book_1
```

where *book\_1* is the label definition of the book.

If you define an authorization policy at the *book\_1* level, you can control access at the *book\_1* book level.

[Table 9-8](#) shows an authorization policy that grants the *view* privilege to the *book\_1* book for visitors in the *SampleVisitor* role.

**Table 9-8 Book\_1 Authorization Policy**

| Effect | Privilege | Resource                                                     | Policy Subject     |
|--------|-----------|--------------------------------------------------------------|--------------------|
| Grant  | view      | /myrealm/portalapp/wlp/sampleportal/com_bea_p13n/Book/book_1 | SampleVisitor role |

## Configuring Policy for Pages

A page is the primary holder of individual portal elements such as portlets. A page is referenced as a resource in AquaLogic Enterprise Security in the following manner:

```
//app/policy/myrealm/portalapp/wlp/sampleportal/com_bea_p13n/Page/page_2
```

where *page\_2* is the label definition of the page.

If you define an authorization policy at the *page\_2* level, you can control access at the *page\_2* page level.

[Table 9-9](#) shows an authorization policy that grants the `view` privilege to the *page\_2* page for visitors in the `SampleVisitor` role.

**Table 9-9 Page\_2 Authorization Policy**

| Effect | Privilege | Resource                                                     | Policy Subject     |
|--------|-----------|--------------------------------------------------------------|--------------------|
| Grant  | view      | /myrealm/portalapp/wlp/sampleportal/com_bea_p13n/Page/page_2 | SampleVisitor role |

## Configuring Policy for Portlets

Portlets are the visible components that act as the interface to applications and content. A portlet is referenced as a resource in AquaLogic Enterprise Security in the following manner:

```
//app/policy/myrealm/portalapp/wlp/sampleportal/com_bea_p13n/Portlet/portlet_login1
```

where *portlet\_login1* is the label definition of the portlet.

If you define an authorization policy at the *portlet\_login1* level, you can control access at the *portlet\_login1* Portlet level.

[Table 9-10](#) shows an authorization policy that grants the `view` privilege to the `portlet_login1` Portlet for visitors in the `SampleVisitor` role.

**Table 9-10 Portlet\_login1 Authorization Policy**

| Effect | Privilege | Resource                                                                    | Policy Subject        |
|--------|-----------|-----------------------------------------------------------------------------|-----------------------|
| Grant  | view      | /myrealm/portalapp/wlp/sampleportal/com_bea_p13n<br>/Portlet/portlet_login1 | SampleVisitor<br>role |

## Configuring Policy for Look and Feels

A Look and Feel is a selectable combination of skins and skeletons that determine the physical appearance of a portal desktop. A Look and Feel is referenced as a resource in AquaLogic Enterprise Security in the following manner:

```
//app/policy/myrealm/portalapp/wlp/sampleportal/com_bea_p13n/LookAndFeel/textLookAndFeel
```

where `textLookAndFeel` is the label definition of the Look and Feel.

If you define an authorization policy at the `textLookAndFeel` level, you can control access at the `textLookAndFeel` level.

[Table 9-11](#) shows an authorization policy that grants the `view` privilege to the `textLookAndFeel` Look and Feel visitors in the `SampleVisitor` role.

**Table 9-11 Portlet\_login1 Authorization Policy**

| Effect | Privilege | Resource                                                                         | Policy Subject        |
|--------|-----------|----------------------------------------------------------------------------------|-----------------------|
| Grant  | view      | /myrealm/portalapp/wlp/sampleportal/com_bea_p13n<br>/LookAndFeel/textLookAndFeel | SampleVisitor<br>role |

## Defining Policy for Portlets using Instance ID

Portlets have a unique instance ID that allows for granular authorization policy definition outside the standard hierarchy of the Desktop->Book->Page->Portlet. To use this in AquaLogic Enterprise Security, add a condition statement in the portlet rule that adds the portlet instance ID. For example:

```
grant( [//priv/maximized, //priv/minimized, //priv/view],
//app/policy/myrealm/portalapp/wlp/sampleportal/com_bea_p13n/Portlet
/portlet_login1, //role/Operator) if instanceid = "portlet_login1";
```

[Table 9-12](#) shows an authorization policy that grants the `view` privilege to the `portlet_login1` Portlet for visitors in the Operator role.

**Table 9-12 Portlet\_login1 Authorization Policy Using Instance ID**

| Effect | Privilege | Resource                                                                | Policy Subject | Condition                         |
|--------|-----------|-------------------------------------------------------------------------|----------------|-----------------------------------|
| Grant  | view      | /myrealm/portalapp/wlp/sampleportal/com_bea_p13n/Portlet/portlet_login1 | Operator role  | if instanceid = "portlet_login1"; |

## Discovering Portal Application Resources

When developing policies for use with a Security Service Module, you can use the discovery mode feature of the AquaLogic Enterprise Security Administration Server to help define your policy components. Instructions for using discovery mode are provided in the [Resource Discovery](#) section in the *Policy Managers Guide*.

## Distributing Policy and Security Configuration

Distribute policy and security configuration to the WebLogic Server Security Service Module.

For information on how to distribute policy and security configuration, see the Console Help. Be sure to verify the results of your distribution.

## Starting the WebLogic Portal Server

To start a WebLogic Portal server, perform the following steps:

1. Open a shell (command prompt) on the machine on which you created the portal domain.
2. Change to the portal domain directory:
  - For WebLogic Portal 9.2, `BEA_HOME\weblogic92\samples\domains\portal.`
  - For WebLogic Portal 8.1, `BEA_HOME\user_projects\domains\portalDomain.`
3. Run the following script:

On Windows: `startWebLogic.cmd`

On UNIX: `startWeblogic.sh`

## Configuring Portal Administration to Use the WebLogic Authenticator

To use the WebLogic Authentication provider to manage administrative users for portal administration, perform the following steps:

1. Within the Portal Administration console, go to `Service Administration`.
2. Select `Authentication Hierarchy Service`.
3. Add `webLogicAuthenticator` to the `Authentication Providers to Build` list.

## Using Portal Administration Tools to Create a Portal Desktop

Before you can use AquaLogic Enterprise Security to control access to a portal desktop, you must use WebLogic Portal Administration Tools to create a portal desktop.

To create a portal desktops, perform the following steps:

1. To have full control of the definition and instance labels, use WebLogic Workshop to create the initial portal application.
2. Using the initial portal application as a template, use the Portal Administration Tools to create the portal desktops.

For instructions on using Portal Administration Tools to create portal desktops, see "[Create a Desktop](#)" in the WebLogic Administration Portal Online Help.

To create a portal desktop for the sample portal application using the Portal Administration Tools, perform the following steps:

1. Open a browser and point it to: `http://<hostname>:7001/testportalappAdmin`. The Sign In page appears.
2. Enter Username: `portaladmin`, Password: `portaladmin`, and click Sign In. The Portal Resources navigation tree appears.
3. Right click on Portals and select New Portal. The Portal Resource page appears.

4. Enter a Name for this Portal, for example: `myportal`, enter a partial URL for this Portal, for example: `myportal`, and click Save. The Available Portals list appears.
5. Select the `myportal` link from the list. The Editing Portal: `myportal` page appears.
6. Click Create New Desktop. The New Desktop Properties dialog appears.
7. Enter Title, for example: `mydesktop` and a Partial URL, for example: `mydesktop`.
8. Choose a Template. Choose: `testportalweb/test.portal` and click Create New Desktop. Desktops contained in Portal `myportal` list appear.
9. Select `mydesktop` from the list. The Editing Desktop: `mydesktop` page appears.
10. Select View Desktop. `mydesktop` is displayed in a new browser window. Verify that the desktop contains the sample portal application.
11. Close the `mydesktop` browser and Logout. The Sign In page appears.
12. Exit the Portal Administration Tool by closing its browser.

## Accessing the Portal Application

To access a portal application running on a portal server, open a browser and point it to the desktop URL. For example, if you set up the desktop for the sample portal application as described in [“Using Portal Administration Tools to Create a Portal Desktop”](#) on page 9-31, you can access the sample portal application using the following URL:

```
http://<host_name>:7001/sampleportal/appmanager/myportal/mydesktop
```

where `<host_name>` is the machine on which portal application is running.

# Integrating with AquaLogic Data Services Platform

This section describes how to integrate AquaLogic Enterprise Security with AquaLogic Data Services Platform, using the WebLogic Server 8.1 SSM or WebLogic Server 9.x SSM. It includes the following topics:

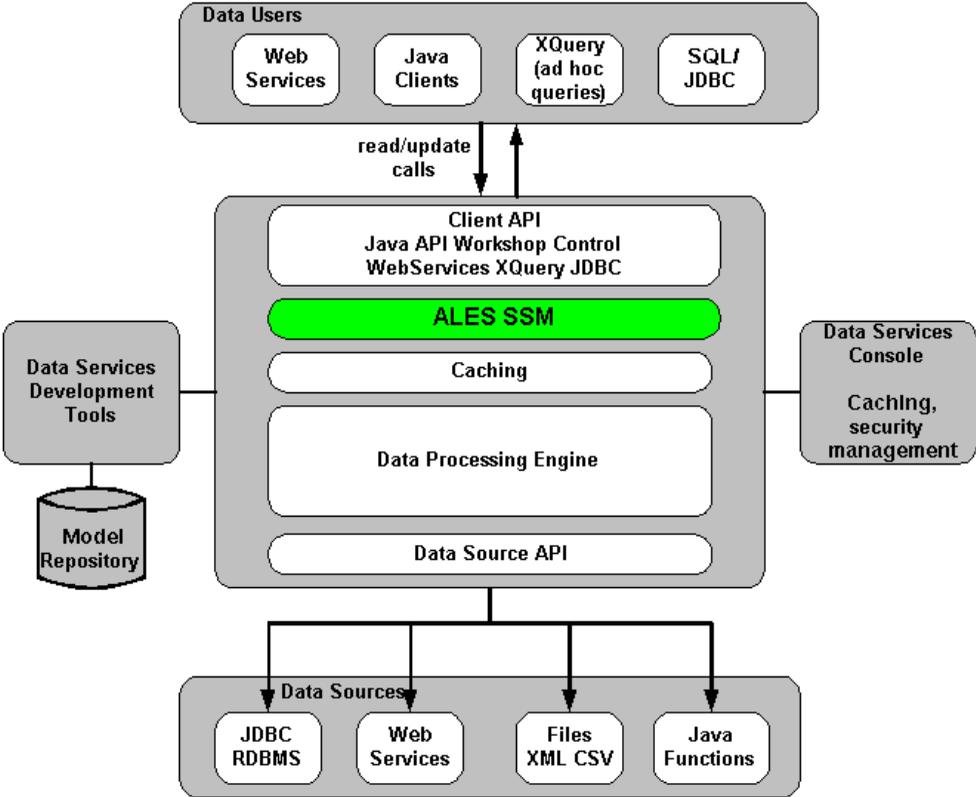
- [“Introduction” on page 10-1](#)
- [“Integration Pre-Requisites” on page 10-4](#)
- [“Integrating with AquaLogic Data Services Platform: Main Steps” on page 10-4](#)
- [“Enabling Elements for Access Control” on page 10-5](#)
- [“Creating the WebLogic Server SSM Configuration” on page 10-6](#)
- [“Configuring Policy for Data Services” on page 10-8](#)

## Introduction

AquaLogic Enterprise Security (ALES) can provide fine-grained entitlements for Data Services serviced by AquaLogic Data Services Platform (ALDSP) 2.1. AquaLogic Enterprise Security can be used to manage access control to entire services or elements of those services. AquaLogic Enterprise Security allows you to have common set of security policies for a heterogeneous environment, and a single security infrastructure that supports WebLogic Portal, WebLogic Server, and custom applications.

The ALES service does not replace all of the management functionality provided by the ALDSP. The ALDSP Administrative console (Idconsole) is still used to manage all of the attributes of the various data services aggregated by ALDSP (see [Figure 10-1](#)).

**Figure 10-1 ALDSP Integration Overview**



The AquaLogic Enterprise Security WebLogic SSM enables you to write, deploy, and manage fine-grained policy for controlling access to all WebLogic server application resources, including data services. A specific resource type `ld` allows a security administrator to represent the data services in the ALES resource hierarchy. Elements of that data service are also converted to the ALES format for evaluation by the ASI authorization engine.

For more information, see the following topics:

- [“Integration Features” on page 10-3](#)
- [“Supported Use-case Scenario” on page 10-3](#)
- [“Constraints and Limitations” on page 10-3](#)

## Integration Features

AquaLogic Data Service Platform (ALDSP) 2.1 requires WebLogic Server 9.2, 9.1, or 8.1 (with Service Pack 4 or 5) and uses the WLS 9.x or WLS 8.1 SSM. While the ALES framework allows for different security providers to be used with ALDSP, the following providers were certified:

- The ASI Authorization and Role Mapping providers enable you to use AquaLogic Enterprise Security to write, deploy, and manage fine-grained authorization of Data Services.
- The Database Authentication provider performs authentication services for the SSM.

## Supported Use-case Scenario

The following use-case scenario is supported when you integrate AquaLogic Enterprise Security with AquaLogic Data Services Platform:

- The AquaLogic Enterprise Security Administration Server assumes responsibility for management and policy of data services and elements of those services through the ALES console or Policy Management API.
- The AquaLogic Enterprise Security Administration Server is responsible for access control of J2EE applications deployed on the ALDSP WebLogic Server.
- The ALDSP Administration Console continues to be the management point for data services.

## Constraints and Limitations

AquaLogic Enterprise Security integration with ALDSP has the following constraints and limitations:

- Data service elements must be enabled for security through the ALDSP Admin console before ALES can manage element-based access control.
- ALES cannot provide entitlements and control which records are returned by the data service. This can still be done, but must be performed through the ALDSP Admin console.

# Integration Pre-Requisites

Before you begin, you must ensure that the following pre-requisites are satisfied:

- WebLogic Server 9.1 or 9.2, or 8.1 with Service Pack 4 or 5
- WebLogic Server 8.1 or 9.x Security Service Module
- You must have access to an ALES Administration Console that is running on the AquaLogic Enterprise Security 2.2 Administration Server on either the local machine or a remote machine.
- AquaLogic Data Services Platform 2.1

## Integrating with AquaLogic Data Services Platform: Main Steps

This section describes how to integrate AquaLogic Enterprise Security with AquaLogic Data Services Platform. Once integrated, you can use the AquaLogic Enterprise Security Administration Console to write and deploy a set of authorization and role mapping policies to protect Data Services and elements of those services.

**Note:** The instructions provided in this section use as an example the ALDSP sample application RTL App that ships with the ALDSP 2.1 software distribution. This procedure is representative of any integration of AquaLogic Enterprise Security with ALDSP.

To integrate AquaLogic Enterprise Security with AquaLogic Data Services Platform, perform the following tasks:

1. Install the ALES Administration Server, as described in [Installing the Administration Server](#).
2. Install the WebLogic Server SSM, as described in [Installing Security Service Modules](#). If you use ALSDSP with WebLogic Server 8.1, install the WebLogic Server 8.1 SSM. If you use ALSDSP with WebLogic Server 9.1 or 9.2, install the WebLogic Server 9.x SSM.
3. Use the ALDSP Administration console to enable the elements to which you want to control access, as described in [“Enabling Elements for Access Control” on page 10-5](#).
4. Configure the WebLogic Server SSM, as described in [“Creating the WebLogic Server SSM Configuration” on page 10-6](#). Note that this procedure varies, depending on whether you are using WLS 8.1 or WLS 9.x.

5. Bind the WebLogic Server SSM configuration, as described in [“Binding the SSM Configuration” on page 10-7](#).
6. Distribute the WebLogic Server SSM configuration, as described in [“Distributing the SSM Configuration” on page 10-7](#).
7. Create an instance of the WebLogic Server SSM, as described in [“Creating an Instance of the Security Service Module” on page 10-7](#).
8. Enroll the instance of the WebLogic Server SSM, as described in [“Enrolling the Instance of the Security Service Module” on page 10-7](#).
9. Create the `startWebLogicALES` file for WebLogic Server, as described in [“Creating the WebLogic Server startWebLogicALES File” on page 10-7](#). Note that this procedure varies, depending on whether you are using WLS 8.1 or WLS 9.x.
10. Create the security configuration file, as described in [“Creating the security.properties File” on page 10-8](#). Note that this step applies if you are using WLS 8.1 and does *not* apply if you are using WLS 9.x.
11. Configure security policies for your data services, as described in [“Configuring Policy for Data Services” on page 10-8](#).

## Enabling Elements for Access Control

Before enabling your ALDSP domain for ALES, open the ALDSP Administration console:

1. Open a browser to visit `http://<hostname>:<port>/ldconsole`.
2. Login as Administrator.
3. Browse to the data services elements that are to be controlled by ALES. For this example, enable the following:
  - a. Expand `RTLServices/OrderSummaryView` and select Security Tab.
  - b. Select Secured Elements Tab.
  - c. Expand elements and check `OrderSummary > OrderDate` as an element to be secured. (This allows the element call to go to the security check.)

Do the same to secure `CustomerView.ds > CUSTOMER > ORDERS > ORDER_SUMMARY > OrderDate`.

# Creating the WebLogic Server SSM Configuration

Securing ALDSP with ALES employs either the WLS 8.1 SSM or the WLS 9.x SSM. Install the WLS SSM on the machines on which you have installed ALDSP, as described in [Installing Security Service Modules](#).

Next, create a new WLS SSM configuration named `aldsprealm`. An SSM configuration defines the set of security providers to use for adjudication, authentication, auditing, authorization, role mapping, and credential mapping services.

Refer to [Table 10-1](#) and use the AquaLogic Enterprise Security Administration Console (for the WLS 8.1 SSM) or the WebLogic Server Administration Console (for the WLS 9.x SSM) to configure the security providers listed there. Set the Configuration ID to `aldsprealm`. For instructions on creating an SSM configuration, see [Configuring and Binding a Security Service Module](#) in *Installing Security Service Modules* and the Console Help.

**Table 10-1 Providers for Use in ALDSP Integration**

| Provider                            | Configuration Settings                                                                                                                                                                                                                                                                                                                                                                                                                           |
|-------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ASI Adjudication Provider           | Accept default settings.                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Log4j Auditor                       | Accept the default settings, and click Create.                                                                                                                                                                                                                                                                                                                                                                                                   |
| Database Authentication Provider    | Set the Control Flag to SUFFICIENT, and click Create. For the Details tab settings, except for the Identity Scope, the parameters are populated automatically. Set the Identity Scope to <code>aldspusers</code> , and click Apply.<br><br>Note: Even though you set the Identity Scope to <code>aldspusers</code> , you do not actually create the <code>aldspusers</code> identity until you perform the steps in Creating the Realm Resource. |
| ASI Authorization Provider          | On the General tab, accept the default settings, and click Create.<br>On the Details tab, set the Identity Scope to <code>aldspusers</code> .                                                                                                                                                                                                                                                                                                    |
| WebLogic Credential Mapper Provider | Uncheck the Credential Mapping Deployment Enabled check box, and click Create.                                                                                                                                                                                                                                                                                                                                                                   |
| ASI Role Mapping Provider           | On the General tab, accept the default settings, and click Create. On the Details tab, set the Identity Scope to <code>aldspusers</code> .                                                                                                                                                                                                                                                                                                       |

## Binding the SSM Configuration

The SSM configuration must be bound to a Service Control Manager (SCM).

To bind the `aldsprealm` SSM configuration, see “Binding a Security Service Module to a Service Control Manager” in the Console Help.

## Distributing the SSM Configuration

The `aldsprealm` SSM configuration must be distributed.

To distribute the `aldsprealm` SSM configuration, see “Distributing Configuration” in the Console Help.

## Creating an Instance of the Security Service Module

Before starting a WebLogic Server Security Service Module, you must first create an instance of the WebLogic Server Security Service Module using the Create New Instance Wizard.

For information about creating an instance of a WebLogic Server Security Service Module, see [Creating an Instance of a Security Service Module](#) in *Installing Security Service Modules*.

## Enrolling the Instance of the Security Service Module

You must have the ALES Administration Server running prior to enrolling the Security Service Module. For information about enrolling a security service module, see [Enrolling the Instance of the Security Service Module](#) in *Installing Security Service Modules*.

## Creating the WebLogic Server startWebLogicALES File

Before you can start a WebLogic Server instance that uses BEA AquaLogic Enterprise Security, you must create the `startweblogicALES` file based on the `startweblogic` file that is located in the WebLogic domain. For information about how to do this, see:

- For the WLS 8.1 SSM: [“Modifying the startWebLogic File” on page 7-2](#)
- For the WLS 9.x SSM: [“Modifying the startWebLogic File” on page 8-7](#)

The `startweblogic` file for the ALDSP domain for RTLApp is located in:

```
<bea_home>\<weblogic_home>\samples\domains\ldplatform
```

## Creating the security.properties File

If you are using the WLS 8.1 SSM, create a text file named `security.properties` and place it in the domain directory. You use this file to define the AquaLogic Enterprise Security realm and the default realm.

```
# AquaLogic Enterprise Security Configuration File
#
# This file contains AquaLogic Enterprise Security configuration
# properties. By default, the AquaLogic Enterprise Security runtime
# looks for a property file called 'security.properties' in the
# working directory
wles.realm=aldsprealm
wles.default.realm=aldsprealm
```

**Note:** This step does not apply if you are using the WLS 9.x SSM

## Configuring Policy for Data Services

Developing a set of policies typically begins by determining which resources you need to protect and your access control requirements. You then create the identity directory, resources, groups, users, and roles that you will use to write policies to protect those resources. Next you write a set of authorization and role mapping policies to define access control on those resources. Finally, you deploy the set of policies to the WebLogic Server Security Service Module that you use to control access to your data services.

For more information on how to use the ALES Administration Console to write policy, see the [Policy Managers Guide](#) and the Console Help. In addition, the ALES Administration Server installation includes a set of sample policies for BEA AquaLogic Data Services Platform, located at `BEA_HOME/ales22-admin/examples/policy/aldsp_sample_policy`. You can import these sample policies and use them as a starting point for developing a full set of policies for your applications. For information about how to import the sample policies, see the README file in the sample directory and see also [Importing Policy Data](#) in the *Policy Managers Guide*.

This section covers the following topics:

- [“Creating the Identity Directory and Users” on page 10-9](#)
- [“Creating the RTLApp Application Resources” on page 10-10](#)
- [“Creating the ALDSP Resources” on page 10-10](#)
- [“Creating the Role Mapping Policies” on page 10-12](#)

- [“Creating Authorization Policies” on page 10-13](#)

## Creating the Identity Directory and Users

This section describes how to use the ALES Administration Console to create an identity directory, groups, and users for an ALDSP application.

**Note:** This procedure uses `aldspusers` as the name of the Identity directory; however, you can use a different name.

To create the Identity directory and users:

1. In the left pane, click Identity. The Identity page displays the name of each directory available.
2. Click New. The Create Directory dialog box appears.
3. In the Name text box, type `aldspusers` and click OK. The `aldspusers` directory appears in the list of Identity directories.
4. In the left pane, click Groups. The `aldspusers > Groups` page displays.
5. Click New. The Create Group dialog box appears.
6. Create the `LDSampleUsers` Group.
7. Create the sample users used in `RTLApp` and add them to the `LDSampleUsers` group:
  - Jack (password: `weblogic`)
  - Steve (password: `weblogic`)
  - Tim (password: `weblogic`)
8. Create `Idconsole` administrator:
  - `weblogic` (password: `weblogic`)

## Configuring Resources and Privilege

This section describes how to use the ALES Administration Console to define the application resources that you will protect using ALES.

To configure resources, perform the following tasks:

- [“Creating the RTLApp Application Resources” on page 10-10](#)
- [“Creating the ALDSP Resources” on page 10-10](#)

## Creating the RTLApp Application Resources

**Note:** You can choose any application name for your ALDSP application.

To create application resources, use the Administration Console to perform the following steps:

1. Expand the Resources folder, and click Resources. The Resource page displays.
2. In the Resources page, select the Policy node, and click New. The Create Resource dialog box appears.
3. In the Name text box, enter `aldsprealm`, select Binding from the Type drop-down list box, and click Ok. The `aldsprealm` resource appears under the Policy node.
4. Select the `aldsprealm` resource and click Configure. The Configure Resource dialog box appears.
5. From the Type drop-down list box, select Binding Application, check the Distribution Point and Allow Virtual Resources check boxes, and click Ok.
6. Refer to [Table 10-2](#) and modify the configuration of the ASI Authorization provider and the ASI Role Mapper provider as described there.

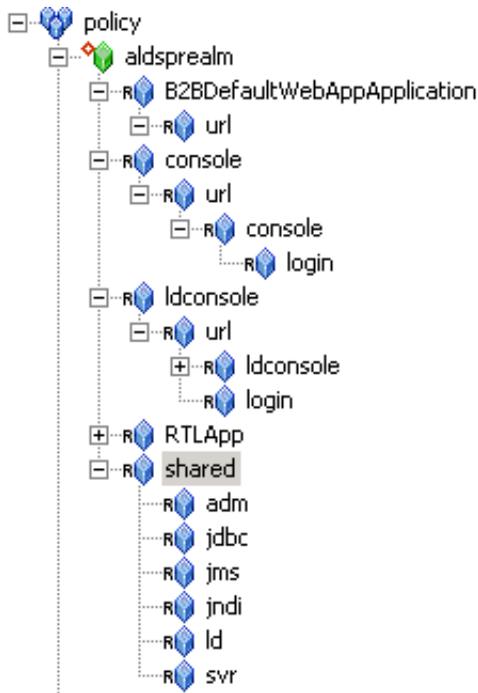
**Table 10-2 ALDSP SSM Configuration Modifications**

| Security Provider          | Configuration Setting                                                                                                                                                                                                                                                                            |
|----------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ASI Authorization Provider | <ol style="list-style-type: none"><li>1. On the Details tab, set the Application Deployment Parent to <code>//app/policy/aldsprealm</code> and click Apply.</li><li>2. On the Bindings tab, from the Bind drop-down menu, select <code>//app/policy/aldsprealm</code>, and click Bind.</li></ol> |
| ASI Role Mapper Provider   | <ol style="list-style-type: none"><li>1. On the Details tab, set the Application Deployment Parent to <code>//app/policy/aldsprealm</code> and click Apply.</li><li>2. On the Bindings tab, from the Bind drop-down menu, select <code>//app/policy/aldsprealm</code>, and click Bind.</li></ol> |

## Creating the ALDSP Resources

[Figure 10-2](#) shows the ALDSP resource tree with all nodes expanded except the RTLApp node. The resources under that RTLApp node are shown in [Figure 10-3](#). You must create the resources shown in [Figure 10-2](#) and [Figure 10-3](#).

**Figure 10-2 ALDSP Resource Tree with RTLApp Node Collapsed**



**Figure 10-3 ALDSP Resource Tree with RTLApp Node Expanded**



## Creating the Role Mapping Policies

This section describes how to use the Administration Console to create the role mapping policies that will be used to control access the sample ALDSP application.

Table 10-3 lists the role mapping policies required for the WebLogic domain.

**Table 10-3 ALDSP Application Role Mapping Policy**

| Role Mapping Policy                                                                                          | Description                                                                                                                                                                                                                              |
|--------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>grant(/role/Everyone,<br/>//app/policy/aldsprealm,<br/>//sgrp/aldspusers/allusers/<br/>if true;</code> | Creates the role mapping policy necessary for the Everyone role to be used in the aldsprealm Identity directory.<br><br>Note: If you do not create the Everyone role mapping policy correctly, none of the policy rules defined in Table |
| <code>grant(/role/Admin,<br/>//app/policy/aldsprealm,<br/>//user/aldspusers/weblogic/<br/>) if true;</code>  | Grants the weblogic user Admin role within the aldsp realm.                                                                                                                                                                              |

To create the role mapping policies, refer to [Table 10-3](#) and perform the following steps.

**Note:** If you do not create the Everyone role mapping policy correctly, none of the authorization policies defined in [Figure 10-4](#) will work.

1. Expand the Policy folder in the left pane, and click Role Mapping Policies. The Role Mapping Policies page appears.
2. Click New. The Create Role Mapping Policy dialog box appears.
3. Select the Grant radio button.
4. Select the Roles tab, select Everyone in the Available Roles list box, and click Add.
5. Select the Resources tab, select `aldsprealm`, and click Add.
6. Select the Policy Subjects tab, select `allusers` in the list box, click Add, and click Ok.

## Creating Authorization Policies

This section describes how to use the Administration Console to create authorization policies to protect data services and application resources. [Table 10-4](#) lists the authorization policies required for WebLogic Server, the WebLogic Server console, and the RTL sample application.

**Table 10-4 Authorization Policies**

| Authorization Policy                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        | Description                                                                                                     |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------|
| <pre>grant(any, //app/policy/alDSPrealm/shared/svr, //role/Admin) if true; grant(any, //app/policy/alDSPrealm/shared/adm, //role/Admin) if true; grant(any, [//app/policy/ alDSPrealm /RTLApp/ejb, //app/policy/alDSPrealm/RTLApp/ld, //app/policy/alDSPr ealm/RTLApp/url/rtlselfservice/pages], [//role/Admin]) if true; grant(any, [//app/policy alDSPrealm /RTLApp/ejb/RTLApp_ejb.jar/Metadata, //app/policy/alDSPrealm/RT LApp/ejb/RTLApp_ejb.jar], [//role/Admin]) if true; grant([any, //priv/create], //app/policy/ alDSPrealm /RTLApp/ejb/.workshop, //role/Admin) if true; grant(any, [//app/policy/ alDSPrealm /console, //app/policy/alDSPrealm/shared/svr, //app/policy/alDSPrealm/s hared/adm], [//role/Admin]) if true;</pre> | Grants Admin Role and/or weblogic user permission to boot the WebLogic Server and perform administrative tasks. |

**Table 10-4 Authorization Policies (Continued)**

| Authorization Policy                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    | Description                                                                                                                      |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------|
| <pre>grant(/priv/lookup, //app/policy/alDSPrealm/shared/jms, //role/Everyone) if true; grant(any, //app/policy/alDSPrealm/shared/ld, //role/Everyone) if true; grant(/priv/lookup, [/app/policy/alDSPrealm/shared/jdbc, //app/policy/alDSPrealm/shared/j ndi], //role/Everyone) if true; grant(/priv/send, //app/policy/alDSPrealm/shared/jms, //role/Everyone) if true; grant(/priv/GET, //app/policy/alDSPrealm/console/url/console/login, //role/Everyone) if true; grant(/priv/reserve, //app/policy/alDSPrealm/shared/jdbc, //role/Everyone) if true; grant([/priv/GET, /priv/POST], //app/policy/alDSPrealm/ldconsole/url/ldconsole/login, //role/Everyone) if true; grant([/priv/GET, /priv/POST], //app/policy/alDSPrealm/RTLApp/url/elecws, //role/Everyone) if true; grant(/priv/GET, //app/policy/alDSPrealm/ldconsole/url/ldconsole/images, //role/Everyone) if true; grant(/priv/GET, [/app/policy/ alDSPrealm /B2BDefaultWebAppApplication/url, //app/policy/alDSPrealm/RTLAp p/url/rtlselfservice/resources, //app/policy/alDSPrealm/RTLApp/url/rtlse lfservice/wlwdir], //role/Everyone) if true;</pre> | <p>Grants permission to those in the role Everyone (includes the anonymous user) to access all of the shared open resources.</p> |
| <pre>grant([/priv/GET, /priv/POST], //app/policy/ alDSPrealm /RTLApp/url/rtlselfservice, //user/alDSPusers/Steve/) if true; deny(any, [/app/policy/ alDSPrealm /RTLApp/ld/DataServices/RTLServices/OrderSummaryView.ds/OR DER_SUMMARY/OrderDate, //app/policy/alDSPrealm/RTLApp/ld/D ataServices/RTLServices/CustomerView.ds/CUSTOMER/ORDERS/ ORDER_SUMMARY/OrderDate], //user/alDSPusers/Steve/) if true;</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        | <p>Denies Steve access to the Order Date element of the Customer View Data Service</p>                                           |
| <pre>deny(any, //app/policy/alDSPrealm/RTLApp/ld/DataServices/RTLServices/Profi leView.ds, //user/alDSPusers/Jack/) if true;</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | <p>Denies Jack access to an entire data service</p>                                                                              |

**Table 10-4 Authorization Policies (Continued)**

| Authorization Policy                                                                                                                                                                                   | Description                                           |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------|
| grant(any, [//app/policy/alDSPrealm/RTLApp/ejb, //app/policy/alDSPrealm/RTLA pp/ld, //app/policy/alDSPrealm/RTLApp/url/rtlselfservice/pages], [//sgrp/alDSPusers/LDSampleUsers, /role/Admin]) if true; | Grants Admin and Sample Users access to Data Services |

Perform the following steps to create the authorization policies listed in [Table 10-4](#).

1. Expand the Policy folder in the left pane, and click Authorization Policies. The Authorization Policies page appears.
2. Click New. The Create Authorization Policy dialog box appears.
3. Select the Grant radio button.
4. To create the first authorization policy listed in [Table 10-4](#), click the Policy Subjects, select the Admin role from the Roles List box, click Add, and click Ok.

**Note:** If [Table 10-4](#) lists multiple resources for a single privilege and role, you may define all of the resources in one authorization policy.

5. Repeat for each of the remaining authorization policies listed in [Table 10-4](#).

## Discovering Data Services

When developing policies for use with a Security Service Module, you can use the Discovery mode feature to help define your policy components. Instructions for using Discovery mode are provided in the [Resource Discovery](#) section in the *Policy Managers Guide*.

## Distributing Policy and SSM Configuration

Distribute policy and SSM configuration to the WebLogic Server SSM.

For information on how to distribute policy and SSM configuration, see “Deployment” in the Console Help. Be sure to verify the results of your distribution.

## Starting the WebLogic Server

To start a WebLogic Server instance, perform the following steps:

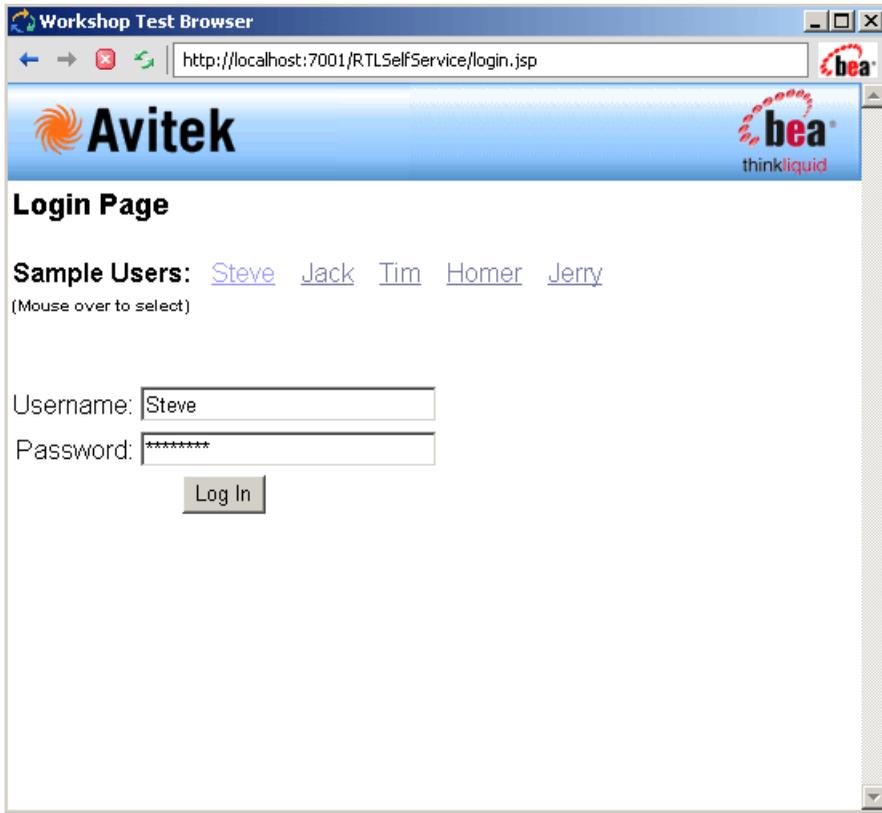
1. Open a shell (command prompt) on the machine on which you created the domain.
2. Change to the ALDSP sample domain directory:  
`<bea_home>\<weblogic_home>\samples\domains\ldplatform`
3. Run one of the following scripts:  
On Windows: `startWebLogicALES.cmd`  
On UNIX: `startWeblogicALES.sh`

## Accessing the ALDSP Application

To access the RTLApp running on an ALDSP server:

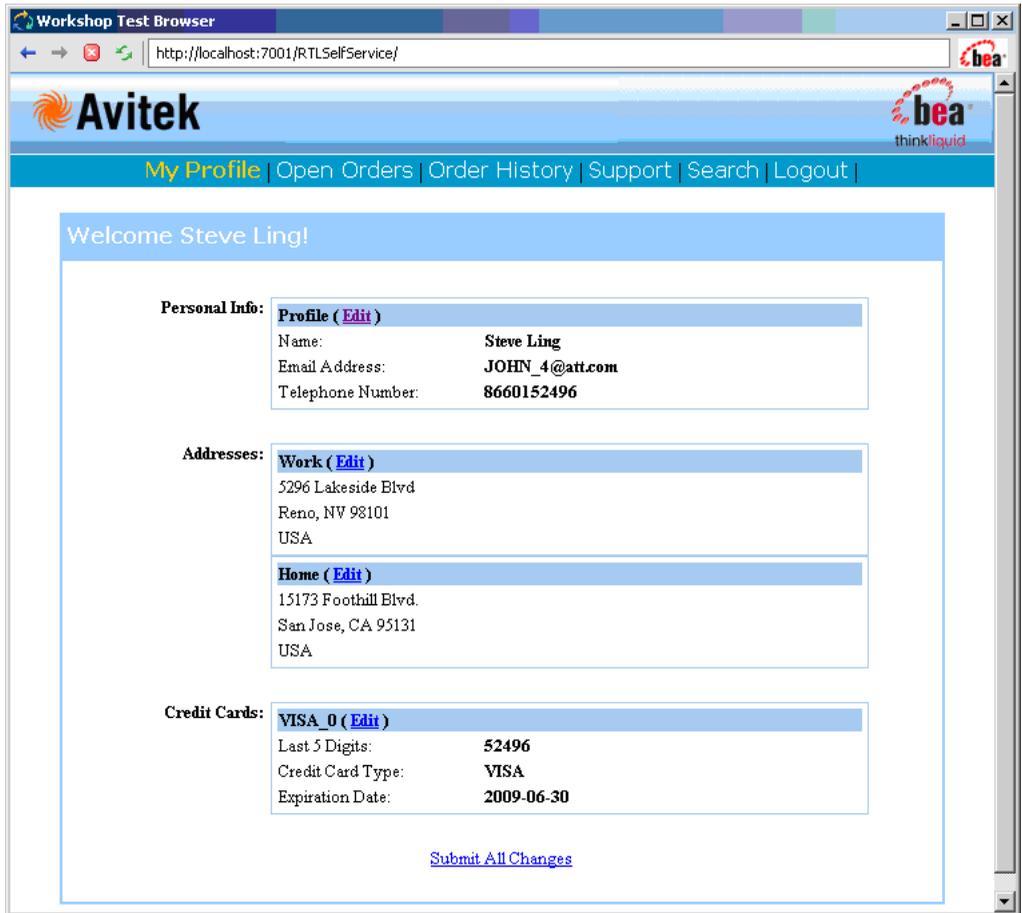
1. Open browser to visit `http://<hostname>:<port>/RTLSelfService`, where `<hostname>` is the machine on which RTL application is running. The browser is redirected to the authentication page (see [Figure 10-4](#)).

Figure 10-4 Authentication Page



2. Set username as Steve by dragging over link, then click Login button. Your client should be granted access to the Profile Page (see [Figure 10-5](#)).

Figure 10-5 Profile Page



3. Select Open Orders Page from top menu. Open orders should be visible (see Figure 10-6). Order Data should have "ACCESS DENIED".

Figure 10-6 Open Orders Page

Workshop Test Browser  
<http://localhost:7001/RTLSelfService/Pages/Home.do>

**Avitek** bea thinkliquid

My Profile | **Open Orders** | Order History | Support | Search | Logout

### Open Orders for Steve Ling

| Order Date            | Amount    | Order Type | Items                                                                                                                                                                                 |                            |
|-----------------------|-----------|------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------|
| *** ACCESS DENIED *** | \$164.25  | APPL       | <ul style="list-style-type: none"> <li>• 2 of: Gap personal jean</li> <li>• 1 of: denim front-slit skirt Gap</li> <li>• 1 of: Hooded Pullover Fleece Sweatshirt</li> </ul>            | <a href="#">Edit Order</a> |
| *** ACCESS DENIED *** | \$356.65  | APPL       | <ul style="list-style-type: none"> <li>• 2 of: Lands End Athletic Slides</li> <li>• 1 of: Hush Poppies Angella II</li> <li>• 1 of: Debra Sandal at Nodstrom</li> </ul>                | <a href="#">Edit Order</a> |
| *** ACCESS DENIED *** | \$1679.65 | APPL       | <ul style="list-style-type: none"> <li>• 1 of: Burberry Nova Check Hobo</li> <li>• 1 of: Prada Patent Leather Handbag</li> <li>• 1 of: Prada tote</li> </ul>                          | <a href="#">Edit Order</a> |
| *** ACCESS DENIED *** | \$106.65  | APPL       | <ul style="list-style-type: none"> <li>• 1 of: Osh Kosh Lt Lilac Poplin Jumper Dress</li> <li>• 1 of: Guess Garden Denim Skirt</li> <li>• 1 of: Gap denim front-slit skirt</li> </ul> | <a href="#">Edit Order</a> |

**Liquid Data 8.5 Demonstration Options**

Query Response Time: 938 ms.

[Show SQL Report](#)  
[Show Liquid Data Concepts slide](#)

# Integrating with AquaLogic Service Bus

This section covers the following topics:

- [“Introduction” on page 11-1](#)
- [“Integration Pre-Requisites” on page 11-2](#)
- [“Integrating with AquaLogic Service Bus: Main Steps” on page 11-2](#)
- [“Enabling Elements for Access Control” on page 11-3](#)
- [“Creating the WebLogic Server SSM Configuration” on page 11-3](#)
- [“Configuring Policy for ALSB Resources” on page 11-4](#)

## Introduction

AquaLogic Service Bus 2.5 (ALSB) is a configuration-based, policy-driven Enterprise Service Bus. It facilitates a loosely coupled architecture, facilitates enterprise-wide reuse of services, and centralizes management. AquaLogic Enterprise Security can be used to manage access control to ALSB’s runtime resources, using the ALES WebLogic Server 9.x Security Service Module.

ALES secures only the runtime resources of ALSB, in general those resources that ALSB passes to `isAccessAllowed()`; it does not secure the resources used during ALSB configuration, such as the ALSB console.

# Integration Pre-Requisites

Before you begin, you must ensure that the following pre-requisites are satisfied:

- WebLogic Server 9.1 or 9.2
- WebLogic Server 9.x Security Service Module
- You must have access to an Administration Console that is running on the AquaLogic Enterprise Security 2.2 Administration Server on either the local machine or a remote machine.
- AquaLogic Service Bus 2.5

## Integrating with AquaLogic Service Bus: Main Steps

This section describes how to integrate AquaLogic Enterprise Security with the AquaLogic Service Bus. Once integrated, you can use the AquaLogic Enterprise Security Administration Console to write and deploy a set of authorization and role mapping policies to protect ALSB runtime resources.

To integrate AquaLogic Enterprise Security with AquaLogic Service Bus, perform the following tasks:

1. Install the ALES Administration Server, as described in [Installing the Administration Server](#).
2. Install the WebLogic Server 9.x SSM, as described in [Installing Security Service Modules](#).
3. Use the ALSB Administration console to enable the elements to which you want to control access, as described in [“Enabling Elements for Access Control”](#) on page 11-3.
4. Configure the WebLogic Server 9.x SSM, as described in [“Creating the WebLogic Server SSM Configuration”](#) on page 11-3.
5. Create an instance of the WebLogic Server SSM, as described in [“Creating an Instance of the Security Service Module”](#) on page 11-3.
6. Enroll the instance of the WebLogic Server SSM, as described in [“Enrolling the Instance of the Security Service Module”](#) on page 11-3.
7. Configure security policies for ALSB, as described in [“Configuring Policy for ALSB Resources”](#) on page 11-4.

## Enabling Elements for Access Control

Before enabling your ALSB domain for ALES, open the ALSB Administration console:

1. In a browser, access the ALSB console at `http://<hostname>:<port>/sbconsole` and log in as an Administrator.
2. Click Create in the Change Center to create a session.
3. Click the Project Explorer to create a new project.
4. Select your project and add the appropriate WSDL resources. Click Save.

## Creating the WebLogic Server SSM Configuration

Securing ALSB with ALES employs the WebLogic Server 9.x SSM. Integration of ALES with ALSB is not supported for versions of WebLogic Server prior to WebLogic Server 9.1. Install the WLS SSM on the machines on which you have installed ALSB, as described in [Installing Security Service Modules](#).

Create a new WLS SSM configuration. An SSM configuration defines the set of security providers to use for adjudication, authentication, auditing, authorization, role mapping, and credential mapping services. For instructions on creating a SSM configuration, see [Configuring and Binding a Security Service Module](#) in [Installing Security Service Modules](#) and the Console Help. See also [Chapter 8, “Configuring the WebLogic Server 9.x SSM.”](#)

## Creating an Instance of the Security Service Module

Before starting a WebLogic Server Security Service Module, you must first create an instance of the WebLogic Server Security Service Module using the Create New Instance Wizard.

For information about creating an instance of a WebLogic Server Security Service Module, see [Creating an Instance of a Security Service Module](#) in [Installing Security Service Modules](#).

## Enrolling the Instance of the Security Service Module

You must have the ALES Administration Server running prior to enrolling the Security Service Module. For information about enrolling a security service module, see [Enrolling the Instance of the Security Service Module](#) in [Installing Security Service Modules](#).

# Configuring Policy for ALSB Resources

Developing a set of policies typically begins by determining which resources you need to protect and your access control requirements. You then create the identity directory, resources, groups, users, and roles that you will use to write policies to protect those resources. Next you write a set of authorization and role mapping policies to define access control on those resources. Finally, you deploy the set of policies to the WebLogic Server Security Service Module that you use to control access to your data services.

The ALES Administration Server installation includes a set of sample policies for BEA AquaLogic Service Bus, located at `BEA_HOME/ales22-admin/examples/policy/alsb_sample_policy`. You can import these sample policies and use them as a starting point for developing a full set of policies for your applications. For information about how to import the sample policies, see the README file in the sample directory and see also [Importing Policy Data](#) in the *Policy Managers Guide*.

This section covers the following topics:

- [“Creating the ALSB Application Resources” on page 11-4](#)
- [“Creating the ALSB Proxy Service Resources” on page 11-5](#)

## Configuring Resources and Privileges

This section describes how to use the ALES Administration Console to define the application resources that you will protect using ALES.

To configure resources, perform the following tasks:

- [“Creating the ALSB Application Resources” on page 11-4](#)
- [“Creating the ALSB Proxy Service Resources” on page 11-5](#)

### Creating the ALSB Application Resources

**Note:** You can choose any application name for your ALSB application.

To create application resources, use the Administration Console to perform the following steps:

1. Expand the Resources folder, and click Resources. The Resource page displays.
2. In the Resources page, select the Policy node, and click New. The Create Resource dialog box appears.

3. In the Name text box, enter `alsbrealm`, select Binding from the Type drop-down list box, and click Ok. The `alsbrealm` resource appears under the Policy node.
4. Select the `alsbrealm` resource and click Configure. The Configure Resource dialog box appears.
5. From the Type drop-down list box, select Binding Application, check the Distribution Point and Allow Virtual Resources check boxes, and click Ok.
6. Refer to [Table 11-1](#) and modify the configuration of the ASI Authorization provider and the ASI Role Mapper provider as described there.

**Table 11-1 ALSB SSM Configuration Modifications**

| Security Provider          | Configuration Setting                                                                                                                                                                                                                                                                            |
|----------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ASI Authorization Provider | <ol style="list-style-type: none"> <li>1. On the Details tab, set the Application Deployment Parent to <code>//app/policy/alsbrealm</code> and click Apply.</li> <li>2. On the Bindings tab, from the Bind drop-down menu, select <code>//app/policy/alsbrealm</code> and click Bind.</li> </ol> |
| ASI Role Mapper Provider   | <ol style="list-style-type: none"> <li>1. On the Details tab, set the Application Deployment Parent to <code>//app/policy/alsbrealm</code> and click Apply.</li> <li>2. On the Bindings tab, from the Bind drop-down menu, select <code>//app/policy/alsbrealm</code> and click Bind.</li> </ol> |

## Creating the ALSB Proxy Service Resources

Create resources in ALES corresponding to the ALSB Proxy Services. An ALSB Proxy Service is defined by a reference string and an operation string. The reference string uses this format:

```
ProxyService/<Project Name>/[Folder name]/<Proxy Service Name>
```

where:

- `ProxyService` is a string fixed in the `ALSBProxyServiceResource` instance.
- `Project Name` is the project name. Each proxy service is associated with an ALSB project.
- `Folder name` is an optional field and may be multiple. Folders are represented as: `/folder/subfolder...`
- `Proxy Service Name` is the proxy service name.

ALES resource definitions for ALSB use this format:

```
//app/policy/<binding app>/<Proxy Service App name>/ProxyService/<Project Name>/[Folder name]/<Proxy Service Name>
```

Table 11-2 describes how ALSB Proxy Service reference elements map to ALES resource and privilege elements:

**Table 11-2 ALSB Proxy Service Elements Represented in ALES Resources and Privileges**

| Resource/Privilege Element | Description                                                                                              |
|----------------------------|----------------------------------------------------------------------------------------------------------|
| binding app                | The ALES binding node name.                                                                              |
| Proxy Service app name     | The default application name, shared.                                                                    |
| ProxyService               | The ALES resource type.                                                                                  |
| Folder name                | The ALSB Proxy Service folder name.                                                                      |
| //priv/<operation>         | The operation field of the ALSB Proxy Service, representing one of the Web Services operations provided. |

For example, suppose you have an ALSB Proxy Service named `SampleProxyService`, which is associated with an ALSB project named `SampleProject` and has a Web Service action named `sayHello`. This service belongs to a folder named `Mortgage/ProxyService`. This `ALSBProxyServiceResource` instance is represented as:

```
reference:
ProxyService/SampleProject/Mortgage/ProxyService/SampleProxyService
operation: sayHello
```

The corresponding ALES resource (presuming the WLS SSM binding application is `swag/ALSB`) would be:

```
//app/policy/swag/ALSB/shared/ProxyService/SampleProject/Mortgage/ProxyService/SampleProxyService
```

The corresponding ALES privilege is:

```
//priv/sayHello
```

## Discovering Services

When developing policies for use with a Security Service Module, you can use the Discovery mode feature to help define your policy components. Instructions for using Discovery mode are provided in the [“Resource Discovery”](#) section in the *Policy Managers Guide*.



# Enabling SAML-based Single Sign-On

The ALES provides support for the producing and consuming SAML 1.1 assertions, and for sending/receiving them using the Browser/POST Profile.

This section covers the following topics:

- [“Overview” on page 12-1](#)
- [“Configuring ALES as a SAML Assertion Consumer” on page 12-2](#)
- [“Configuring ALES as a SAML Assertion Producer” on page 12-3](#)

## Overview

The use of SAML assertions allows servers in different domains to operate in a federation of trusted servers and grant access to users based on a single login to one of the servers. In a given federation, there are SAML ‘producers’ and SAML ‘consumers’. The SAML providers authenticate users and generate assertions attesting to the user’s identity. The SAML assertion can then be included in user requests to other servers in the federation, making additional logins unnecessary.

ALES SSMs allow a Microsoft IIS or Apache HTTP server to operate as a SAML producer or a SAML consumer (or both) and send/receive SAML assertions using the Browser POST Profile. Note that Browser Artifact Profile is not currently supported.

When set up as a SAML consumer, the SSM running on an IIS or Apache HTTP server will accept requests containing assertions and then use its SAML Identity Asserter to validate the assertions.

# Configuring ALES as a SAML Assertion Consumer

When serving as a SAML consumer, the SSM receives requests specifying a protected resource and SAML assertion attesting to the user's validity. The SSM's SAML Identity Asserter accepts the SAML token and returns the corresponding user. ALES will grant access to the resource based on any policies associated with the resource and/or users role.

To configure a IIS or Apache SSM as a SAML consumer:

**Note:** It is assumed that the necessary Resources and Policies governing access to protected resources have been established.

1. Using the Administration Console, create or use an existing SSM configuration defining a SAML Identity Asserter. The SAML Identity Asserter consumes SAML assertions and returns the corresponding authenticated subjects.

**Note:** The trusted keystore configured for the SAML Identity Asserter must contain the certificate used to sign the Assertion and the certificate that signed that certificate up to the trust anchor. If the trust anchor is a well known CA such as Verisign, the keystore does not have to contain the trust anchor certificate.

2. Install the ALES SCM and SSM on the IIS or Apache web server.
3. Create instances of the Web Service SSM and the Web Server SSM on the web server and configure the web server to call the SSM. Set the SSMs to use the certificate authority keystore. Set the password for the SSM to use when logging in to the ASI database.
4. Set up a file to serve as the target POST URL. The file can be copied to the web server or referred to using a virtual server. It serves as a placeholder that alerts the SSM to receive a SAML assertion. It must be a valid HTML file, but requires nothing more than empty `<HTML>` and `<BODY>` tags. The SSM provides a template file named `SAMLIn.acc` (IIS) or `SAMLIn.html` (Apache) in the `templates` directory.
5. In SSM's `default.properties` file, enable the `set saml.incoming.enable` parameter to 'true'.

Example: `set saml.incoming.enable=true`

6. In SSM's `default.properties` file, set the `saml.incoming.url` parameter to the POST URL you established on the server (see step 4). Make sure you create a policy that allows POST to the SAML consumer URL.

Example: `saml.incoming.url=http://<server>/<dir>/SAMLIn.acc`

## Configuring ALES as a SAML Assertion Producer

When operating a SAML producer, the SSM will receive requests for a SAML assertion. The SSM's Authentication Provider authenticates the user and its SAML Credential Mapper returns a SAML assertion. The SSM then sends a response contain the SAML assertion using the Browser POST Profile.

By default, SAML assertions produced by ALES are 64-base encoded tokens identifying the principals. They include an XML Signature proving that the assertion has not been tampered with in transit from the sender to the provider, and will contain group information about the user if that information is available. Note that these assertions do *not* contain the certificate chain used for signing the assertion. It is up to the SAML consumer to notify the recipient of the certificate that can be used to verify the XML signature.

To configure a IIS or Apache SSM as a SAML Producer:

1. Using the Administration Console, create a SSM configuration defining an a Authentication Provider and a SAML Credential Mapper.
2. Install the ALES SCM and SSM on the IIS or Apache web server.
3. Create instances of the Web Service SSM and Web Server SSM on the web server. Set the SSMs to use the certificate authority keystore. Set the password for the SSM to use when logging in to the ASI database.
4. Configure the IIS or Apache web server to integrate with the SSM.
5. Configure the script for handling the Browser POST to the SAML consumer.

**Notes:** The SSM's `template` directory contains a file named `SAMLXfer.acc` (IIS) or `SAMLXfer.dhtml` (Apache) that can be used.

Make sure you create a policy allowing Everyone access to the script file.



# Enabling SPNEGO-based Single Sign-on

This section covers the following topics:

- [“Configuring Single Sign-On with Microsoft Clients”](#) on page 13-1
- [“Configuring Active Directory Authentication”](#) on page 13-5

## Configuring Single Sign-On with Microsoft Clients

Using a Single Pass Negotiate Identity Asserter shipped with AquaLogic Enterprise Security, 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.

For more information, see the following topics:

- [“Requirements”](#) on page 13-2
- [“Enabling a Web Service or Web Application”](#) on page 13-3

# 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
- AquaLogic Enterprise Security application server
  - AquaLogic Enterprise Security installed
  - WebLogic 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 13-3](#)
- [“Configure the Client .NET Web Service” on page 13-7](#)
- [“Configure the Internet Explorer Client Browser” on page 13-8](#)

## 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.

For configuration instructions, see the following topics:

- [“Configuring the SPNEGO Security Provider” on page 13-3](#)
- [“Editing the Descriptor File” on page 13-3](#)

## Configuring the SPNEGO Security 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.

## Editing the Descriptor File

[Listing 13-1](#) shows a sample `web.xml` file for a protected WebLogic Web Service resource (Conversation) with the login configuration set to `CLIENT-CERT`.

### Listing 13-1 Sample Web.xml File

---

```
<!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. Continuing the example, [Listing 13-2](#) shows a sample `weblogic.xml` file.

### Listing 13-2 Sample `weblogic.xml` File

---

```

<?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](#) in the BEA WebLogic Server 8.1 documentation set available on the Web at <http://e-docs.bea.com/wls/docs81/>.

# Configuring Active Directory Authentication

To configure Active Directory authentication, perform the following tasks:

- [“Utility Requirements” on page 13-5](#)
- [“Configuring and Verifying Active Directive Authentication” on page 13-5](#)
- [“Configure the Active Directory Authentication Provider” on page 13-7](#)
- [“Configure the Client .NET Web Service” on page 13-7](#)
- [“Configure the Internet Explorer Client Browser” on page 13-8](#)

## Utility Requirements

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`

## Configuring and Verifying Active Directive Authentication

The first three steps of this procedure assume that 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 instance 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 13-7](#). For Internet Explorer configuration, complete the steps described in [“Configure the Internet Explorer Client Browser” on page 13-8](#).

## 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.

### Configure the Client .NET Web Service

If you are configuring the client .NET web service, perform the following steps:

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, perform the following steps:

- [“Configure the Sites” on page 13-8](#)
- [“Configure Intranet Authentication” on page 13-8](#)
- [“Verify the Proxy Settings” on page 13-9](#)
- [“Set the Internet Explorer 6.0 Configuration Settings” on page 13-9](#)

## 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 the Include all sites that bypass the proxy server check box 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.



# Authorization Caching

This section covers the following topics:

- [“Understanding Authorization Caching” on page 14-1](#)
- [“Configuring Authorization Caching” on page 14-2](#)
- [“Authorization Caching Expiration Functions” on page 14-3](#)

## Understanding 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 calls are made by the same caller. 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, the authorization cache automatically invalidates 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 14-1](#) lists the switches affect the authorization cache.

**Table 14-1 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	Specifies the number of seconds that authorization decisions for a user will be cached in memory. Upon expiration, the cached information is cleared and then updated if the user makes a subsequent request.  While increasing this value can improve performance, it may also reduce security by making authorization decisions based on outdated information.
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 14-2](#) can be entered as advanced configuration properties to further tune the cache.

**Table 14-2 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 14-2 Advanced Configuration Properties (Continued)**

Setting	Default Value	Description
ASI.AuthorizationCacheDynamicAttribute Limit	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. Because 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 policies that make explicit use of internally computed time values or custom evaluation plug-ins.

[Table 14-3](#) lists the internally computed time values. If these values are referenced in a policy, you should also explicitly set the cache validity for the policy.

**Table 14-3 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
dayofmonthgmt	integer	1–31
dayofyear	integer	1–366
dayofyeargmt	integer	1–366

**Table 14-3 Time Values Used with Expiration Functions (Continued)**

<b>Credential</b>	<b>Value</b>	<b>Range or Format</b>
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”

[Table 14-4](#) lists the expiration functions for the authorization cache. You can use these functions to 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. These functions correspond roughly to each of the internally computed time types.

**Table 14-4 Expiration Functions**

<b>Function</b>	<b>Argument</b>	<b>Description</b>
valid_for_mseconds	integer	Valid for a given number of milliseconds
valid_for_seconds	integer	Valid for a given number of seconds

**Table 14-4 Expiration Functions (Continued)**

<b>Function</b>	<b>Argument</b>	<b>Description</b>
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
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.

**Table 14-4 Expiration Functions (Continued)**

Function	Argument	Description
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.
valid_until_dayofmonth_gmt	integer	Valid until the specified day of the month the evaluation is performed in GMT time.
valid_until_dayofweek_gmt	Dayofweek_type	Valid until the specified day of the week the evaluation is performed in GMT time.

For example, if you had the following policy:

```
GRANT(//priv/order, //app/resturant/breakfast, //sgrp/customers/allusers/)
if hour < 11;
```

When authorization caching is enabled, you write the policy 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 policy is cached in the provider until 11:00 AM, at which time, it expires. Not calling `valid_until_hour` argument results in this policy being cached until the next policy distribution. Therefore, if you are using authorization caching, it is important to update your time dependent policies appropriately.