BEA AquaLogic™ Enterprise Security

Introduction

Version 2.2
Revised: June 2006

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.

Third-Party Software License Agreement

Sun Microsystems, Inc.’s XACML implementation v2.0

Copyright © 2003-2004 Sun Microsystems, Inc. All Rights Reserved.

This product includes Sun Microsystems, Inc.’s XACML implementation v2.0, which is governed by the following terms:

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistribution of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistribution in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

Neither the name of Sun Microsystems, Inc. or the names of contributors maybe used to endorse or promote products derived from this software without specific prior written permission.

This software is provided "AS IS," without a warranty of any kind. ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE HEREBY EXCLUDED. SUN MICROSYSTEMS, INC. ("SUN") AND ITS LICENSORS SHALL NOT BE LIABLE FOR ANY DAMAGES SUFFERED BY LICENSEE AS A RESULT OF USING, MODIFYING OR DISTRIBUTING THIS SOFTWARE OR ITS DERIVATIVES. IN NO EVENT WILL SUN OR ITS LICENSORS BE LIABLE FOR ANY LOST REVENUE, PROFIT OR DATA, OR FOR DIRECT, INDIRECT, SPECIAL, CONSEQUENTIAL, INCIDENTAL OR PUNITIVE DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF THE USE OF OR INABILITY TO USE THIS SOFTWARE, EVEN IF SUN HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

You acknowledge that this software is not designed or intended for use in the design, construction, operation or maintenance of any nuclear facility.

For all third-party software license agreements, see the 3rd_party_licenses.txt file, which is placed in the \ales22-admin directory when you install the AquaLogic Enterprise Security Administration Server.

Trademarks and Service Marks

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

Contents

1. Introduction and Roadmap

Audience and Scope 1-1

Guide to This Document 1-2

Related Information 1-2

New and Changed Features in This Release 1-3

Performance Statistics. 1-3

Web Services Client Authorization Cache 1-4

Authorization Through XACML 1-4

New Installation Features 1-4

Support for Non-U.S.-English Locales. 1-4

WebLogic Server 9.x SSM Added 1-5

WebLogic 9.x Security Providers Supported 1-5

Web Services SSM Now Supports Microsoft .Net and WebLogic Workshop 9.0 Clients 1-5

WebLogic Portal 9.2 Integration Supported 1-6

AquaLogic Service Bus Integration Supported 1-6

Additional Platforms Supported 1-6

New Examples 1-7

Adding Application Context from the BLM API 1-8

2. Overview of BEA WebLogic Enterprise Security

What Is BEA AquaLogic Enterprise Security? 2-1

Key Features 2-3

What is the Problem? 2-5
What is the Solution? 2-8
Distributed Computing Security Infrastructure 2-10
How Our Solution Benefits You 2-13
Application Developers 2-14
Server and Application Administrators 2-14
Security Developers 2-14
Security Architects 2-15
Standards 2-15
Examples 2-17

3. Architecture Overview

Administration Server 3-1
Service Control Manager 3-3
Security Service Modules 3-4
WebLogic Server 8.1 Security Service Module 3-5
WebLogic Server 9.x Security Service Module 3-6
Web Server Security Service Module 3-7
Web Services Security Service Module 3-8
Java Security Service Module 3-9

4. Security Services

Authentication and Identity 4-1
Authentication Service 4-2
Types of Authentication 4-2
Username and Password Authentication 4-3
Perimeter Authentication 4-3
Identity Assertion 4-5

Identity Assertion Service 4-6
Role Mapping 4-7
Role Mapping Service 4-7
Authorization 4-8
Authorization Service 4-9
User Directories 4-10
Resources 4-11
Authorization Policies and Role Mapping Policies 4-12
ContextHandlers 4-13
Adjudication 4-13
Adjudication Service 4-14
Auditing 4-14
Auditing Service 4-14
Performance Statistics 4-15
Credential Mapping 4-15
Credential Mapping Service 4-16
Security Service Providers 4-16

5. Security Administration

Managing Security 5-1
Security Configuration 5-3
Resources 5-4
Resource Attribute 5-6
Privilege 5-7
Privilege Group 5-7
Identity 5-8
Users 5-9
Groups 5-10

Identity Attributes 5-11
Roles. 5-12
Policy. 5-12
Role Mapping Policies 5-12
Authorization Policies 5-13
Role Mapping Policy Reports 5-13
Authorization Policy Reports. 5-13
Declarations. 5-14
Deployment 5-14

Index

CHAPTER 1

Introduction and Roadmap

The following sections describe the content and organization of this document, as well as new features in AquaLogic Enterprise Security 2.2:

- "Audience and Scope" on page 1-1
- "Guide to This Document" on page 1-2
- "Related Information" on page 1-2
- "New and Changed Features in This Release" on page 1-3

Audience and Scope

This document summarizes the features of the BEA AquaLogic™ Enterprise Security products and presents an overview of the architecture and capabilities of the security services. It provides a starting point for understanding the family of BEA AquaLogic Enterprise Security products.

The document is intended for all users of the BEA AquaLogic Enterprise Security product family, including:

- Business analysts
- Security architects
- Security developers
- Application developers
- Server and application administrators

The BEA AquaLogic Enterprise Security products incorporate many terms and concepts that are defined in the glossary. BEA recommends that you review the terminology to become familiar with the various terms and concepts.

Guide to This Document

This document is organized as follows:

- This chapter, Introduction and Roadmap, outlines the scope of this document and summarizes new features in this release of AquaLogic Enterprise Security.
- Chapter 2, "Overview of BEA WebLogic Enterprise Security," describes BEA AquaLogic Enterprise Security products, services, features, and functionality.
- Chapter 3, "Architecture Overview," describes the BEA AquaLogic Enterprise Security product architecture and its components.
- Chapter 4, "Security Services," describes security services, including auditing, authentication, authorization and role mapping, and credential mapping.
- Chapter 5, "Security Administration," describes policies and how you use the Administration Server to design policies.

Related Information

The BEA corporate web site provides all documentation for BEA AquaLogic Enterprise Security, including:

- *WSDL Documentation for the Web Service Interfaces*—Provides reference information for the Web Services Interfaces that are part of this release of BEA AquaLogic Enterprise Security.
- *Administration and Deployment Guide*—Provides step-by-step instructions for performing various administrative tasks.
- *Integrating ALES with Application Environments*—Describes how to integrate AquaLogic Enterprise Security into application environments.
- *Policy Managers Guide*—Describes how to write access control policies for BEA AquaLogic Enterprise Security, and describes how to import and export policy data.
- *Installing Security Services Modules*—Describes how to install ALES Security Services Modules, including the Web Services Security Service Module.

- *Developing Security Providers* —Describes how to develop custom security providers.
- *Javadocs for Security Service Provider Interfaces*—Provides reference information for the Security Service Provider Interfaces that are part of this release of BEA AquaLogic Enterprise Security.
- *Programming Security for Java Applications*—Describes how to implement security in Java applications. It include descriptions of the Security Service Application Programming Interfaces and programming instructions for implementing security in Java applications.
- *Javadocs for Java API*—Provides reference information for the Java Application Programming Interfaces that are part of this release of BEA AquaLogic Enterprise Security.

New and Changed Features in This Release

This release of AquaLogic Enterprise Security has several new and changed features.

Performance Statistics

The performance statistics feature enables the collection of authentication and authorization data for troubleshooting and performance analysis. The performance statistic feature is controlled by an auditing security provider, the PerfDBAuditor provider, which you configure and enable for each Security Service Module for which you want to gather statistics.

The performance statistics feature gathers the following information, for each SSM configuration ID and host name, aggregated for each time interval specified by the Performance Statistics Interval setting:

- Number of requested and successful authentications
- Number of requested and successful authorizations
- Average latency of an authentication request, in milliseconds
- Average latency of an authorization request (the duration of calls to isAccessAllowed from start to end), in milliseconds
- For any user attribute required for policy evaluation or response:
  - Average retrieval time, in milliseconds
  - Total number of retrievals
- For each external function called during evaluation:

- Average execution time, in milliseconds
- Total number of calls

See *Administration and Deployment Guide*.

Web Services Client Authorization Cache

A client-side Authorization cache allows an application that is using the Web Services SSM to take advantage of in-process caching to achieve performance improvements when making authorization calls.

The Web Services Authorization cache is implemented as an Axis handler. The handler implementation allows you to add and remove the Authorization cache without affecting existing code. The Authorization cache can be configured through a Java API. If you do not use the configuration API to configure the cache, the default values for the cache are used.

See Programming Security for Web Services.

Authorization Through XACML

External applications can ask authorization questions through use of the XACML protocol. This capability is supported only in the Web Services SSM.

The XACML service is implemented as an extension to the existing Authorization Service in the Web Service SSM, and uses the same configuration and administration scripts of the Web Service SSM. The XACML service is silently installed together with the Web Service SSM.

See Programming Security for Web Services for additional information.

New Installation Features

This release of AquaLogic Enterprise Security includes a utility to help you upgrade from AquaLogic Enterprise Security 2.1. See Installing the Administration Server for additional information.

As of version 2.2 of ALES, the user who installs the Administration Server and SSMs does not require administrator privileges on a Windows platform, or root access on a Sun Solaris or Linux platform. The installation procedures set the file and directory permissions based on the user who runs the installer.

Support for Non-U.S.-English Locales

You can install the Administration Server and SSMs in non-US-English locales. If you install AquaLogic Enterprise Security on a machine that is part of a non-US-English locale, it is assumed that all other components with which AquaLogic Enterprise Security communicates will also be installed on the same non-US-English locale, including the policy RDBMS and the authentication source (LDAP or RDBMS).

WebLogic Server 9.x SSM Added

The WebLogic Server 9.x Security Service Module (SSM) integrates AquaLogic Enterprise Security with BEA WebLogic Server 9.1 and 9.2. This SSM uses a different security framework from the one used in the WebLogic Server 8.1 SSM and the other AquaLogic Enterprise Security SSMs. When you install the WebLogic Server 9.x SSM, AquaLogic Enterprise Security uses the WebLogic Server 9.x security framework. As a consequence, when you use the WebLogic Server 9.x SSM, you configure security providers in the WebLogic Administration Console, rather than in the AquaLogic Enterprise Security Administration Console. You still use the AquaLogic Enterprise Security Administration Console to create resources and to write security policies for all SSMs, and to configure providers in SSMs other than the WebLogic Server 9.x SSM. You must also use the AquaLogic Enterprise Security Administration Console to configure the deployment parent in the ASI Authorizer and ASI Role Mapper providers.

See Integrating ALES with Application Environments for additional information.

WebLogic 9.x Security Providers Supported

This release of AquaLogic Enterprise Security supports any of the WebLogic 9.x security providers. However, the Security Service Module for WebLogic Server 9.x is configured differently, as described in Integrating ALES with Application Environments.

You can also use the WebLogic Server 9.x WebLogicMBeanMaker to create any of the security provider types described in Developing WebLogic Security Providers.

Web Services SSM Now Supports Microsoft .Net and WebLogic Workshop 9.0 Clients

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*`\ales22-ssm\examples`. For this release, the following new examples are included:

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.

WebLogic Portal 9.2 Integration Supported

This release of AquaLogic Enterprise Security allows you to integrate with WebLogic Portal 9.2 server and portal applications, resulting in an enhanced set of security services for use in protecting WebLogic Portal. 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.

See Integrating ALES With Application Environments for additional information.

AquaLogic Service Bus Integration Supported

This release of AquaLogic Enterprise Security allows you to integrate AquaLogic Service Bus 2.5. 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. You can use AquaLogic Enterprise Security 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.

See Integrating ALES With Application Environments for additional information.

Additional Platforms Supported

This release of AquaLogic Enterprise Security supports the following additional platforms:

- WebLogic Server 9.1 and 9.2 are supported as platforms for the AquaLogic Enterprise Security Administration Server. An AquaLogic Enterprise Security Administration Server running on WebLogic Server 9.1 or 9.2 works with both WLS 8.1 and WLS 9.x SSMs.

- WebLogic Server 9.1 and 9.2 are supported as platforms for SSMs. The SSMs run with either the Sun JVM or the JRockit JVM that ship with WebLogic Server.
- Tomcat 5.5.15 is supported as a platform for AquaLogic Enterprise Security Administration Server.
- AIX 5.3 is supported as a platform for SSMs. AquaLogic Enterprise Security supports the WebLogic Server (8.1x/9.x), Java, and Web Services SSMs on AIX 5.3.
- JRockit JVM is supported. AquaLogic Enterprise Security supports the use of both the Sun JVM and the JRockit JVM. Use of JRockit as a JVM is supported only for 9.x platform products. AquaLogic Enterprise Security supports the JRockit release that ships with WLS 9.x, which is based on the 1.5 version of the Sun JVM. JRockit is supported only on Intel Pentium-compatible hardware (Windows and Red Hat Linux).

New Examples

This release of AquaLogic Enterprise Security includes the following new examples. Each example has a readme file that describes its function.

- For admin:
  - *BEA_HOME*`\ales22-admin\examples\EJBAppExample`
  - *BEA_HOME*`\ales22-admin\examples\arme_extension`
  - *BEA_HOME*`\ales22-admin\examples\DBSetupKit`
  - *BEA_HOME*`\ales22-admin\examples\policy`
  - *BEA_HOME*`\ales22-admin\examples\policymgtwsapi`
  - *BEA_HOME*`\ales22-admin\examples\tools`
- For SSMs:
  - For Java SSM:
    - *BEA_HOME*`\ales22-ssm\java-ssm\examples\AttributeRetriever`
    - *BEA_HOME*`\ales22-ssm\java-ssm\examples\JavaAPIExample`
    - *BEA_HOME*`\ales22-ssm\java-ssm\examples\QueryResources`
    - *BEA_HOME*`\ales22-ssm\java-ssm\examples\tools`
  - For Web Services SSM:
    - *BEA_HOME*`\ales22-ssm\webservice-ssm\examples\ArmeMonitor`

- *BEA_HOME*\ales22-ssm\webservice-ssm\examples\JavaWebServiceClient
- *BEA_HOME*\ales22-ssm\webservice-ssm\examples\SsmNet
- *BEA_HOME*\ales22-ssm\webservice-ssm\examples\SsmWorkshop
- *BEA_HOME*\ales22-ssm\webservice-ssm\examples\tools
- *BEA_HOME*\ales22-ssm\webservice-ssm\examples\XACMLClient

– For WLS 9 SSM:

- *BEA_HOME*\ales22-ssm\wls9-ssm\examples\ALESEnabledWLP92Domain
- *BEA_HOME*\ales22-ssm\wls9-ssm\examples\tools

– For WLS 8 SSM:

- *BEA_HOME*\ales22-ssm\wls-ssm\ALESEnabledWLPDomain
- *BEA_HOME*\ales22-ssm\wls-ssm\ALESEnabledWLSCluster
- *BEA_HOME*\ales22-ssm\wls-ssm\ArmeMonitor
- *BEA_HOME*\ales22-ssm\wls-ssm\ResourceConverter
- *BEA_HOME*\ales22-ssm\wls-ssm\SAMLServletExample
- *BEA_HOME*\ales22-ssm\wls-ssm\taglib
- *BEA_HOME*\ales22-ssm\wls-ssm\tools

Adding Application Context from the BLM API

In this release of AquaLogic Enterprise Security, the BLM API has been enhanced to allow you to send an Application Context to the auditing service.

The following BLM API methods have been added to provide for the Application Context:

- `BLMManager.create(java.util.Hashtable credentials, java.util.Hashtable appCtx)`. This method creates an instance of the BLMContextManager and initializes the BLMContextManager with an Application Context. The BLM then adds the Application Context data to all auditing messages associated with this BLM Context sent to the Audit provider.

- `BLMContextManager.setApplicationContext(Hashtable appCtx)`. This method replaces an existing application context with the new one provided. (You must have called `BLMManager.create(java.util.Hashtable credentials, java.util.Hashtable appCtx)` method prior to calling `setApplicationContext(Hashtable appCtx)`. All

subsequent audit messages associated with this BLM Context have the Application Context added to them when they are sent to the Audit provider.

- `BLMContextManager.clearApplicationContext()`. This method clears the Application Context associated with this BLM Context so that it is no longer included with audit messages sent to the Audit provider.

CHAPTER 2

Overview of BEA WebLogic Enterprise Security

This document summarizes the features of the BEA AquaLogic® Enterprise Security™ products and presents an overview of the architecture and capabilities of the security services. It provides a starting point for understanding the family of BEA AquaLogic Enterprise Security products and security infrastructure.

This chapter covers the following topics:

- "What Is BEA AquaLogic Enterprise Security?" on page 2-1
- "Distributed Computing Security Infrastructure" on page 2-10
- "How Our Solution Benefits You" on page 2-13
- "Standards" on page 2-15
- "Examples" on page 2-17

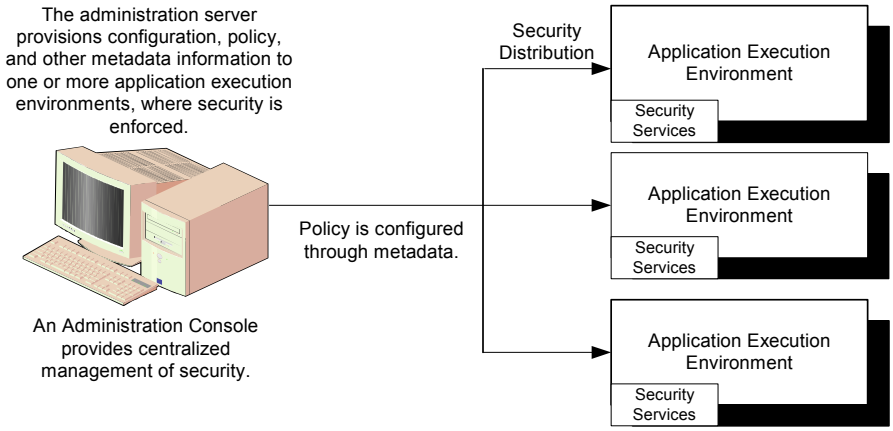
What Is BEA AquaLogic Enterprise Security?

BEA AquaLogic Enterprise Security (ALES) is a fine-grained entitlements engine that allows the user to centrally define and manage policy to control access for both application software components (for example, URLs, EJBs, EJB methods, and so on.) as well as the application business objects (for example, accounts and patients records) that make up the application. Policy is evaluated and enforced locally in the application container so application context can be included as part of the access decision.

A major benefit of ALES as an entitlements engine is that it allows you to remove security logic from the application. Thus, you can take security decisions out of the hands of your developers and define access policy consistently across multiple applications. ALES is also a security infrastructure product that provides a platform for creating a security service layer that can be used by multiple applications. It provides a standard set of security services including authentication, authorization, role mapping, auditing, and credential mapping. The product is distributed in that access decisions are made and enforced at runtime in the application container through a set of Security Service Modules (SSMs).

ALES consists of two major components—the Administrative Server and a set of SSMs. The Administrative Server can be accessed through a browser-based console that you can use to define your users, groups, and roles, the resources in your application, the policies that control access, and the security configurations of the SSMs (see Figure 2-1). ALES also includes Java and Web Services Interfaces that provide administration APIs. You can use these APIs to incorporate ALES administrative functionality into your own applications or even to write your own administrative console. The Administration Server supports delegated administrate in that allows you to delegate administrative functions to other users. Further, it provides a policy analysis function that enables you to analyze your policies to see which policies apply to specific resources, users, groups, and roles. To assist you in defining the access control policies, the Administration Server provides a resource discovery feature that you can use to automatically generate a list of application resources and to create a set of default access policies. Finally, to assist you in transporting your access policies from one Administration Server to another, the Administration Server provides policy exporting and importing tools.

Figure 2-1 Typical Application Execution Environment



Note: The BLM API supports all of the functionality supported by the Administration Console.

When a security configuration and/or policy is changed, you must distribute it so as to take effect throughout the enterprise, across multiple application execution environments. An open standards-based design allows customers, integrators and vendors to develop and incorporate their own custom security services. And, common security functions can be leveraged by applications throughout the enterprise.

The SSMs plug into the application container and intercept requests, enforce access policies, and make access control decisions in real time. SSMs get updated policy information through a real-time policy distribution mechanism that keeps policy sets up to date and consistent between SSMs. The security framework, the heart of the SSM, provides a set of security services including authentication, authorization, role mapping, auditing and credential mapping. ALES provides SSMs for the BEA WebLogic Platform products and for popular Web servers, such as Microsoft Internet Information Services (IIS) and Apache. Java and Web Services Interfaces are also provided that are language and infrastructure neutral. Applications can invoke the ALES security services directly through either the Java or Web Services APIs. In summary, the ALES SSMs provide the runtime enforcement of policy and a powerful security services layer and can be used as an integration platform for multiple security products in your infrastructure.

Key Features

The key features of the BEA AquaLogic Enterprise Security product include:

- ***Support for securing the enterprise***, including Security Service Modules for: BEA WebLogic Platform 8.1, Internet Information Services (IIS), Apache, web services, and Java applications.
- ***Simplified infrastructure***, that improves information technology efficiency because the application security infrastructure can be leveraged by applications across the enterprise. This feature effectively reduces integration costs and provides stronger investment protection in third-party security technologies through a standards-based architecture.
- ***Centralized policy***, leads to lower administration costs. Security administrators can define and deploy access policies without the need for re-coding or re-deploying applications. Policies are all managed through a central Administration Console. In addition, policy inquiry functions allow the administrator to validate access control policy implementations prior to deployment.
- ***Delegation of administrative and user privileges***, allows privileges granted to one user to be assigned to another user under certain conditions, or constraints.
- ***BLM API for policy definition***, provides programmatic access to the ALES policy management infrastructure. This is a Java API that uses SOAP to communicate with the central ALES management services. The API supports all of the functionality of the Administration Console including the creation and management of users, groups, roles, resources, resource authorization policies, and policy distribution and the security configuration and distribution. The BLM API examples (BEA_HOME\ales22-admin\examples\ policymgtwsapi) demonstrate how to use the BLM API for managing ALES roles and Policies.
  For more information on the BLM API, see the BLM API Javadocs.
- ***Java API***, that allows security developers to develop environment interfaces or even integrate an application security infrastructure into an application. These Java interfaces support the most commonly required security functions and are organized into services that are logically grouped by functionality.
- ***Web Services API***, that allows security developers to write custom applications that invoke AquaLogic Enterprise Security services through SOAP. These WSDL interfaces support the most commonly required security functions and are organized into services that are logically grouped by functionality.
- ***Role-based access control***, with integrated authentication, authorization and auditing. Administrators can design role-based access control (RBAC) policies to model their own business requirements, and then implement, test, and distribute them through a centralized Administration Server. You can include both resource and user attributes in your access control policies, along with other logical functions.

- ***A comprehensive and standards-based design***, to leverage Lightweight Directory Access Protocol (LDAP), Security Assertion Markup Language (SAML), Simple and Protected Negotiation mechanism (SPNEGO), eXtensible Access Control Markup Language (XACML), and Java. Additional support for standard security technologies, includes J2EE security technologies such as the Java Authentication and Authorization Service (JAAS), Java Secure Sockets Extensions (JSSE), and Java Cryptography Extensions (JCE).
- ***Maximum performance, scalability, reliability***, delivers a seamless, transparent experience for the user, while maintaining the necessary levels of application security through a distributed architecture.
- ***Support for customizing security services***, with step-by-step instructions, examples, and an open-standards model to help further reduce the cost of development. Security Service Provider Interfaces (SSPI) provide for rapid and simplified development of custom security services.

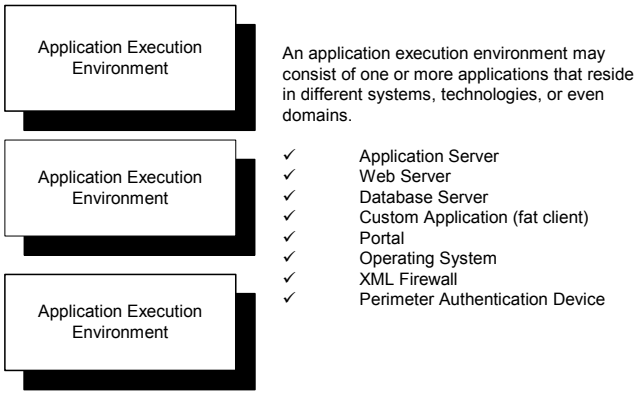
What is the Problem?

With the rush to build web-based applications and market services over the Internet, many developers had little comprehension of the security issues they may soon confront. Managing security is a huge challenge for any information technology organization that is providing new and expanded services to its employees, customers, and partners through both web-based and legacy applications. The advent of the Internet made protecting information and applications increasingly difficult to manage, monitor, and maintain. Financial transactions (ATM machines, bank transfers, credit card purchases and payments, stock market transactions), personal medical information (implementation of new Health Insurance Portability and Accountability Act or HIPPA regulations), Federal government facilities (Homeland Security affecting both military and civilian) provide only a few examples of areas where the concern for security has become essential and sometimes mandated by law.

Most applications require some form of security. As the complexity and volume of users and resources increases and with the rapid changes in business requirements that continue to evolve, the need for more stringent and robust security technologies becomes evident. To serve a worldwide network of users, an information technology organization must address the fundamental issues of maintaining the confidentiality, integrity and availability of the system and its data, providing the right information, to the right person, at the right time, across a diverse enterprise.

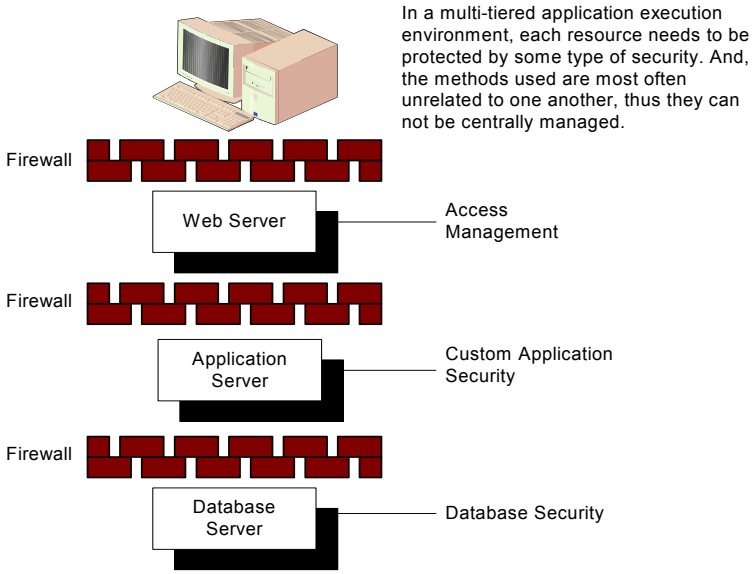
Because these applications often comprise a number of different components that may or may not reside on the same server or even in the same domain, policy management becomes extremely difficult and ensuring enterprise or regulatory compliance can prove impossible.

Figure 2-2 Application Execution Environment



A typical application execution environment is multi-tiered as shown in Figure 2-3 and may be distributed (vertically or horizontally) between multiple machines running on different operating system platforms. In this case, you must protect each tier or application component. The type of access control and technology for each one may be different and you need to be able to enforce security at each layer.

Figure 2-3 Multi-tiered Application Execution Environment



To address the multitude of potential breaches of security associated with multi-tiered environments, companies have had to purchase and integrate a variety of different and custom security technologies from a host of different vendors:

- Perimeter-based authentication for firewalls, with single sign-on, credential mapping, SAML, Virtual Private Networks (VPN), and web servers
- Digital certificates for server authentication
- Secure Sockets Layer (SSL) data encryption software or Hyper-Text Transfer Protocol (HTTPS) for secure transfer of data
- Access control and entitlement software

Integration of security technologies requires the application developer to embed these technologies and hard-code both integrated and unified security requirements within each application. Thus, as the number of applications increases, the expenses associated with application development and maintenance also increases. As a best practice, the application developer should *not* be responsible for developing, implementing, and managing security requirements.

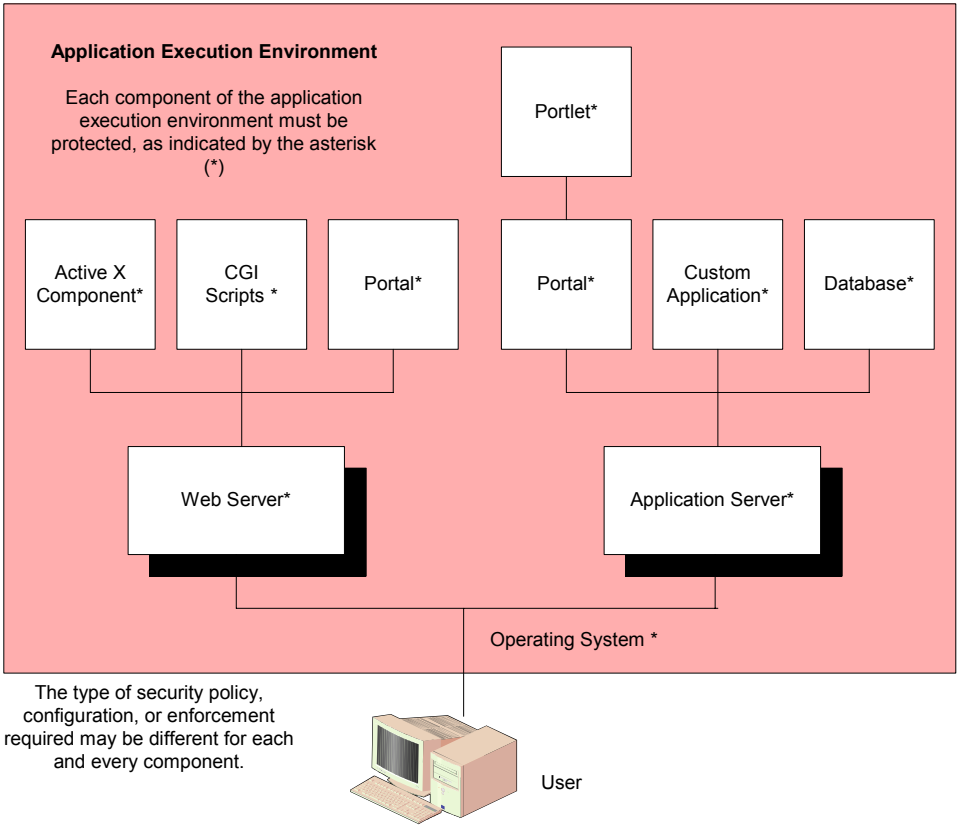
Early authorization implementations used static and inflexible approaches to define the different types of access granted or denied for a user. Because this type of implementation is extremely time-consuming (if only due to the number of users and the different types of user storage methods in use), it has become impractical for many implementations. Further, the cost of maintaining static first-generation security services can be exorbitant.

What is the Solution?

BEA has developed an application security infrastructure (ASI) that can be external to and isolated from the application itself. Using a services-oriented, policy-based architecture, you can replace the integrated security silo technologies and hard-coded policies. Figure 2-4 illustrates how a basic application execution environment can be protected using an integrated approach. Each component in the application requires protection, although the type of security typically varies.

A typical information technology environment consists of various types of servers–HTML, proxy, BEA WebLogic, Legacy, J2EE, and application–that access numerous LDAP and database servers containing information such as your user community (name, address, etc.). While the WebLogic platform servers provide application-level security for J2EE components, J2EE-based web services, portal and portlets (EJB, JSP/Servlet, JDNI, JDBC, JMS, MBeans), BEA AquaLogic Enterprise Security provides application security for additional platforms and web servers.

Figure 2-4 Integrated Application Security

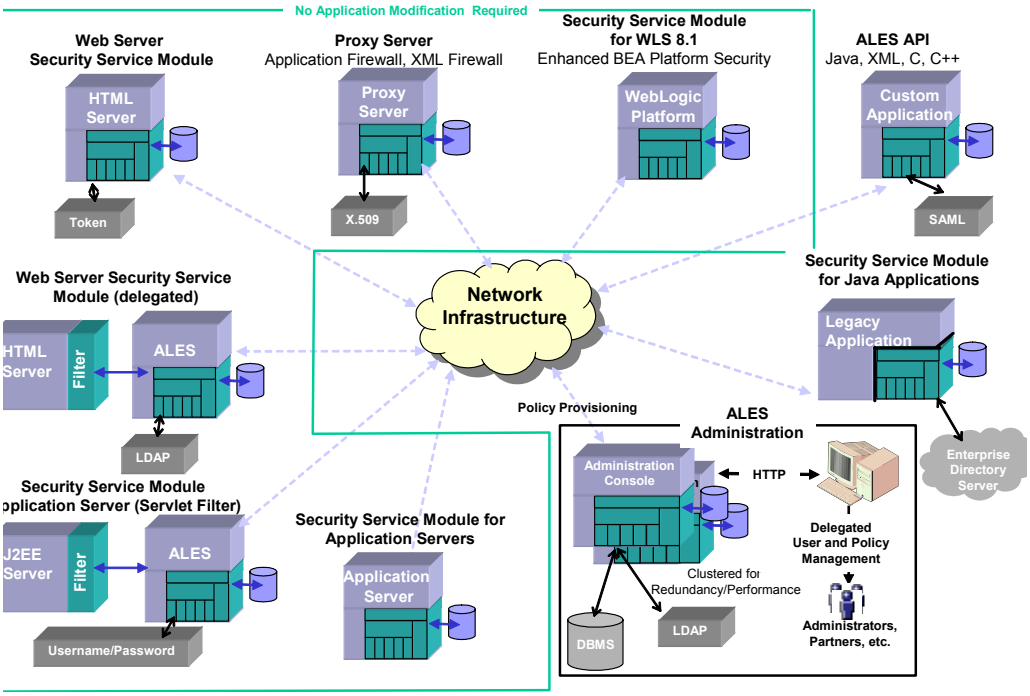


The open flexible architecture of the BEA AquaLogic Enterprise Security products provides advantages to all levels of users and introduces an advanced design for securing your applications. With distributed computing, applications must be integrated across the network, as shown in Figure 2-5. BEA AquaLogic Enterprise Security provides a distributed enterprise security solution that, together with clear and well-documented policies and procedures, can insure the confidentiality, integrity and availability of its applications and data.

Distributed Computing Security Infrastructure

Applications across the enterprise are built on a heterogeneous infrastructure with diverse resources. With an application security infrastructure as shown in Figure 2-5, the BEA AquaLogic Enterprise Security products support a fully distributed architecture; integrating all applications across the network.

Figure 2-5 Distributed Computing Security Infrastructure Vision



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

The type of security services you implement depends on the type of the application component you want to protect. You can use the set of security providers delivered with each Security Service Module to configure and enforce each security service or you can develop custom security providers.

The security services seek to provide ease of use, manageability for end users and administrators, and customizability for application developers and security developers. Administrators who configure and deploy applications can use the security providers included with the product that support most standard security functions or can create custom security providers. The product environments supported include WebLogic Server Versions 8.1 and 9.x, Internet Information Services (IIS) and Apache Web Servers, web services, and Java applications. BEA AquaLogic Enterprise Security will expand this family of Security Service Modules in subsequent releases.

- WebLogic Server 8.1 Security Service Module

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

- WebLogic Server 9.x Security Service Module

  The WebLogic Server 9.x Security Service Module integrates AquaLogic Enterprise Security with BEA WebLogic Server versions 9.1 and 9.2.

- IIS Web Server Security Service Module

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

- Apache Web Server Security Service Module

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

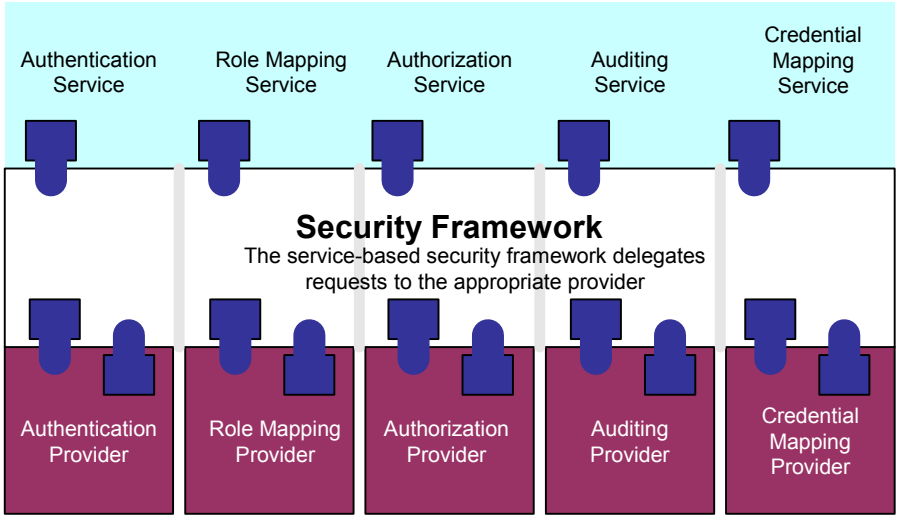
- Web Services Security Service Module

  An application programming interface (API) that allows security developers to write custom applications that invoke AquaLogic Enterprise Security services through SOAP. These interfaces support the most commonly required security functions and are organized into services that are logically grouped by functionality.

- Java Security Service Module

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

Figure 2-6 Security Framework



Each Security Service Module is delivered with a full set of security providers. Table 2-1 lists the types of providers that are available for configuration.

Table 2-1 BEA AquaLogic Enterprise Security Service Providers

Provider	Description
Authentication Provider	Supports open-standard support for SAML, SPNEGO, and X.509 identity assertion, and authentication support for Microsoft Windows NT, Microsoft Active Directory, Netscape LDAP, Novell LDAP, relational database, and OpenLDAP login modules.
Credential Mapping Provider	Maps credentials used by a legacy or any remote system. The application then uses the appropriate credentials to log in to a remote system on behalf of a subject that already authenticated to support single sign on.

Table 2-1 BEA AquaLogic Enterprise Security Service Providers (Continued)

Provider	Description
Authorization Provider	Controls access to resources based on the role and policy assigned to the requested resource. An access decision is the part of the authorization provider that actually determines whether a user has permission to perform an operation on a resource. Authorization providers secure access to resources and transactions, enabling an organization with increasingly complex user communities to provide secure finely-grained access to resources. Access decisions, provided through a role-based authorization provider, incorporate relevant environmental, contextual, and transaction-specific information, allowing security policies to support business processes throughout the organization. In addition, an adjudication provider resolves authorization conflicts when you configure multiple authorization providers.
Role Mapping Provider	Supports dynamic role associations by obtaining a computed set of roles granted to a requestor for a resource.
Auditing Provider	Provides an electronic trail of all transaction activity and can include changes to system configuration parameters, policy changes, and transactions. For each audit item, the information can include who, what, when, where, and sometimes why.

How Our Solution Benefits You

The modular BEA AquaLogic Enterprise Security service architecture provides specific benefits for:

- Application Developers
- Server and Application Administrators
- Security Developers
- Security Architects

Application Developers

Because most security for web applications and EJBs can be implemented by a system administrator, application developers do not need to be concerned about the details of securing the application, unless there are special considerations that must be addressed explicitly in the code. Security developers can also take advantage of BEA-supplied Application Programming Interfaces (APIs). Three sets of APIs are provided:

- The Java APIs are found in the `com.bea.security` package as described in Javadocs for Java APIs.
- The Web Service public interface APIs are found in the `SSM-SOAP Wsdldocs` package as described in Web Services API Wslddocs.
- The Security Provider SSPIs are found in the `weblogic.security` package as described in Javadocs for WebLogic Security Providers.

Server and Application Administrators

Administrators can use the security providers supplied as part of the product to implement an integrated solution. Administrators can use the Administration Server to define security roles and assign security policies to resources to create an authorization scheme that suites the needs of their business. In addition, the administrator can modify, test, and deploy the policy quickly and efficiently.

Security Developers

Third-party providers are integrating their products by using the Security Service Provider Interfaces (SSPI). As the underlying integration mechanism for security providers, the SSPI allows development of custom security providers. The SSPIs are available for Adjudication, Auditing, Authentication, Authorization, Credential Mapping, Identity Assertion, and Role Mapping. For information on the SSPIs, see Javadocs for Security Service Provider Interfaces.

This architecture allows security developers to provide integrated solutions that are easy to use. The result is a reduction in development requirements, which means an increased return on investment when implementing an enterprise security management solution. And, custom security services developed for WebLogic Platform 8.1 are compatible with the BEA AquaLogic Enterprise Security services.

Security Architects

A dynamic role-based policy architecture eliminates the need for application developers to design and implement business policy and embed it within each and every instance of an application. More efficient policy administration enables an organization to adapt quickly to dynamic business processes as security policies are designed, tested, deployed, and distributed quickly by security administrators with no coding required.

Delegated administration allows for centralized control and delegated labor, enabling administrators more familiar with the needs of a particular user constituency to implement business policy.

It also allows the implementation of policies across a much larger, more complex, user community with standard policy (for example, consisting of employees, business partners, customers). If a change to a policy is required, it can be distributed throughout the enterprise and take effect whenever desired. With BEA AquaLogic Enterprise Security products, if your application is already written to use some form of authentication or authorization schema, and the schema changes, no changes are required within the application.

Standards

BEA AquaLogic Enterprise Security products adhere to the following standards.

Table 2-2 BEA AquaLogic Enterprise Security Standards

XML Standard	Used to
SAML	Participate in SAML-based single sign-on (SSO) environment.
WSDL 1.1	The Web Services Description Language (WSDL) is an XML-based specification that describes a web service. A WSDL document describes web service operations, input and output parameters, and how a client application connects to the web service.
Java Standards	**Used to**
CertPath	Retrieve X.509 digital certificates associated with infrastructure protection; available for customer direct use.
KeyStore	Retrieve RSA private keys associated with X.509 digital certificates associated with infrastructure protection; available for customer direct use.

Table 2-2 BEA AquaLogic Enterprise Security Standards (Continued)

XML Standard	Used to
JSSE	Protect infrastructure network connections for establishment of mutual trust.
JCE	Integrate cryptographic libraries.
JAAS	Provide authentication service implementations.
Miscellaneous Standards	**Used to**
XACML 2.0	XACML, the eXtensible Access Control Markup Language, is an OASIS standard. XACML is a standard language for expressing access control, or authorization, policy, and a standard format for expressing queries over these policies.
X.509	Validate the identity of infrastructure components through digital certificates; supported as proof of identity for customer use.
LDAP v3	Retrieve configuration information from the Service Control Manager, and user identity and user attributes from an LDAP v3 directory server.
ISAPI	Support compliant runtimes for authentication, SAML single sign-on, and protection of hosted web pages.
FIPS 140	Support certification of the embedded cryptographic libraries used for cryptographic protection and TLS protocol.
TLS v1 and SSL	Protect network communication between infrastructure components.
JDBC	Provide access to database stores using the database provider.

Examples

This release of AquaLogic Enterprise Security includes the Administration Console and SSM examples shown inTable 2-1.

Table 2-3 AquaLogic Enterprise Security Examples

Example Location	Description
Admin Examples	
BEA_HOME\ales22-admin\arme_extension\README	Provides an example of an extension library for the Authorization and Role-Mapping engine (ARME) used by the ASI Authorization provider.
BEA_HOME\ales22-admin\DBSetupKit\README	Provides an example of installing and configuring a policy database for ALES. It also provides an example of installing an Oracle database.
BEA_HOME\ales22-admin\policy\aldsp_sample_policy\README.txt	Provides an example of importing the AquaLogic Data Services Platform sample policy.
BEA_HOME\ales22-admin\policy\alsb_sample_policy\README.txt	Provides an example of importing the AquaLogic Service Bus sample policy.
BEA_HOME\ales22-admin\policy\portal_sample_policy\README.txt	Provides an example of importing the WebLogic Portal sample policy.
BEA_HOME\ales22-admin\policymgtapi\readme.txt	Demonstrates how to use the ALES BLM JAVA API for managing ALES.
BEA_HOME\ales22-admin\policymgtwsapi\readme.txt	Demonstrates how to use the ALES BLM Web Service API for managing ALES roles and policies. This example demos how to establish a session with a remote BLM service, and how to do some simple operations using the BLM Web Service API.
BEA_HOME\ales22-admin\examples\EJBAppExample	Demonstrates the principles of an EJB application functioning under the ALES framework. The example shows how the additional layer provided by ALES can make an EJB application more secure by defining custom dynamic authorization policies without any changes in the application or the deployment descriptors.

Table 2-3 AquaLogic Enterprise Security Examples

Example Location	Description
Java SSM Examples	
BEA_HOME\ales22-ssm\java-ssm\examples\AttributeRetriever\README	This example shows how to implement a custom Attribute Retriever.
BEA_HOME\ales22-ssm\java-ssm\examples\JavaAPIExample\README	This example shows how to use Java SSM APIs to retrieve basic security services, and use them to do authentication, authorization, and auditing.
BEA_HOME\ales22-ssm\java-ssm\examples\QueryResources\README	This example shows how to get a list of the resources granted to a user. It also retrieves the list of resources denied to a user.
Web Services SSM Examples	
BEA_HOME\ales22-ssm\webservice-ssm\examples\ArmeMonitor\README	The sample ARME monitor is a Java Security Manager, which can run with Java SSM, WLS SSM or Web-Service SSM. This Security Manager creates a thread to monitor the ARME process status associated with this SSM.
BEA_HOME\ales22-ssm\webservice-ssm\examples\JavaWebServiceClient\README	This example shows how to connect to a Web Service SSM, retrieve basic security services, and use them to do authentication, authorization, and auditing.
BEA_HOME\ales22-ssm\webservice-ssm\examples\SsmNet\README	This sample demonstrates how to access ALES Web Service SSM through its published WSDL in the .NET environment. This sample contains two parts: 1. An abstraction layer that encapsulates the stub code generated from the WSDL. 2. A simple ASP.NET web application that covers all the interfaces exposed by the Web Service SSM.
BEA_HOME\ales22-ssm\webservice-ssm\examples\SsmWorkshop\readme.txt	This example is a simple web application based on page flow, which uses a service control generated from ALES SSM web service WSDL file by Workshop to protect it's content.

Table 2-3 AquaLogic Enterprise Security Examples

Example Location	Description
BEA_HOME\ales22-ssm\webservice-ssm\examples\XACMLClient\README	XACML client is a simple client that does authorization to the Web Service SSM using the XACML protocol. It needs to first authenticate with the Web Service SSM and retrieve an identity assertion, then authorize with the identity assertion as subject.
WLS 9 SSM Examples	
BEA_HOME\ales22-ssm\wls9-ssm\examples\ALESEnabledWLP92Domain\README	This example shows how to protect a WebLogic Portal 9.2 application using WLS 9.x SSM.
WLS 8 SSM Examples	
BEA_HOME\ales22-ssm\wls-ssm\examples\ALESEnabledWLPDomain\README	This example shows how to protect a Portal application using WLS SSM.
BEA_HOME\ales22-ssm\wls-ssm\examples\ALESEnabledWLSCluster\README	ALES Enabled WLS Cluster Example.
BEA_HOME\ales22-ssm\wls-ssm\examples\ArmeMonitor	The sample ARME monitor is a Java Security Manager, which can run with Java SSM, WLS SSM or Web-Service SSM. This Security Manager creates a thread to monitor the ARME process status associated with this SSM.
BEA_HOME\ales22-ssm\wls-ssm\examples\ResourceConverter	Resource Converter is a Java-based plug-in that acts as a provider extension to extend the capabilities of the ASI Authorization provider. The example implements a custom Resource Converter.

Table 2-3 AquaLogic Enterprise Security Examples

Example Location	Description
BEA_HOME\ales22-ssm\wls-ssm\examples\SAMLServletExample	The SAML Servlet example demonstrates how an application can use the SAML servlet to use SAML Assertions to allow servers in different domains to operate in a federation of trusted servers based on successful Single Sign On (SSO) login to one of the servers by a user.
BEA_HOME\ales22-ssm\wls-ssm\examples\taglib	This example shows how to use ALES with taglibs. This example includes a set of sample taglibs and a simple web application that uses those tags in its index.jsp.

CHAPTER 3

Architecture Overview

This section describes the general architecture of the BEA AquaLogic Enterprise Security services, providers and service modules. Each Security Service Module comes with a set of security providers. Although applications can leverage the services offered through the existing security providers, the flexible infrastructure also allows security vendors, integrators, and customers to write their own security providers. The BEA AquaLogic Enterprise Security providers and third-party security providers can be mixed and matched to create unique security solutions, allowing organizations to take advantage of new technology advances in some areas while retaining proven methods in others. The Administration Server allows you to configure and manage all your security providers and service modules through one unified management console.

The architecture comprises the following major components, which are discussed in the following sections:

- "Administration Server" on page 3-1
- "Service Control Manager" on page 3-3
- "Security Service Modules" on page 3-4

Administration Server

BEA AquaLogic Enterprise Security lets you control all of your security policies and configuration data from a single console.

Note: This release of AquaLogic Enterprise Security supports any of the WebLogic 9.x security providers. However, the Security Service Module for WebLogic Server 9.x is configured differently, as described in Integrating ALES with Application Environments.

The Administration Server gives you the enterprise-wide visibility you need to analyze security policies and ensure that applications and resources are properly protected. BEA AquaLogic Enterprise Security also lets you delegate security administration to remote administrators who often better understand local users and business needs and who are better positioned to manage the security policies. By combining centralized control with delegated administration, you can define and manage overall policies while specifying the management responsibilities to be handled by organizational administrators. For additional information on the Administration Server features, see Chapter 5, "Security Administration." The Administration Server consists of several components (as shown in Figure 3-1), including:

Policy Distributor—Ensures that the correct policies are provided to each Security Service Module and maintains policy synchronization.

Policy Database—Maintains policy data managed by the Administration Server in a relational database. The database management system provides the authoritative source of configuration and policy. Data from the policy database is distributed to the Security Service Modules by the Policy Distributor.

Policy Exporter—Exports policy data from the policy database for later use with the same Administration Server or another instance of the Administration Server.

Policy Loader—Imports policy data from an external file, generated in another system, exported from another instance of an Administration Server, or manually coded.

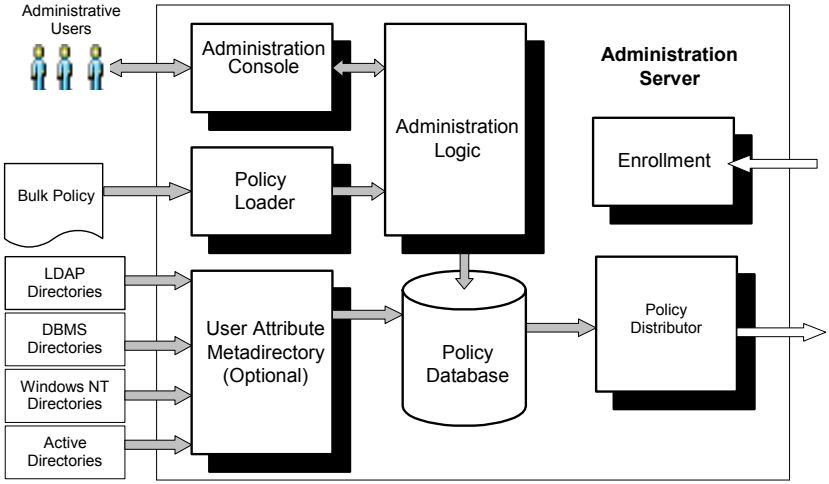
Administration Console—Supports administrative policy security and administration delegation through a web browser-based user interface. Security configuration, policy configuration, user attributes (if required), resources, and rules are all managed through the console.

Administration Logic—Maintains the Policy Database used by both the Administration Console and the Policy Loader.

Metadirectory—Application security policies frequently need more information than simple identity data to resolve sophisticated policy decisions. BEA AquaLogic Enterprise Security provides a metadirectory that accesses and synchronizes user attributes from a variety of sources to support these policies. These attributes are typically not be stored in any single place. The metadirectory assembles attributes for each user and caches them for use by Security Service Modules. For additional information on metadirectories, see "User Directories" on page 4-10. BEA AquaLogic Enterprise Security products can access user directories from a variety of

sources, including: application databases, Lightweight Directory Access Protocol (LDAP) directory servers, network databases, and others.

Figure 3-1 Administration Server Architecture



Service Control Manager

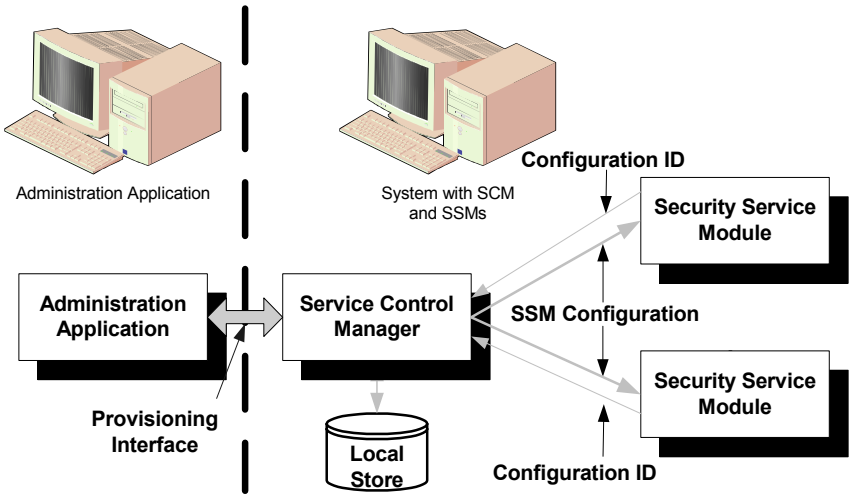
BEA AquaLogic Enterprise Security employs a fully-distributed security enforcement architecture consisting of Security Service Modules embedded in applications, application servers, and web servers throughout an enterprise (see Figure 3-2). To facilitate the management of a potentially large number of distributed Security Service Modules, the Administration Server uses a remote administration mechanism to distribute appropriate configuration and policy data to each Security Service Module. (The Security Service Module for WebLogic Server 9.x is configured differently, as described in Integrating ALES with Application Environments.)

The Service Control Module (SCM) is an essential component of this remote administration mechanism. Each Service Control Module is responsible for storing and maintaining the configuration data for all Security Service Modules running its machine. Once started, a Security Service Module receives its configuration data from the local Service Control Module. When a change is made and distributed from the Administration Server, the Service Control Manager receives the change and updates the cached copy of the configuration. On restart, the Security

Service Module receives updated configuration data from the Service Control Manager. Policy data does not require a restart, but is applied based on the desired provisioning characteristics.

In addition to facilitating management, the Service Control Manager enables Security Service Modules to operate in the absence of the Administration Server. Because the Service Control Manager maintains a persistent copy of each configuration, new Security Service Modules can be started and existing Security Service Modules continue to function, even if the Administration Server goes down or is intentionally unavailable, such as in occasionally connected computing environments.

Figure 3-2 Service Control Manager



Security Service Modules

BEA AquaLogic Enterprise Security supports a variety of Security Service Modules that you integrate with the security framework and provision as needed. The primary function of the security framework is to provide a simple application programming interface (API) that can be used by security and application developers to define security services. For a complete discussion of BEA AquaLogic Enterprise Security services, see Chapter 4, "Security Services." You may incorporate as many Security Service Modules as you need to secure the enterprise, and configure and manage them directly through a central Administration Server as shown in Figure 3-2. The distributed nature of the architecture allows you to configure, manage and distribute policy throughout the enterprise.

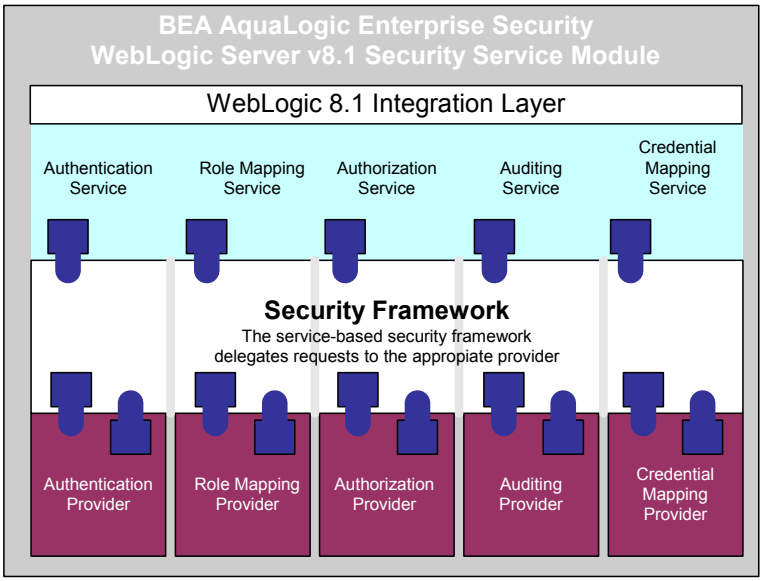
Configuration data for each Security Service Module is maintained within each machine and handled by a Service Control Manager. One additional benefit of this architecture is that even if the administration server goes down (either for maintenance or due to failure), there is no impact on the applications or security services provided by those Security Service Modules. At this time, the following Security Service Modules are available:

- "WebLogic Server 8.1 Security Service Module" on page 3-5
- "WebLogic Server 9.x Security Service Module" on page 3-6
- "Web Server Security Service Module" on page 3-7
- "Web Services Security Service Module" on page 3-8
- "Java Security Service Module" on page 3-9

WebLogic Server 8.1 Security Service Module

The WebLogic Server 8.1 Security Service Module is a security enhancement product that supports BEA WebLogic Server, Version 8.1. Further, the Security Service Module ties the application server into the Administration Server so that all application server administrative security activities are performed through the Administration Server. The application server with the Security Service Module add-on supports enterprise-level security by making security for WebLogic Server host applications an integral part of the enterprise policy. All WebLogic Server security-related functions remain available, but those functions are provided through the Security Service Module. Figure 3-3 shows the major components of the WebLogic Server 8.1 Security Service Module.

Figure 3-3 WebLogic Server 8.1 Security Service Module Architecture



WebLogic Server 9.x Security Service Module

The WebLogic Server 9.x Security Service Module is a security enhancement product that supports BEA WebLogic Server, Version 9.x.

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 WebLogic Server 8.1 SSM and the other AquaLogic Enterprise Security SSMs. When you install the WebLogic Server 9.x SSM, AquaLogic Enterprise Security uses the WebLogic Server 9.x security framework. As a consequence, when you use the WebLogic Server 9.x SSM, you configure security providers and other aspects of the SSM in the WebLogic Administration Console, rather than the AquaLogic Enterprise Security Administration Console.

You still use the AquaLogic Enterprise Security Administration Console to write security policies for all SSMs, and to configure SSMs other than the WLS 9.x SSM. You must also use the AquaLogic Enterprise Security Administration Console to configure the ASI Authorizer and ASI Role Mapper providers.

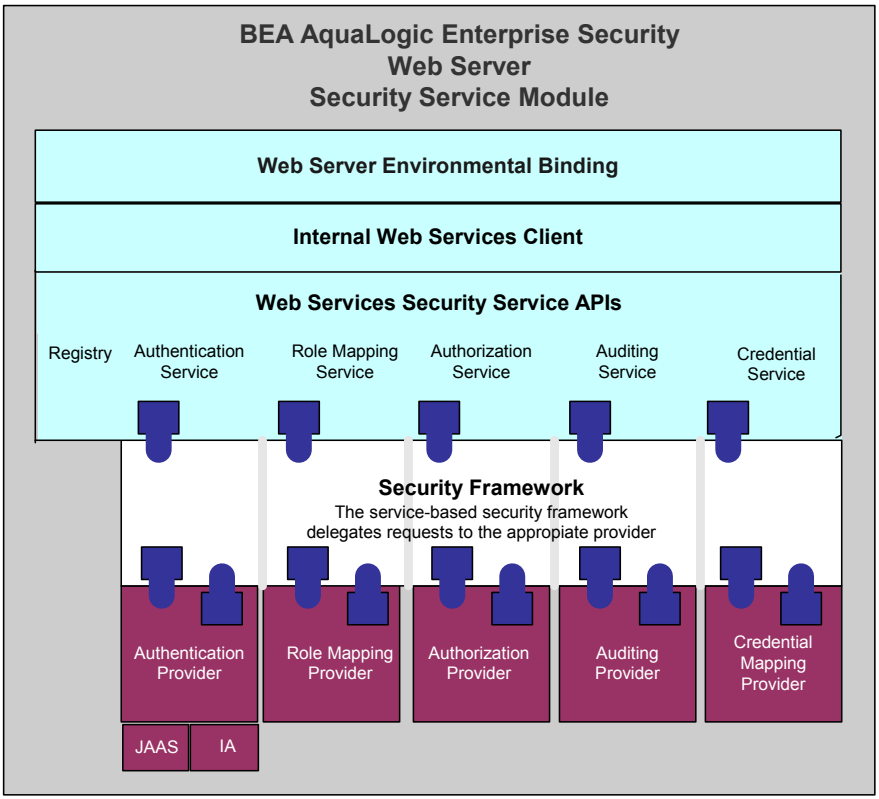
Web Server Security Service Module

The Web Server Security Service Module (SSM) provides an environmental binding between the AquaLogic Enterprise Security infrastructure and IIS and Apache web servers. The SSM consists of three components, a Web Server Environmental Binding, an Internal Web Services Client, and the Web Services SSM (which includes the Security Service APIs, Security Framework and security providers) (See Figure 3-4). The AquaLogic Enterprise Security infrastructure provides six distinct services: Registry, Authentication, Authorization, Auditing, Role Mapping, and Credential Mapping. Each of these services is expressed in a way that is understandable to applications running within a web server that is protected by the AquaLogic Enterprise Security infrastructure. Therefore, the SSM can be used to configured and enforce security for web server applications and resources.

The Web Server SSM makes access control 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.

You can tailor the Web Server SSM to meet your 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. Additionally, the Web Server SSM enables security administrators and web developers to perform security tasks for applications running on a web server.

Figure 3-4 Web Server SSM Components

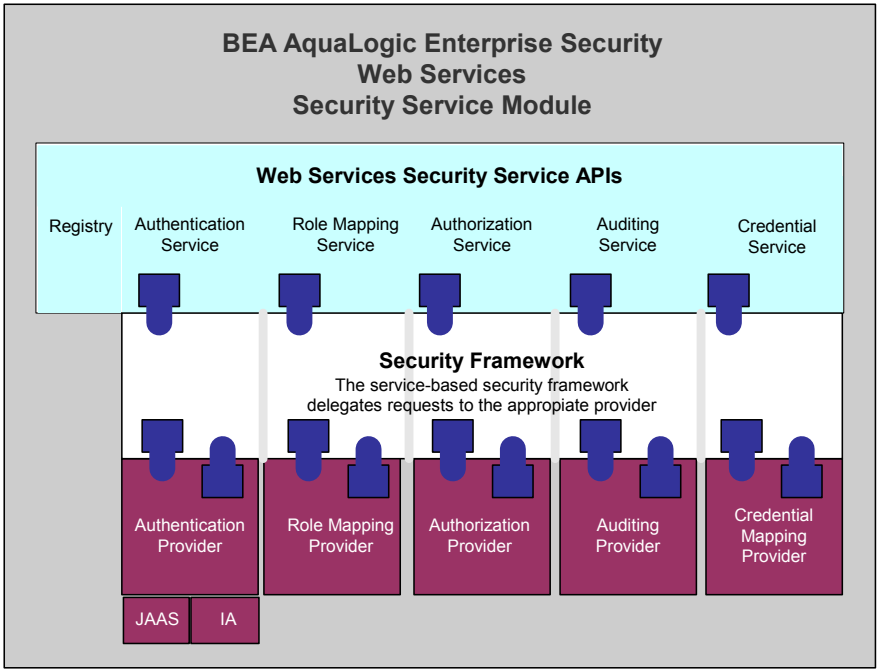


Web Services Security Service Module

The Web Services Security Service Module (SSM) provides five security services: Authentication, Authorization, Auditing, Role Mapping, and Credential Mapping (see Figure 3-5). The WSDL provided for these services can be used to developed web services clients to access the AquaLogic Enterprise Security infrastructure and use it to make access control decisions for custom applications.

Figure 3-5 shows the components of the Web Services SSM.

Figure 3-5 Web Services SSM Components

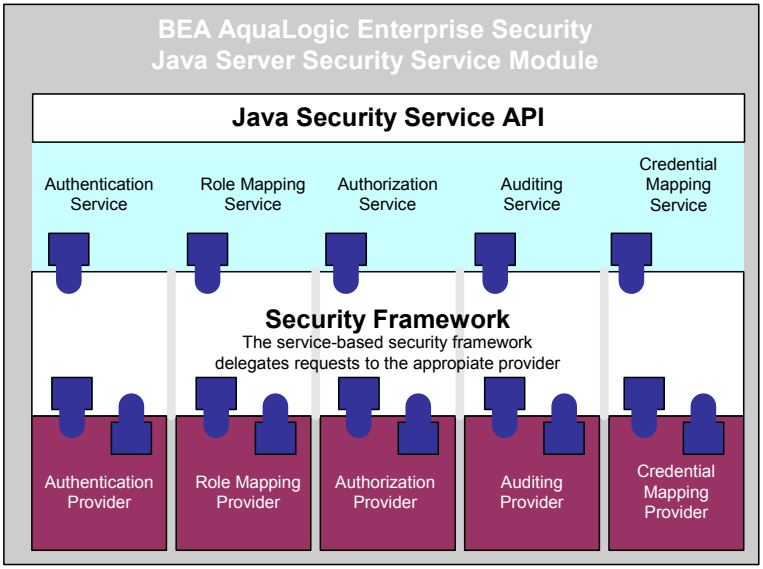


Java Security Service Module

The Java Security Service Module provides an application programming interface (API) that allows security developers to insert security into their applications. These interfaces support the most commonly required security functions and are organized into services that are logically grouped by functionality.

After you use the Java Security Service Module interfaces to implement security functions in your Java application, you can deploy and run your application on any instance of a Java Security Service Module that supports the configuration requirements of your application. The Java Security Service Module offers five security services: Authentication Service, Authorization Service, Auditing Service, Role Service, and Credential Mapping Service. The name of each service indicates a type of function that can be implemented within a Java application. Each of these services is discussed in Chapter 4, "Security Services." Figure 3-6 shows the major components of the Java Security Service Module. The Java Security Service Module comprises the security service APIs, the security framework, and the security providers that you configure.

Figure 3-6 Java Security Service Module Architecture



CHAPTER 4

Security Services

The following sections describe the fundamentals of the BEA AquaLogic Enterprise Security services:

- "Authentication and Identity" on page 4-1
- "Role Mapping" on page 4-7
- "Authorization" on page 4-8
- "Adjudication Service" on page 4-14
- "Auditing" on page 4-14
- "Credential Mapping" on page 4-15
- "Security Service Providers" on page 4-16

Authentication and Identity

Authentication answers the question, *Who are you?* using credentials such as username and password combinations. An Authentication provider is used to prove the identity of users or system processes. The Authentication provider also stores, transports, and makes identity information available to various components of a system when needed.

Authentication is the process of verifying an identity claimed by or for a user or system process. An authentication process consists of two steps:

1. Identification—presenting an identifier to the security system.
2. Verification—presenting or generating authentication information that corroborates the binding between the entity and the identifier.

Authentication Service

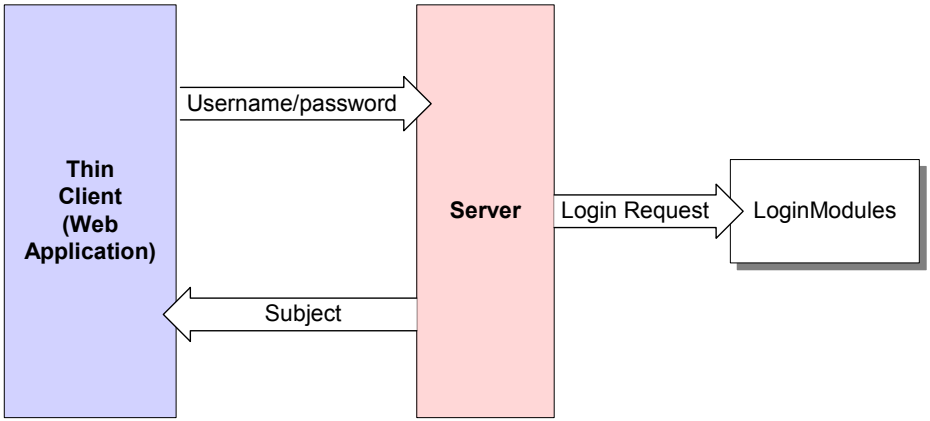
The Authentication providers support numerous methods of authentication services, including user username/password and identity assertion. In addition, a set of Authentication providers is supplied that access various types of user directories and identity tokens.

Figure 4-1 shows how the authentication process works for a web application that requires a username and password to perform the login. When a user attempts to log into a system using a username and password combination, the Authentication provider establishes trust by validating that username and password. This process also requires the use of a Login Module as described in "Identity Assertion" on page 4-5.

Note: Because the authentication service is implemented using the JAAS standard, developers of custom Authentication providers are involved with the JAAS process directly.

The authentication process requires a valid list of users and groups, through which their identity can be verified. BEA AquaLogic Enterprise Security supports any number of LDAP servers or a database.

Figure 4-1 Authentication Service Example for Username and Password



Types of Authentication

Users must be authenticated whenever a request is made to access a protected resource. For this reason, each user is required to provide a credential (for example, a password). The following types of authentication are supported by the Authentication providers:

- "Username and Password Authentication" on page 4-3

- "Perimeter Authentication" on page 4-3

You can use one of the Authentication providers provided as part of BEA AquaLogic Enterprise Security or you can create a custom authentication provider to perform a different type of authentication. Support for the following methods of user and password authentication include:

- Microsoft Windows NT
- OpenLDAP
- Novell
- Active Directory
- Database

Username and Password Authentication

In username and password authentication, a user ID and password are requested from the user and sent to the Security Framework. The Security Framework authenticates the user.

You can use Secure Sockets Layer (SSL) or Hyper-Text Transfer Protocol (HTTPS) to provide an additional level of security to username and password authentication. Because SSL encrypts the data transferred between the client and the Security Framework, the username and password are encrypted and not transferred in clear text. Therefore, BEA AquaLogic Enterprise Security services can authenticate the user without compromising the confidentiality of the username and password.

Perimeter Authentication

Perimeter authentication is the process of authenticating the identity of a remote user outside of the application server domain. You can use an Identity Assertion provider to establish **perimeter authentication**—a special type of authentication that uses tokens. The Security Framework supports identity assertion providers that perform perimeter-based authentication and handle multiple security token types and protocols.

How is Perimeter Authentication Accomplished?

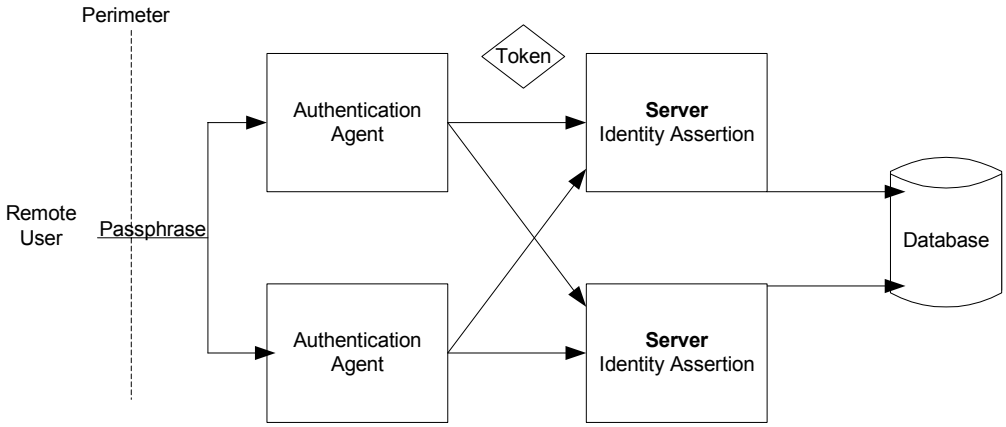
BEA AquaLogic Enterprise Security services are designed to extend the single sign-on concept all the way to the perimeter by using identity assertion (see Figure 4-2). Provided as a critical piece of the Security Framework, the concept of identity assertion allows you to use the authentication mechanism provided by perimeter authentication schemes such as the Checkpoint OPSEC, the emerging Security Assertion Markup Language (SAML), Simple and Protected

Negotiation Mechanism (SPNEGO), or enhancements to protocols such as Common Secure Interoperability (IIOP-CSIv2) to achieve this functionality.

Perimeter authentication is typically accomplished by the remote user specifying an asserted identity and some form of corresponding proof material, normally in the form of a passphrase, which is used to perform the verification. The **authentication agent**, the entity that actually vouches for the identity, can take many forms, including: Web Server, VPN, firewall, enterprise authentication service, or some other form of global identity service. Each of these forms of authentication has one common characteristic–they all perform an authentication process that results in an artifact or **token** presented to vouch for the authenticated user at a later time.

Currently, the format of the token varies from vendor to vendor, but there are efforts underway to define a standard token format. SAML is an emerging XML-based protocol for exchanging information securely. In addition, there is a current standard for identity attribute certificates that is based on the X.509 standard for digital certificates. But, if the applications and the infrastructure on which they are built are not designed to support this concept, enterprises are still forced to require remote users re-authenticate to the applications within the network. You can develop custom authentication providers that support different token types, including SAML.

Figure 4-2 Perimeter Authentication



Support for perimeter authentication requires the use of one or more identity assertion providers designed to support one or more token formats. You can implement multiple and different identity assertion providers. The tokens are transmitted as part of any normal business request, using the mechanism provided by each protocol supported. Once a request is received, the entity that handles the processing of the protocol message recognizes the existence of the token in the

message. This information is used in a call to the Security Framework, which then calls the appropriate identity assertion provider to handle the verification of the token. It is the responsibility of the identity assertion provider implementation to perform whatever actions are necessary to establish validity and trust in the token and to provide the identity of the user with a reasonable degree of assurance, without the need for the user to re-authenticate to the application.

The passphrase entered by the user is a string of characters (typically, much longer than a password) used to create a digitally-encrypted signature (or private or secret key). The passphrase adds an extra level of authentication security to ensure that you are who you say you are.

The following section describes additional authentication concepts and functionality.

- "Identity Assertion" on page 4-5

Identity Assertion

Identity Assertion providers support the mapping of a valid token to a user. You use an Identity Assertion provider to support the specific types of tokens that you use to assert the identities of users or system processes. You can develop an Identity Assertion provider to support multiple token types, but the administrator must configure the Identity Assertion provider so that it validates only one active token type. While you can have multiple Identity Assertion providers in a application domain with the ability to validate the same token type, only one Identity Assertion provider can actually perform this validation.

The security architecture supports Identity Assertion providers that perform perimeter-based authentication to Web Servers, firewalls, and VPNs. Identity Assertion providers support different token types, such as X.509 certificates (Kerberos), SAML and handles multiple security protocols (such as SOAP and IIOP-CSIv2). When used with an Authentication provider Login Module, Identity Assertion providers support single sign-on. For example, the Identity Assertion provider can generate a token from a digital certificate, and that token can be passed around the system so that users are not asked to sign on more than once. The Identity Assertion providers in BEA AquaLogic Enterprise Security support the following token types:

- `Authenticated User` type
- CSIv2 principal name identity
- CSIv2 anonymous identity
- CSIv2 X.509 certificate chain identity
- CSIv2 distinguished name identity

- X.509 client certificate
- SAML
- Single Pass Negotiate Identity Asserter (SPNEGO)

Identity Assertion Service

When perimeter authentication is used (see Figure 4-3), a token from outside of the application domain is passed to an Identity Assertion provider responsible for validating the type of token configured as active. If the token is successfully validated, the Identity Assertion provider maps the token to a username, sends that username back, and the authentication process continues. Specifically, the username is sent through a JAAS Callback Handler and is passed to each configured Authentication provider Login Module, so that the Login Module can populate the subject with the appropriate principals. Perimeter-based authentication requires the same components as the authentication process, but includes an Identity Assertion provider.

Identity Assertion providers are used as part of perimeter-based authentication process to validate the token type, and then map it to a username. It also specifies a list of trusted client principals to use for identity assertion. The wildcard character (*) can be used to verify that all principals are trusted. If a client is not listed as a trusted client principal, the identity assertion fails and the login is rejected. You can use an Identity Assertion provider in place of an Authentication provider if you create a Login Module for the Identity Assertion provider, or you can use an Identity Assertion provider in addition to an Authentication provider if you want to use the Authentication provider Login Module.

You can configure multiple Identity Assertion providers to support different types of tokens within the same application domain, but you do not need Identity Assertion provider unless you are supporting perimeter authentication. Two Identity Assertion providers are supplied: one supports X.509 and IIOP- CSIv2 and one supports SAML. A third Identity Assertion provider supports identity assertion using HTTP authentication tokens from the SPNEGO protocol, to enable single sign-on Cross-Platform authentication capabilities.

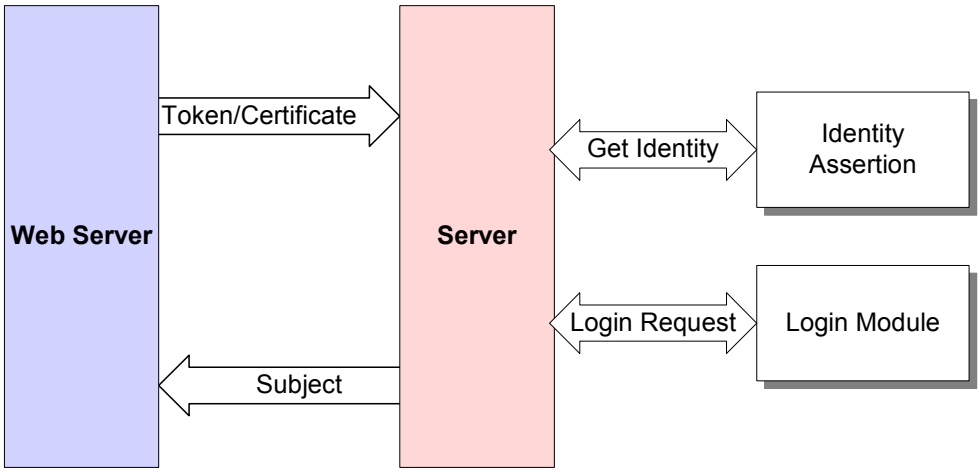
When used with a Login Module, Identity Assertion providers support external authentication of a user based on industry standard methods, for example, perimeter authentication or single sign-on. An Identity Assertion provider can generate a local subject. The Login Module that an Identity Assertion provider uses can be part of:

- A custom Authentication provider you develop
- The Authentication provider that BEA provides

- An Authentication provider developed by a third-party security vendor

Unlike a simple authentication situation, the Login Modules that Identity Assertion providers use *do not* verify proof material such as usernames and passwords; they simply verify that the user exists. The token provides the proof that the user is trusted.

Figure 4-3 Identity Assertion Example



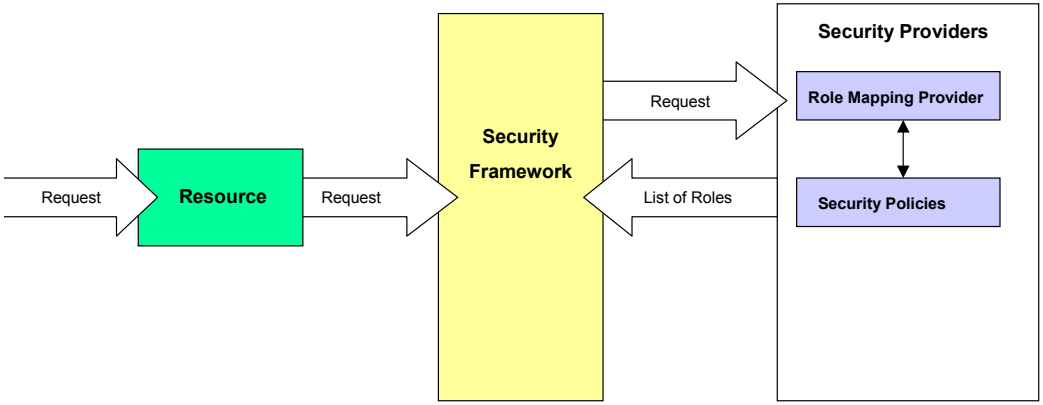
Role Mapping

Role mapping is the process by which the security service compares users or groups against a security role condition to determine whether a security role is granted. Role mapping occurs dynamically at runtime, just before an access decision is rendered for a protected resource.

Role Mapping Service

The Role Mapping provider determines dynamic roles for a specific user with respect to a specific protected resource. The Security Framework calls each Role Mapping provider that is configured as part of an authorization decision. It can also be used to determine the dynamic roles for the specific user to support personalization (for example, with BEA WebLogic Portal 8.1). Figure 4-4 shows how the Role Mapping provider interacts with the Security Framework to create dynamic role associations.

Figure 4-4 Role Mapping Service



The role mapping process is initiated when a user or system process requests a resource on which it attempts to perform an operation. The resource container that handles the type of resource requested receives the request (for example, the EJB container receives the request for an EJB resource). The resource container calls the Security Framework, and then passes in the request parameters, including information such as the subject (users and groups) of the request, the resource, and the action. The Security Framework calls each configured Role Mapping provider to obtain a list of the roles that apply. If an access control policy specifies that the requestor is entitled to a particular role, then the role is added to the list of roles that are applicable to the subject. This process continues until all security policies that apply to the resource or the resource container are evaluated. The list of roles is returned to the framework, where it can be used as part of other operations, such as access decisions.

The result of the dynamic role association performed by the Role Mapping provider is a set of roles that apply to the principals stored in a subject at the time of the request. These roles can then be used to make authorization decisions for protected resources, as well as a resource container and application code. For example, an EJB could use the Java 2 Enterprise Edition (J2EE) method to retrieve fields from a record in a database, without having knowledge of the business policies that determine whether access is allowed.

Authorization

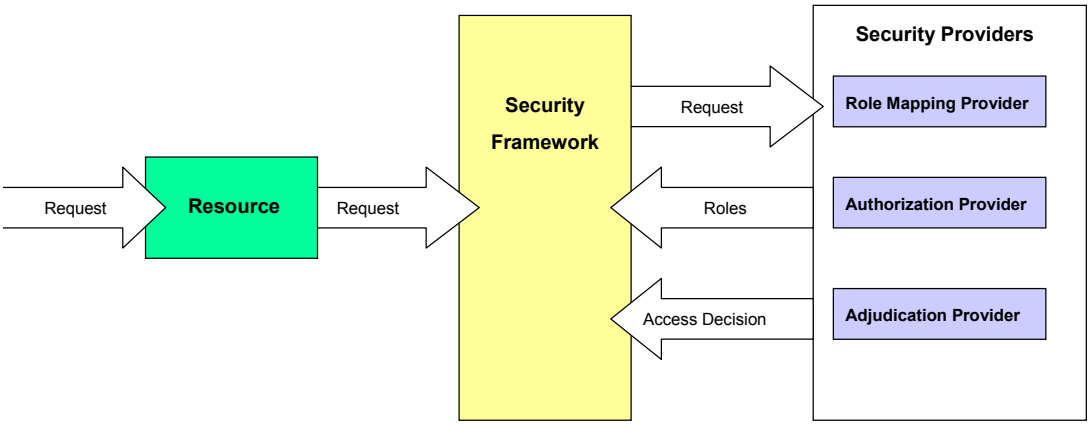
Authorization is the process whereby the interactions between users and resources are controlled, based on user identity or other information. In other words, authorization answers the

question, *Can this user access this resource under these conditions*? An Authorization provider is used to limit the interactions between users and resources to ensure integrity, confidentiality, and availability. You can optionally define a time constraint for an authorization policy to answer the question, *When can the user access the resource?*

Authorization Service

The Authorization service determines whether or not a user can access a resource based on the user's name, group, or role and the authorization policy assigned to the requested resource. Figure 4-5 illustrates how an Authorization provider, and the associated Adjudication and Role Mapping providers, interact with the Security Framework during the authorization process.

Figure 4-5 Authorization Service Example



The authorization process is initiated when a user request is made to perform an operation on a resource. The resource receives the request, calls the Security Framework, and then passes the request parameters to the security providers, including information such as the subject (users and/or groups) of the request, the resource requested, and the action. The Security Framework calls the configured Role Mapping provider using the request parameters to dynamically compute a list of roles to which the user making the request is entitled, and then passes the list of applicable roles back to the Security Framework.

The Authorization provider determines whether the user is entitled to perform the requested action on the resource, based on the user's profile and the supplied resource and the user's role, and makes an access control decision.

The following sections describe authorization concepts and functionality:

- User Directories
- Resources
- Authorization Policies and Role Mapping Policies
- ContextHandlers

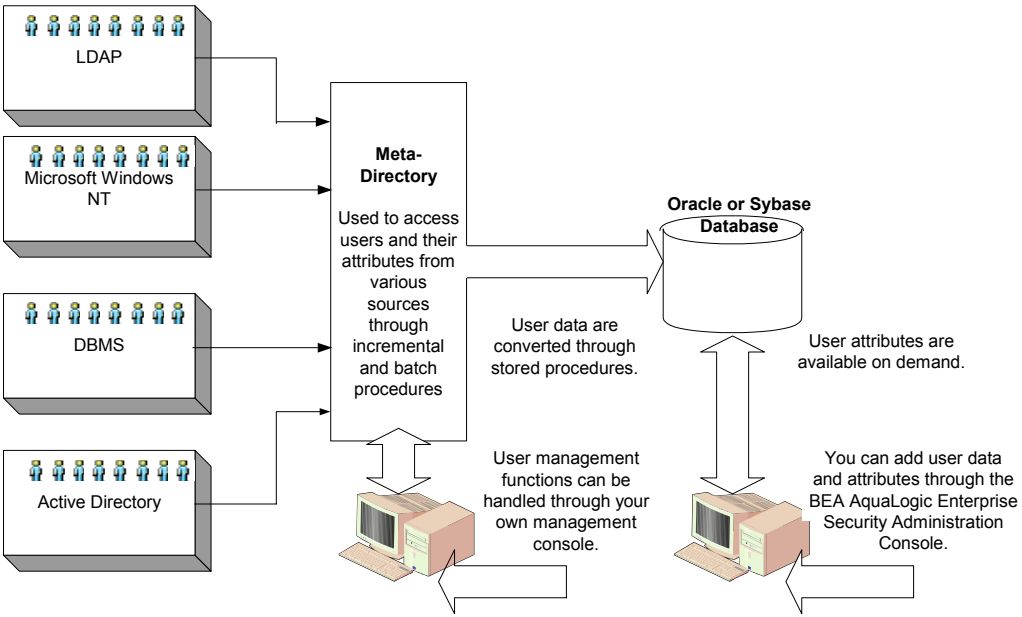
User Directories

BEA AquaLogic Enterprise Security products can access user directories from a variety of sources, including: application databases, Lightweight Directory Access Protocol (LDAP) directory servers, network databases, and others. The type of information or attributes collected—a method typically referred to as profiling—also varies and typically includes: name and address, phone, e-mail address, personal preferences, and more. The information contained in these directories is the core of any authentication service and is key to providing an effective identity management solution.

Users and group information, along with their attributes, are cached in an Oracle or Sybase database table. User data is then available for access to create or enforce authorization policies.

BEA AquaLogic Enterprise Security has no control over your user community; your method of storage is completely under your control. For administrative purposes, however, all users must be a member of at least one group.

Figure 4-6 User Attribute Aggregation through Metadirectories



Resources

A **resource** is a structured object used to represent an underlying entity that can be protected from unauthorized access using security roles and policies. Resources are hierarchical by nature. Therefore, the level at which you define these roles and policies is up to you. For example, you can define roles and policies on entire EAR, an EJB JAR containing multiple EJBs, a particular EJB within that JAR, or a single method within that EJB. Some typical resources you might want to protect include.

- An application, an application window, or a dialog box
- Specific business transactions, such as a money transfer or security trade
- Application controls, such as buttons and menu selections
- Database or directory server structures
- Web pages, servlets, and EJBs

- Products or services in a portal

Note: This release of AquaLogic Enterprise Security includes an example that demonstrates the principles of an EJB application functioning under the ALES framework. The example shows how the additional layer provided by ALES can make an EJB application more secure by defining custom dynamic authorization policies without any changes in the application or the deployment descriptors. The example consists of an EJB application that runs on the server side, an EJB client, and the sets of resources, policies, and users that determine the access rights.

The example is installed in `BEA_HOME\ales22-admin\examples\EJBAppExample`.

Authorization Policies and Role Mapping Policies

A resource has no protection until it is protected by a policy. In early security systems, an access control list (ACL) was used to protect resources. BEA AquaLogic Enterprise Security authorization policies replace ACLs through the use of role-based access control (RBAC). Like Login Modules for Authentication providers, an access decision is the component of an Authorization provider that actually answers the question, *Is access allowed?* Specifically, an access decision is asked whether a subject has permission to perform a given operation on a resource, with specific parameters in an application. Given this information, the access decision responds with a result of `PERMIT`, `DENY`, or `ABSTAIN`.

You create an authorization policy by establishing the permissions, or privileges, for access, based on the resource you want to protect and the role of the requestor; authorization policy focuses on the business process or role, rather than on a user or group. Using RBAC you separate the process of identifying the requestor (defined through a separate authentication process), from the process of authorization (defined by an access decision). The performance improvement gained by this separation is really quite significant. Once the user is authenticated, the access decision can be made more quickly because there is no need to re-authenticate. The importance and benefit of basing access decisions on roles becomes quite apparent as you can assign large groups of users, who's business functions are the same, to the same role. Because role changes less frequently than user or group status, this mechanism is easily scalable.

You assign authorization policies to any of your resources (for example, a URL, an EJB resource or a JNDI resource) or to attributes or operations of a particular instance of a resource (for example, an EJB method or a servlet within a web application). If you assign a authorization policy to a type of resource, all new instances of that resource inherit that policy.

To use a user or group to create an authorization policy, the user or group must be defined for the authorization provider that is configured for the service module to which the provider is associated. For efficiency, this is typically done through the use of roles rather than user or

groups. To use a role to create a role mapping policy, the role must be defined for the Role Mapping provider that is configured for that service module. All authorization and role mapping policy data is stored in the policy database. When the Authorization provider and Role Mapping provider are configured, their configurations are also stored in the policy database.

Note: Although you have the option of basing an authorization policy on a user or group, BEA recommends basing authorization policies on roles. Basing authorization policies on roles is a more efficient method of management.

ContextHandlers

A ContextHandler is a high-performing class that obtains additional context and container-specific information from the resource container, and provides that information to security providers making access or role mapping decisions. The ContextHandler interface provides a way for an internal resource container to pass additional information to a Security Framework call, so that a security provider can obtain contextual information beyond what is provided by the arguments to a particular method. A ContextHandler is essentially a name/value list and, as such, it requires that a security provider know what names to look for. In other words, use of a ContextHandler requires close cooperation between the resource and the security provider. Each name/value pair in a ContextHandler is known as a context element and is represented by a Context Element object.

For example, with WebLogic Server 8.1, three types of resource containers pass Context Handlers to the Security Framework: the Servlet, EJB, and Web Service containers. Thus, URL (web), EJB, and Web Service resource types have different context elements whose values Authorization and Role Mapping providers can inspect. An implementation of the Audit Context interface (used when a security provider is implemented to post audit events) may also examine the values of context elements.

Adjudication

Adjudication involves resolving any authorization conflicts that may occur when more than one Authorization provider is configured in a application domain, by weighing the result of each Authorization provider's access decision. An Adjudication provider tallies the results that multiple access decisions return, and determines the final `PERMIT` or `DENY` decision. An Adjudication provider may also specify what to do when an answer of `ABSTAIN` is returned from an access decision from a single Authorization provider.

Adjudication Service

If there are multiple Authorization providers configured, the Security Framework delegates the job of reconciling any conflicts in the access decisions rendered by the Authorization providers to the Adjudication provider, which determines the ultimate outcome of the access decision. The Adjudication provider returns either a `TRUE` or `FALSE` verdict to the Authorization providers, and then forwards it to the resource through the Security Framework.

Auditing

In the BEA AquaLogic Enterprise Security architecture, an Auditing provider is used to support auditing services, for example, using the Log4j Auditing Channel provider. An auditing service is used to collect all security event information and distribute that information as configured.

If configured, the Security Framework calls the Auditing provider before and after security operations are performed (such as authentication or authorization), enabling audit event recording. The decision to audit a particular event is made by the Auditing provider itself and can be based on specific audit criteria and severity levels. The records that contain the audit information may be written to output repositories such as an LDAP server, a database, or a file.

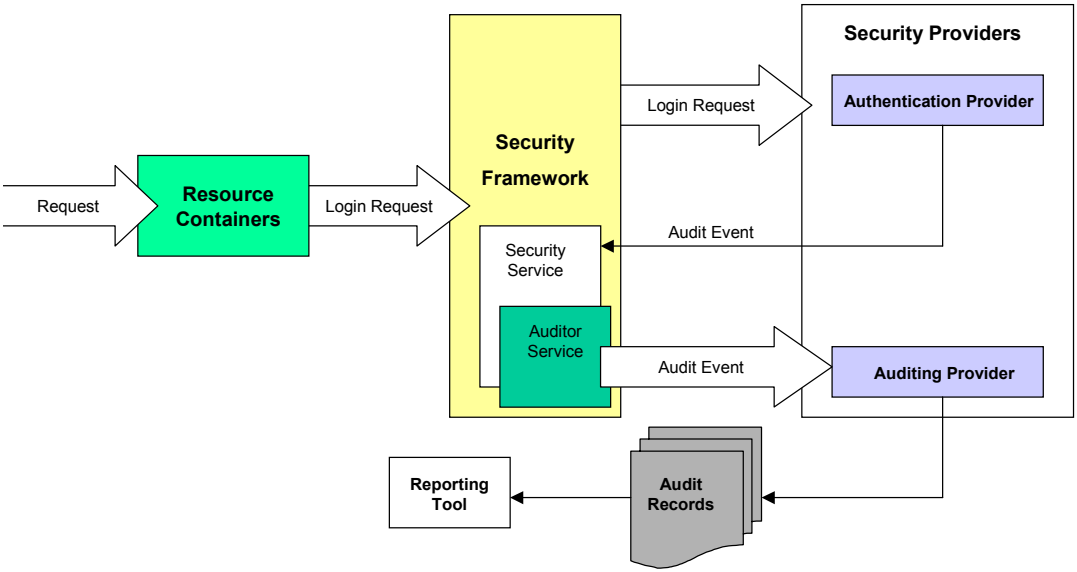
Auditing Service

An auditing service is a process whereby information about service requests and the outcome of those requests are made available to configured auditing providers. Auditing provides an electronic trail of transaction activity, and can include changes to system configuration parameters, policy changes, transactions, and security breach attempts of any type. For each audit item, the information may include who, what, when, where, and sometimes why.

Records can be directed to a file, a database or other persistent storage medium, and then used to generate reports from any reporting generating software you choose. Figure 4-7 shows how the Auditing provider interacts with the Security Framework and other types of security providers using a Authentication provider as an example.

The auditing process is initiated when a resource container passes authentication information for a user (for example, a username and password combination) to the Security Framework as part of a login request. The Security Framework passes the information associated with the login request to the configured Authentication provider. The Authentication provider passes the audit event back through the Security Framework, which then makes that audit event available to the configured Auditing provider.

Figure 4-7 Auditing Service Example for Authentication



Performance Statistics

The performance statistics feature enables the collection of data about authentication and authorization for purposes of troubleshooting and performance analysis. The performance statistic feature is controlled by an Auditing security provider, the PerfDBAuditor provider. Performance statistics are gathered for each Security Service Module in your AquaLogic Enterprise Security installation. In order to collect performance statistics for an SSM, you must enable and configure a PerfDBAuditor provider for that SSM.

See *Administration and Deployment Guide* for additional information.

Credential Mapping

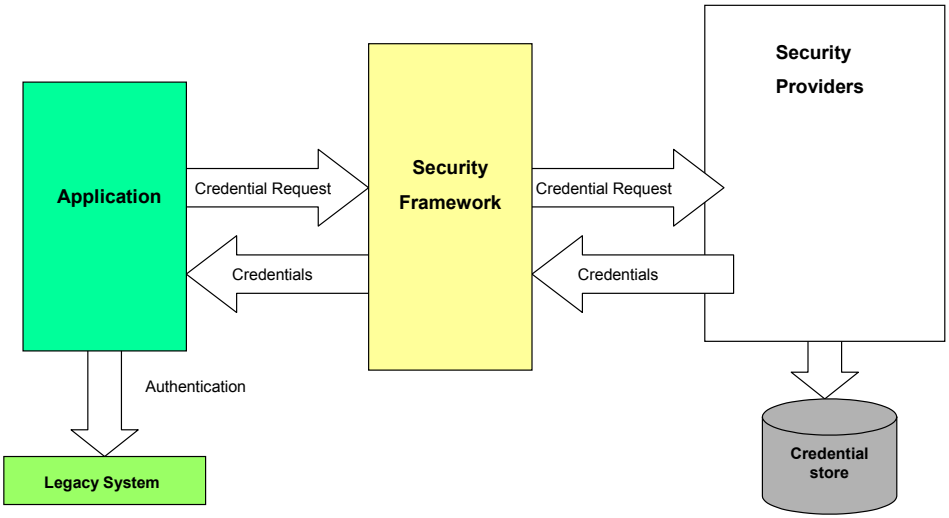
Credential mapping is the process of looking up the authentication credentials associated with a user for a given resource.

Credential Mapping Service

The Credential Mapping service provides authentication credentials for a user of another application, for example, some type of mainframe transaction processing, a database system, or a legacy application. Three Credential Mapping providers are supplied: one supports usernames and passwords stored in a relational database; one supports usernames and passwords stored in a WebLogic Server 8.1 embedded LDAP directory (this provides support for WebLogic Portal 8.1); and one produces SAML credentials.

Figure 4-8 illustrates how the Credential Mapping provider interacts with the Security Framework to support the credential mapping service.

Figure 4-8 Credential Mapping Service



Security Service Providers

Security services are based on a set of Security Service Provider Interfaces (SSPIs). Developers and third-party vendors can use the SSPIs to develop custom security providers for BEA AquaLogic Enterprise Security. SSPIs are available for Authorization, Role Mapping and Adjudication; Auditing; Authentication, and Identity Assertion; and Credential Mapping (for a description of providers, see Table 4-1). The SSPIs allow customers to use their security providers to secure resources. Customers can use the SSPIs to develop custom security providers or they can purchase customer security providers from third-party vendors.

Table 4-1 Security Providers

Provider Type	Provider Name	Description
AquaLogic Enterprise Security Providers		
Authentication	Novell Authenticator	Authenticates users using a Novell LDAP directory.
	Database Authenticator	Authenticates users using the ALES relational database provider.
	Open LDAPAuthenticatorr	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.
Identity Assertion	X.509 Identity Assertion Provider	Supports identity assertion through an X.509 digital certificate (ASN.1 encoding and decoding).
	SAML Identity Assertion Provider	Supports identity assertion through Security Assertion Markup Language v1.1.
	Single Pass Negotiate Identity Asserter	Supports identity assertion using HTTP authentication tokens from the SPNEGO protocol. The provider supports the identity assertion using the Kerberos tokens contained within the SPNEGO token.
	ALES Identity Assertion Provider	Supports identity assertion through an ALES cookie.

Table 4-1 Security Providers (Continued)

Provider Type	Provider Name	Description
Credential Mapping	Database Credential Mapping Provider	Provides authentication credentials for a user (username and password) from a database for use in another application.
	SAML Credential Mapping Provider	Provides authentication credentials for a user (a SAML assertion) for use in another application.
	ALES Credential Mapping Provider	Provides authentication credentials for a user (a ALES cookie) for use in another application.
Auditing	Log4j Audit Channel Provider	Provides auditing features that support the Apache Log4j capabilities.
Authorization	Authorization Provider	Provides enforcement of authorization for the policy with which it is used. The Authorization Provider returns an access decision that determines which resources are protected and if a particular user is allowed access to a resource.
	Adjudication Provider	Provides an authorization decision based on results from one or more authorization providers.
Role Mapping	Role Mapping Provider	Provides assignment to roles based on the access control policy with which it is used. The Role Mapping Provider returns a set or roles granted to a user on a protected resource.

Table 4-1 Security Providers (Continued)

Provider Type	Provider Name	Description
WebLogic Server 8.1 Providers	WebLogic Authentication Provider	Supports authentication through an embedded LDAP Directory for WebLogic Server.
	WebLogic Authorization Provider	Provides enforcement of authorization for the access control policy in WebLogic Server. The Authorization Provider returns an access decision that determines which resources are protected and if a particular user is allowed access to a resource.
	WebLogic Role Mapping Provider	Provides assignment to roles based on the access control policy defined for use with WebLogic Server 8.1. The WebLogic Role Provider returns a set or roles granted to a user on a protected resource.
	WebLogic Credential Mapping Provider	Provides authentication credentials for a user (username and password) for single sign-on use into another application or resource.

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

- Configuring WebLogic Security Providers
- Configuring Authentication Providers

In addition, you can use the following AquaLogic Enterprise Security 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: To assist customers in developing custom security providers, sample custom security providers are also available from the BEA dev2dev web site at http://dev2dev.bea.com. For more information on developing custom security providers, see *Developing Security Providers for BEA AquaLogic Enterprise Security*.

CHAPTER 5

Security Administration

This section covers the following topics:

- "Managing Security" on page 5-1
- "Security Configuration" on page 5-3
- "Resources" on page 5-4
- "Identity" on page 5-8
- "Policy" on page 5-12
- "Declarations" on page 5-14
- "Deployment" on page 5-14

Managing Security

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

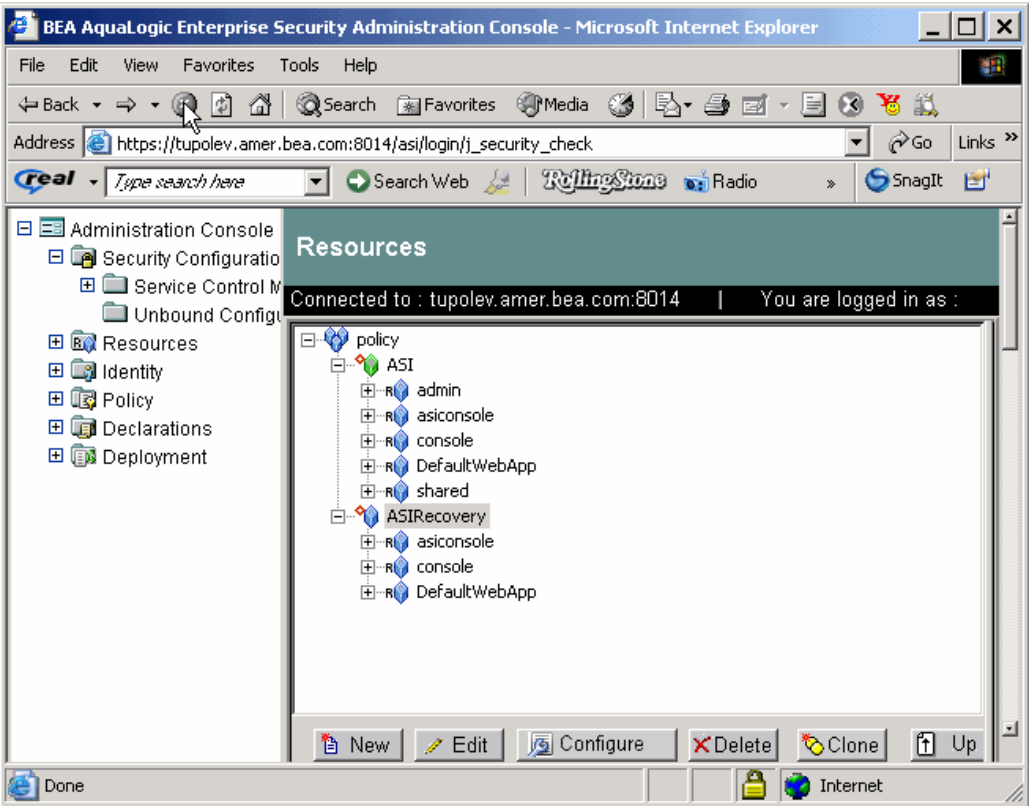
The policies that define your access control policy are built from policy elements: identity (users and groups with their associated attributes), privileges, roles, and the resources you want to protect. A policy can be applied to either a single resource or to many resources across the enterprise. An enterprise-wide policy represents the superset of all of your authorization and role

mapping policies. The following sections describe security management concepts and how you implement security using the Administration Console:

- "Security Configuration" on page 5-3
- "Resources" on page 5-4
- "Identity" on page 5-8
- "Policy" on page 5-12
- "Deployment" on page 5-14

Figure 5-1 shows how the Administration Console represents the various policy elements and resources. The resources shown here represent the initial policy settings used to protect the console.

Figure 5-1 Administration Console Resource Representation



Security Configuration

A Security Configuration is represented by the icon in the Administration Console, and consists of one or more Service Control Managers, one or more Security Service Modules represented by the icon, and the providers associated with each module. You can use the security providers that are provided, purchase custom security providers from third-party security vendors, or develop your own custom security providers. Table 5-1 shows how each provider type is represented in the Administration Console.

Table 5-1 Security Providers and their Representations

	Adjudication		Authorization
	Auditing		Credential Mapping
	Authentication		Role Mapping

To learn more about security providers and their services, see "Architecture Overview" on page 3-1. For information on how to develop custom security providers, see *Developing Security Providers for AquaLogic Enterprise Security*.

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

A configuration also defines the method of authentication and authorization used to provide access to a resource for one or more users and groups. The authorization policy protects the resource against unauthorized access. Each Security Service Module has a Configuration ID associated with it that defines the security providers, policy, and settings for that Security Service Module.

Resources

A resource is a general term that refers to any object that you can protect. Resources can include applications, data, or system components. Resources may also include background services with which the user has no direct interaction. For example, a bank might offer banking services on a web page that simulate the actions of a teller. The components (buttons, tables, data fields) displayed on that web page are all resources. The Administration Console displays resources in a tree structure called a resource hierarchy as shown in Figure 5-2. The resources shown here represent the administration services and policy data elements that are accessible to the Administration Server on startup.

A page generated from a JSP is also a resource. The page can call EJBs or COM resources to execute a transaction. The back office services that transfer money between accounts, issue a

payment, or run a report are also resources, although they may not appear on the web page or execute on the application server.

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

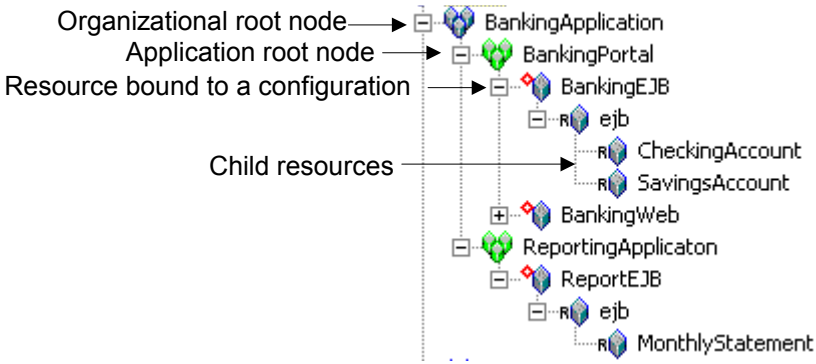
Another type of organizational node is an application node represented by a green resource icon . This icon depicts a collection of resources that provide a specific set of services. Applications are considered organizational because they often consist of multiple, distributed components, such as an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

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

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

Any resource at or above a binding node can be marked as a policy distribution point. When a policy distribution is initiated, you can choose to distribute either all updates (by selecting the root node) or you can limit which updates are distributed by selecting resources from the existing distribution points. Only updates that were made at or below the selected distribution points are distributed. Figure 5-2 shows how organizations, applications and resources are represented and bound to Security Service Modules in the Administration Console.

Figure 5-2 Organizations, Applications, and Resources form your Security Configuration



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

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

The following topics provide more information:

- "Resource Attribute" on page 5-6
- "Privilege" on page 5-7
- "Privilege Group" on page 5-7

Resource Attribute

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

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

Privilege

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

You can use the privileges provided or you can create your own. Figure 5-3 shows how privileges appear in the Administration Console. Notice that each privilege refers to an action. A related collection of privileges may be organized into a privilege group for management purposes.

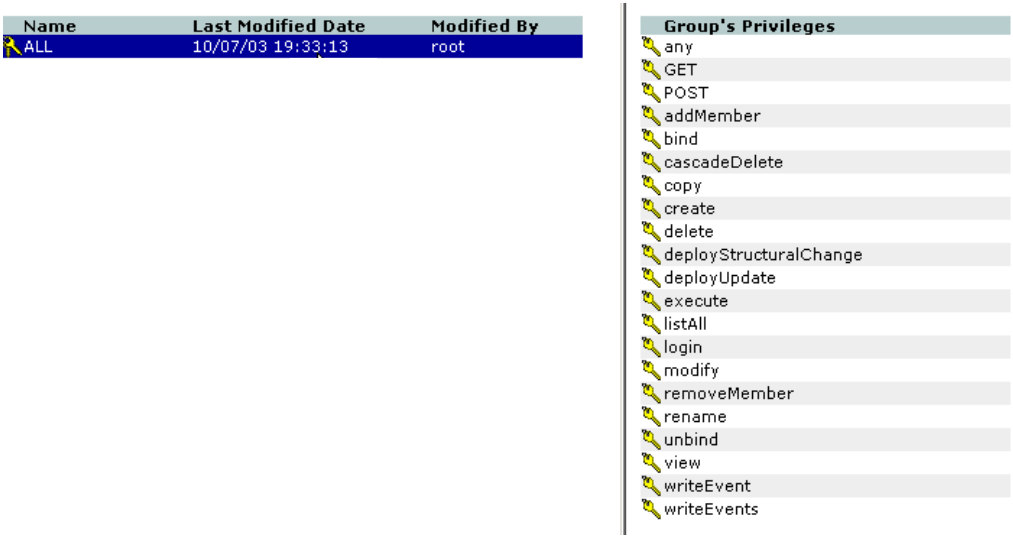
Figure 5-3 Privilege Representation in the Administration Console

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

Privilege Group

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

Figure 5-4 Privilege Group Representation in the Administration Console



Identity

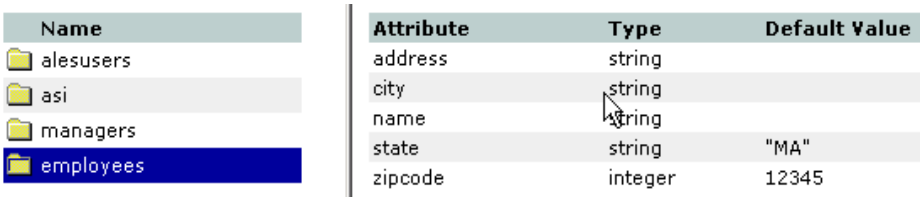
The Identity icon represents your directories and user communities in the Administration Console. Although BEA AquaLogic Enterprise Security provides tools to manage users and groups locally, they are typically managed through an external repository, such as a Lightweight Directory Access Protocol (LDAP)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

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

A directory typically represents groups of users of a particular application or resource, or users in a specific location. Each directory has an associated attribute schema. The schema defines the attributes applied to members of the directory. Figure 5-5 shows how directories are represented in the Administration Console. In this example, there are two directories: ales and Employees.

The `employees` directory shows the attributes that are stored for each member of the directory: zipcode, state, name, city and address. The zipcode and state attributes have default values set.

Figure 5-5 Directory Representation in the Administration Console



The number of directories you have depends on the level of granularity needed to separate your user community. You may want to have one global directory containing all users. In this case, you can populate a single directory using multiple external repositories. Or, you might want separate directories–one directory for customers, one for employees, and one for partners, where the method of authentication is different for each group. Having one directory requires a unique name for each user and group. If you are not able to guarantee this when you integrate your repositories, you should maintain separate directories.

The following topics provide more information about Identity:

- "Users" on page 5-9
- "Groups" on page 5-10
- "Identity Attributes" on page 5-11
- "Roles" on page 5-12

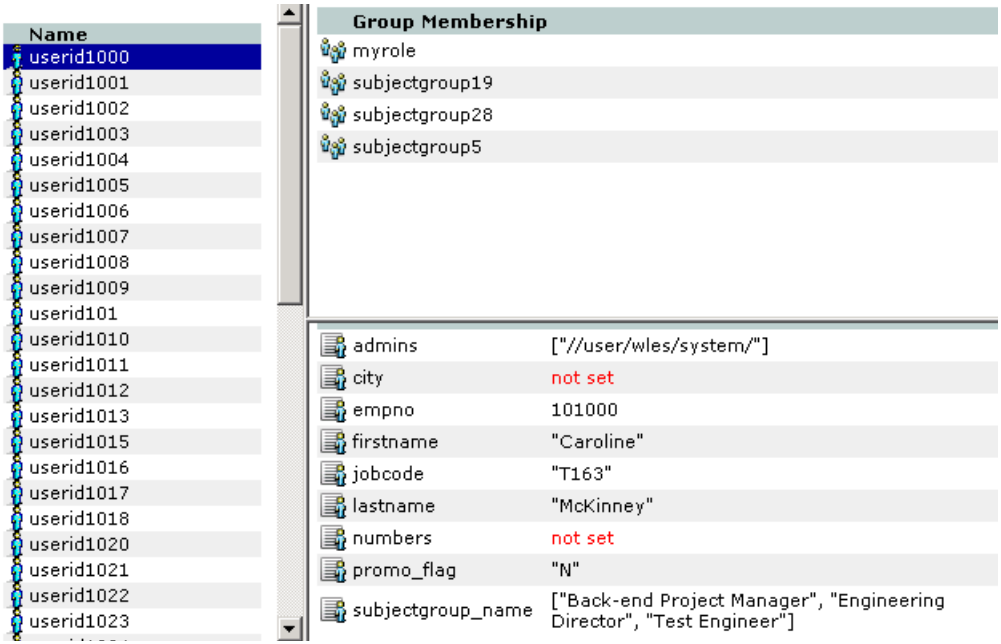
Users

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

Users can be associated with certain characteristics, referred to as identity attributes; these attributes store information about the user. The list of attributes that can be set for a user is

dictated by the attribute schema of the directory to which the user belongs. Figure 5-6 shows an example of a user representation with identity attributes. In this example, `userid1000` belongs to four groups and there are nine attributes associated with the user.

Figure 5-6 User Representation in the Administration Console



Groups

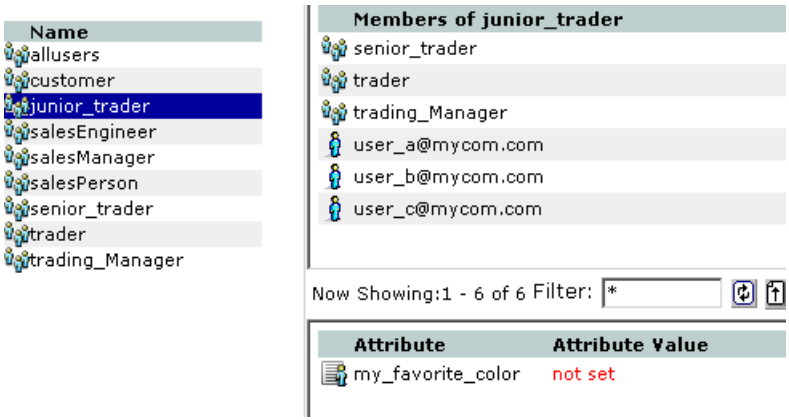
A group is represented by the icon in the Administration Console and is a logical collection of users that share some common characteristics, such as department, job function, or job title. For example, a company may separate its sales staff into two groups, Sales Representatives and Sales Managers,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

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

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

Figure 5-7 shows how groups are represented in the Administration Console. In this illustration, the junior_trader group contains three group members, three user members, and one attribute (my_favorite_color).

Figure 5-7 Group Representation in the Administration Console



Identity Attributes

Identity attributes are represented by the icon in the Administration Console (as shown in Figure 5-7). Each user and group can have different characteristics defined as identity attributes. The type of information or attributes collected—a method typically referred to as profiling—also varies and typically includes information such as name and address, phone, e-mail address, personal preferences, and so forth. Identity attributes can be extracted from the external data source.

An identity attribute is declared specifically to contain identity information. An attribute value can be used in policies to set limits for that user. Attributes provide a very powerful way to refer

to users and groups indirectly in policies, which results in a more dynamic and versatile access control policies.

Roles

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

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

Policy

Authorization policy controls what actions a user can perform on a resource. Authorization policies are typically written to grant specific privileges upon resources to users within a given role under a defined set of conditions.

The following topics provide more information:

- “Role Mapping Policies” on page 5-12
- “Authorization Policies” on page 5-13
- “Role Mapping Policy Reports” on page 5-13
- “Authorization Policy Reports” on page 5-13

Role Mapping Policies

A role mapping policy is represented by the icon in the Administration Console and displays the policies that determine the membership for each role you have created, to which a user or group can be assigned. BEA recommends defining roles, and then assigning users and groups to those roles. A delegation policy assigns a role or privilege on a resource from one user or group (delegator) to another user or group (delegate). To delegate, the delegator must first have the role

assigned to them. Role mapping policies are used to assign users or groups to membership in roles. The membership can be limited based on a number of items including the resource hierarchy, identity, contextual, and resource attributes. Roles may also be delegated from one user to another using delegation policies.

Authorization Policies

Authorization policies are represented by the icon in the Administration Console and displays the policies that determine what actions can be performed on a resource. Authorization policies are typically written to grant specific privileges upon specific resources to a role with a defined set of constraints. An authorization policy can define privileges, resources, policy subjects (users, groups, and roles), constraints, and delegators. For ease of policy management, BEA recommends that authorization policies be written to grant privileges to roles, rather then granting them directly to users or groups. A delegation policy that delegates a privilege assigns the privileges granted to one user (the delegator) on a resource to another user, group, or role.

Role Mapping Policy Reports

Role mapping policy reports are represented by the icon in the Administration Console Using this function you can create a role mapping policy inquiry and use it generate a report that you can use for analysis. Role mapping policy inquiries search for role-based, access-control policies written against specific resources that match specified characteristics exactly. You can define inquiries that include a policy subject list (user and group), a role list, a resource list, and a delegator list. Role mapping policy inquiries ask the question, What role can do what to what resource?

Authorization Policy Reports

Authorization policy reports are represented by the icon in the Administration Console. Using this function you can create an authorization policy inquiry and use it generate a report that you can use for analysis. Authorization policy inquiries search for privilege-based policies that match specified characteristics exactly. You can define inquiries that include a policy subject list (user, group and role), a privilege list, a resource list, and a delegator list. Authorization policy inquiries ask the question, Who can do what to what resource?

Declarations

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

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

Deployment

Deployment is represented by the icon in the Administration Console.

Once you have designed your access control policies, you must deploy it before it can take effect in your environment. Before you deploy policy, choose the distribution point for the policy. The distribution point identifies what portions of the policy updates are made active in your environment. After the distribution, you can view the results of the policy distribution. You also distribute security configuration data. After a configuration update, you must restart the effected modules to make use of the new configuration; this means you must restart the Security Service Module component and your application.

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

Index

A

access control 2-7
access decision 2-13, 4-12, 4-13
adjudication
    defined 4-13
Adjudication provider 4-18
    purpose 4-13
administration
    delegated 3-2
Administration Console 3-2, 5-2
Administration Server 3-2
administration, delegated 2-15
application node 5-5
architecture 2-13, 3-1
    administration server 3-3
attribute
    defined 5-14
    identity 5-11
    resource 5-6
auditing 4-18
Auditing provider 2-13, 4-14
auditing service 4-14
authentication 4-17
    defined 4-1
    perimeter-based 2-7
    username and password 4-3
Authentication provider 2-12
    purpose 4-1
authorization 4-18
Authorization provider 2-13, 4-18
    purpose 4-9

B

binding node 5-5

C

Callback Handler 4-6
configuration 5-4
configuration data 3-5
constant
    defined 5-14
credential 4-2
credential mapping 4-18
Credential Mapping provider 2-12
custom authentication provider 4-3
custom security providers 2-14

D

Database Credential Mapping provider 4-18
delegate policy 5-12
digital certificates
    server authentication 2-7
directories
    user 4-10
directory server 5-8
distributed architecture 2-10
dynamic role
    association
        result of 4-8
dynamic role association
    using Role Mapping providers 2-13

E

enumerated type 5-14
evaluation function
    understanding 5-14

F

features 2-3

G

group
    membership 5-12

H

HTTPS
    protocol 4-3
Hyper-Text Transfer Protocol 2-7, 4-3

I

identity
    unique 5-9
identity assertion 4-17
Identity Assertion provider 4-5, 4-6
    support for single sign-on 4-5
identity attribute 5-12
    defined 5-11

J

Java 2 Enterprise Edition (J2EE) 4-8
Javadocs 2-14
JDBC 2-16

L

LDAP 2-16, 4-10
Lightweight Directory Access Protocol 3-3
Lightweight Directory Access Protocol (LDAP) 4-10
Log4j Audit Channel Provider 4-14
Log4j Audit Channel provider 4-18

M

Metadirectory 3-2

O

OpenLDAP 2-12
organizational node 5-5

P

password 4-3
perimeter authentication 4-6
policy
    authorization 5-12
    defined 5-1
Policy Database 3-2
Policy Distributor 3-2
policy elements 5-2
Policy Importer 3-2
policy inquiry 5-13
privilege
    defined 5-7
privilege group
    defined 5-7
providers 5-1

R

resource 5-1, 5-12
    attribute 5-6
    defined 5-4
    examples of 5-6
resource node 5-5
resources
    grouping of 5-5
role
    granting 5-12
role association

dynamic 4-8
Role Mapping 4-18
role mapping 4-7
Role Mapping provider 2-13, 4-18
Role Mapping providers
support for dynamic role associations 2-13
roles
dynamic association
result of 4-8

S

SAML 2-15
SAML Credential Mapping provider 4-18
SAML Identity Assertion provider 4-17
Secure Sockets Layer 2-7
security
administration 3-2
configuration 5-2, 5-3
managing 2-5
security providers 4-16
Adjudication
purpose 4-13
Authentication
purpose 4-1
Authorization
purpose 4-9
representation 5-4
Security Service Module 5-1, 5-3
embedded 3-3
support 3-4
Security Service Provider Interfaces 4-16
security, managing 5-1
Service Control Manager 5-1, 5-3
maintains configuration 3-4
Service Control Module
definition 3-3
Single Pass Negotiate Identity Asserter 4-17
single sign-on
using Identity Assertion provider and LoginModule 4-5
SSL
protocol
Secure Sockets Layer 4-3
SSPIs 2-14
standards 2-15
subject 4-6, 4-8, 4-12

T

token 4-6
token types supported 4-5
tokens
types
for identity assertion 4-5
supported 4-5

U

understanding evaluation functions 5-14
user 2-15
defined 5-9
directories 4-10
username 4-3

W

WebLogic Server 8.1 provider 4-19
WebLogic Server 8.1 providers 4-17

X

X.509 2-16
X.509 Identity Assertion provider 4-17