BEA WebLogic Enterprise Security™

Introduction to BEA WebLogic Enterprise Security

Version 4.2 Service Pack 2
Revised: September 22, 2005
Part Number: 887-001001-003

Copyright

Restricted Rights Legend

Trademarks or Service Marks

Contents

About This Document

1. Application Security Infrastructure

2. BEA WebLogic Enterprise Security Architecture

3. Security Services

4. Security Administration

Index

About This Document

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

This document is organized as follows:

- Chapter 1, "Application Security Infrastructure," introduces the BEA WebLogic Enterprise Security products and services, lists its key features, and provides an overview of their value.
- Chapter 2, "BEA WebLogic Enterprise Security Architecture," describes the BEA WebLogic Enterprise Security product architecture and its components.
- Chapter 3, "Security Services," describes security services as they relate to BEA WebLogic Enterprise Security products. This section includes discussions of auditing, authentication, authorization and role mapping, and credential mapping.
- Chapter 4, "Security Administration," discusses what a policy is and how you use the Administration Application to design policies.

Product Documentation on the dev2dev Web Site

The BEA WebLogic Enterprise Security documentation is available from the BEA dev2dev e-docs web site:

http://e-docs.bea.com/wles/docs42/index.html

Each link refers to a specific topic related to the product. For a complete list of documents available, see the Documents page on the e-docs web site:

http://e-docs.bea.com/wles/docs42/document.html

A PDF version of each document is also available on the BEA WebLogic Enterprise Security. For a complete list of documents available, see the Documentation Download page on the e-docs web site:

http://e-docs.bea.com/wles/docs42/download.html

You can print a copy of this document from a web browser, one file at a time, by using the File—>Print option on your web browser. You can open the PDF in Adobe Acrobat Reader and print the entire document (or a portion of it) in book format.

To download a single book, open the Documentation Download page, right click the document link, and then choose Save Target As from the menu.

To download all of the documents, open the Documentation Download page, right click one of the links listed under Zip Files, and choose Save Target As from the menu. Javadocs are saved as separate zip files that you can also download. Adobe Acrobat Reader is available at no charge from the Adobe Web site at http://www.adobe.com.

Related Information

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

- *BEA WebLogic Enterprise Security Policy Managers Guide*—This document defines the policy model used by BEA WebLogic Enterprise Security, and describes how to import and export policy data.
- *BEA WebLogic Enterprise Security Administration Guide*—This document provides a complete overview of the product and includes step-by-step instructions on how to perform various administrative tasks.
- *Javadocs for BLM API—*This document provides reference documentation for the Business Logic Manager (BLM) Application Programming Interfaces that are provided with and supported by this release of BEA WebLogic Enterprise Security. This API can be used to write and manage security policy (users, groups, roles, resources, and resource policies).
- *Programming Security for Java Applications*—This document describes how to implement security in Java applications. It includes descriptions of the Security Service Application Programming Interfaces and programming instructions.
- *Javadocs for Java API—*This document provides reference documentation for the Java Application Programming Interfaces that are provided with and supported by this release of BEA WebLogic Enterprise Security.
- *Programming Security for Web Services*—This document describes how to implement security in web servers using the Web Services Security Service Module. It includes descriptions of the Web Services Application Programming Interfaces.
- *Wslddocs for Web Services API—*This document provides reference documentation for the Web Services Security Service Module Application Programming Interfaces that are provided with and supported by this release of BEA WebLogic Enterprise Security.
- *Developing Security Providers for BEA WebLogic Enterprise Security* —This document provides security vendors and security and application developers with the information needed to develop custom security providers.
- *Javadocs for Security Service Provider Interfaces—*This document provides reference documentation for the Security Service Provider Interfaces that are provided with and supported by this release of BEA WebLogic Enterprise Security.

Contact Us!

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

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

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

- Your name, e-mail address, phone number, and fax number
- Your company name and company address
- Your machine type and authorization codes
- The name and version of the product you are using
- A description of the problem and the content of pertinent error messages

CHAPTER 1

Application Security Infrastructure

Introduction to BEA WebLogic Enterprise Security

This document summarizes the features of the BEA WebLogic® Enterprise Security™ products (version 4.2, Service Pack 2) and presents an overview of the architecture and capabilities of the security services. It provides a starting point for understanding the family of BEA WebLogic Enterprise Security products and security infrastructure.

This chapter covers the following topics:

Audience for This Guide

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

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

The BEA WebLogic 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.

Securing Your Enterprise

As the world's leading application infrastructure company, BEA supplies a complete platform for building, integrating, and extending J2EE applications to provide business solutions. Companies select the BEA WebLogic Enterprise Platform as their underlying software foundation to decrease the cost of information technology, leverage current and future assets, and improve productivity and responsiveness.

Now, BEA is extending its Application Security Infrastructure by offering the BEA WebLogic Enterprise Security product line–a family of security solutions that provides enhanced application security and includes: policy-based delegated administration, authentication with single sign-on, consolidated auditing, and dynamic-role and policy-based authorization with delegation.

BEA WebLogic Enterprise Security products are designed with an open and flexible standards-based framework that enforces security through a set of services. Resources and applications are protected by configuring and managing these services to meet the specific requirements of your business.

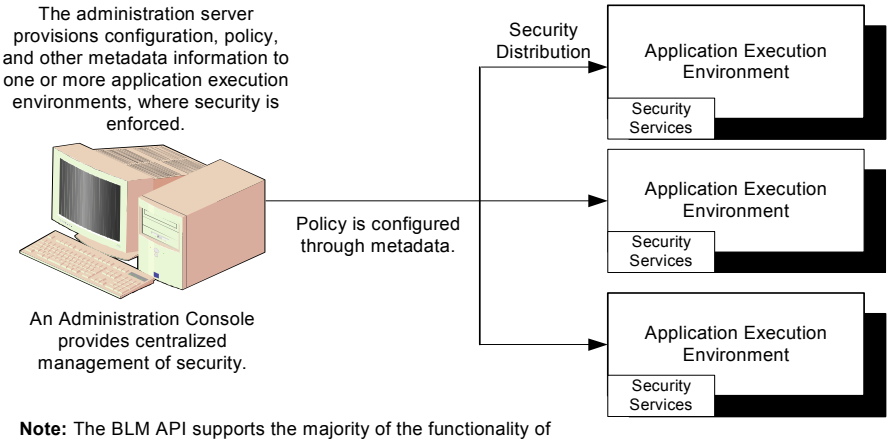
What is BEA WebLogic Enterprise Security?

The BEA WebLogic Enterprise Security product family provides an **application security infrastructure** consisting of an Administrative Application and a family of Security Service Modules for heterogeneous distributed environments. This infrastructure allows you to remove security technology and hard-coded policies from the application business logic. The security policy is no longer embedded in the application and the developer is no longer responsible for enforcing security policies through the application.

Using a distributed computing architecture, BEA WebLogic Enterprise Security products provide an enterprise-wide application security solution. The BEA WebLogic Enterprise Security product family is the only enterprise-wide application security infrastructure solution on the market.

The Administration Application allows you to centrally manage and distribute security configuration and policy as shown in Figure 1-1. In addition to the Administration Application, the Business Logic Manager (BLM) API is provided. This API supports the management of security policy (users, groups, roles, resources, and resource policies), but it does not support security configuration or policy distribution. When a security configuration and/or policy is changed, you must use the Administration Application to 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.

Figure 1-1 Typical Application Execution Environment



Note: The BLM API supports the majority of the functionality of the Administration Console, however, the BLM API does not support security configuration or policy distribution.

Key Features

The key features of the BEA WebLogic Enterprise Security product family include:

- ***Support for securing the enterprise***, including Security Service Modules for: BEA WebLogic Platform 8.1, Internet Information Services (IIS) and Apache web servers, web services, and Java applications. Future Security Service Modules will support a wider variety of platforms and environments.
- ***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 with delegation***, leads to lower administration costs. Security administrators can define and deploy security policies without the need for re-coding or re-deploying applications. Rules and policies are all managed through a central Administration Console. In addition, policy verification and policy inquiry functions allow the administrator to validate security policy implementations prior to deployment.
- ***Centralized BLM API for policy definition***, provides programmatic access to the WLES policy management infrastructure. This is a Java API that uses SOAP to communicate with the central WLES management services. The API supports the majority of the functionality of the Administration Console (allowing for creating and management of users, groups, roles, resources, and resource policies), however, the BLM API does not support the distribution of either security configuration or policy. The Administration Console must be used for that purpose. For more information on the BLM API, see the BLM API Javadocs.
- ***A rules-based policy model***, with integrated authentication, authorization and auditing. Administrators can design security policies to model their own business policies, and then implement, test and distribute them through a centralized Administration Application. You can include both resource and user attributes in your rules, along with various 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), 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.

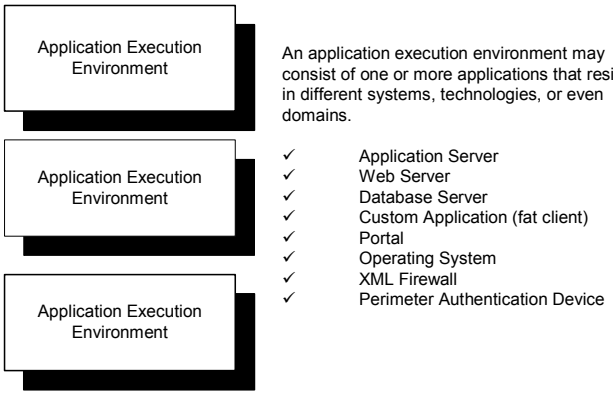
What is the Problem?

With the rush to build web-based applications and market services over the Internet, many developers had little comprehension of the security issues they would 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.

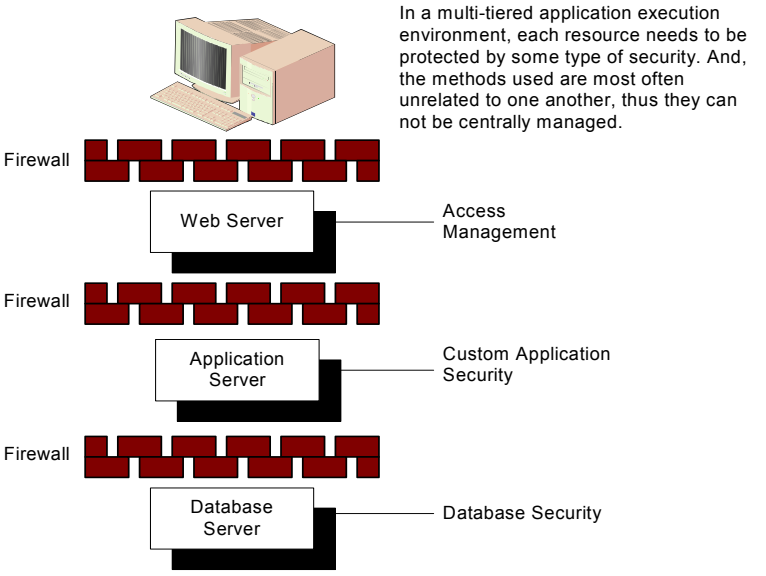
Because these applications often comprise a number of different components that may or may not reside on the same server or even in the same domain, policy management becomes extremely difficult and ensuring enterprise or regulatory compliance can prove impossible.

Figure 1-2 Application Execution Environment



A typical application execution environment is multi-tiered as shown in Figure 1-3 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 security policy and technology for each one may be different and you need to be able to enforce security at each layer.

Figure 1-3 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 policy 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 policy.

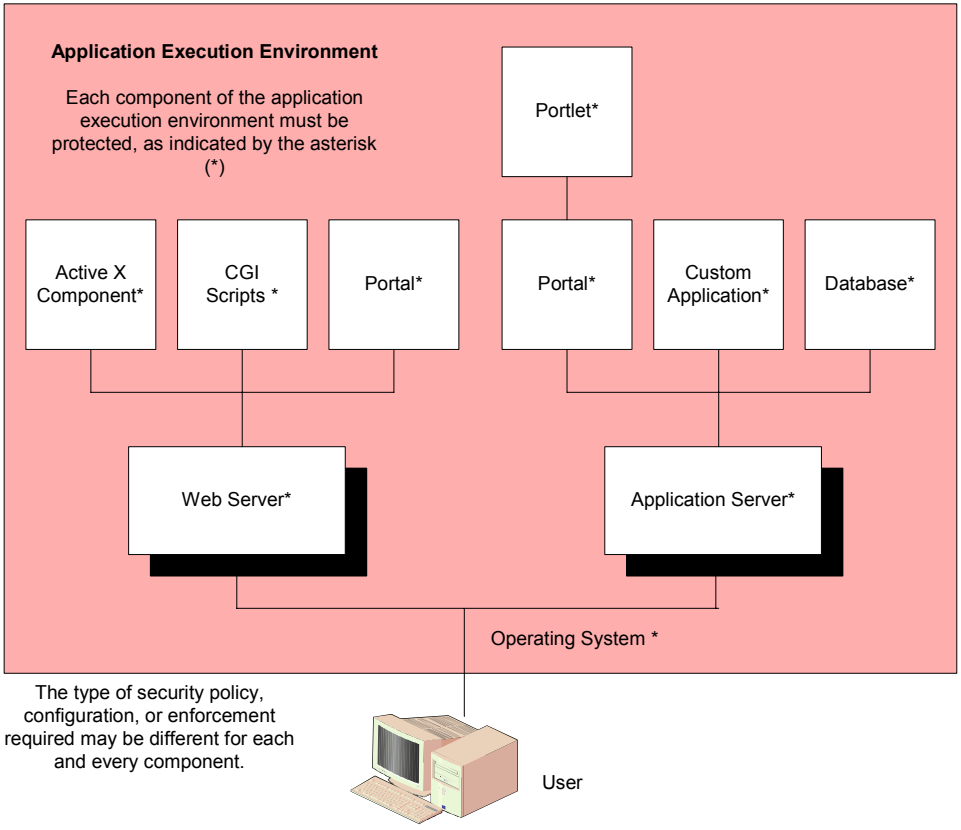
Early authorization implementations used static and inflexible approaches to define the different types of access granted or denied for a user. Because this type of implementation is extremely time-consuming (if only due to the number of users and the different types of user storage methods in use), 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** 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 1-4 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 WebLogic Enterprise Security provides application security for additional platforms and web servers.

Figure 1-4 Integrated Application Security

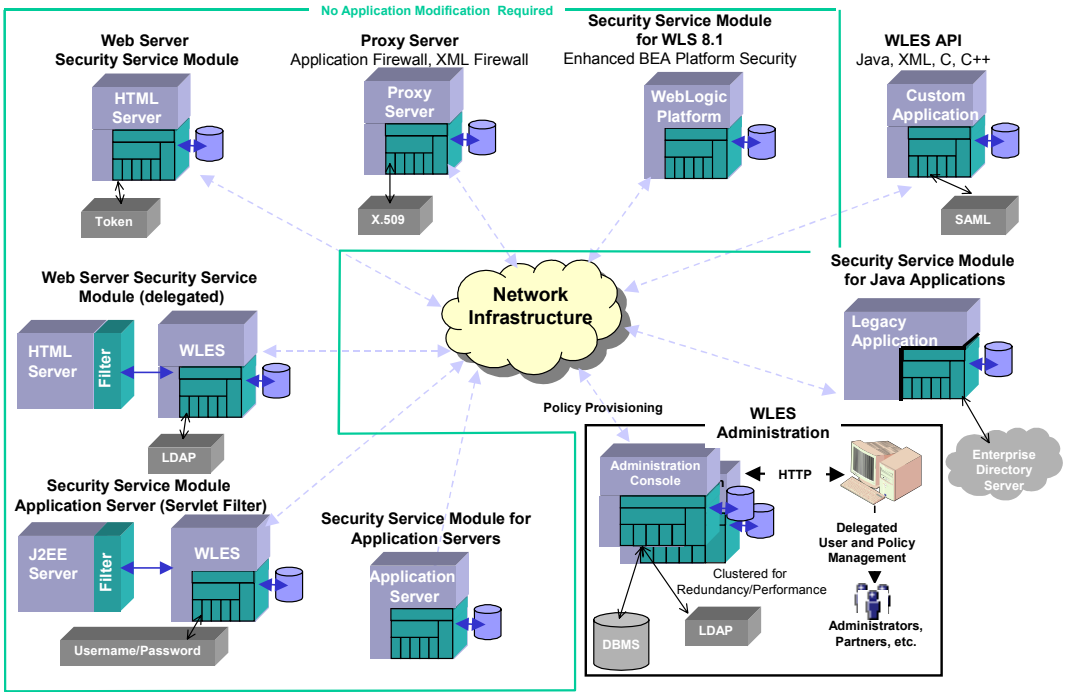


The open flexible architecture of the BEA WebLogic Enterprise Security products provides advantages to all levels of users and introduces an advanced design for securing your applications. With distributed computing, applications must be integrated across the network, as shown in Figure 1-5. BEA WebLogic 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 1-5, the BEA WebLogic Enterprise Security products support a fully distributed architecture; integrating all applications across the network.

Figure 1-5 Distributed Computing Security Infrastructure Vision



The BEA WebLogic Enterprise Security products provide a variety of services that use the WebLogic 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. A set of security providers delivered with each Security Service Module provide the ability to configure and enforce each security service, using the Administration Console described in "Security Administration" on page 4-1.

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 Version 8.1, Internet Information Services (IIS) and Apache Web Servers, web services, and Java applications. BEA WebLogic 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.

- IIS Web Server Security Service Module

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

- Apache Web Server Security Service Module

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

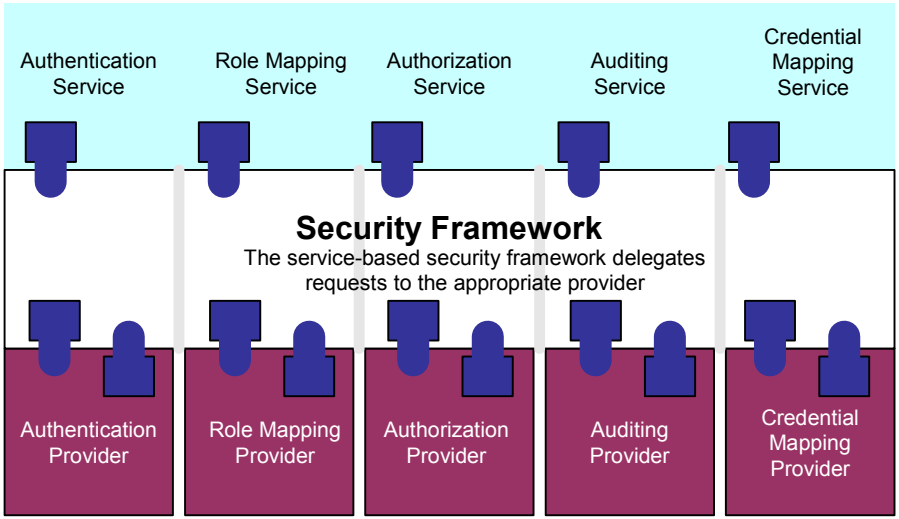
- Web Services Security Service Module

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

- Java Security Service Module

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

Figure 1-6 Security Framework



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

Table 1-1 BEA WebLogic 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. Authentication providers also support principal validation.
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 1-1 BEA WebLogic 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 WebLogic 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). These APIs 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 Application 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 security 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 WebLogic 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 security 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 security 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 WebLogic 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 WebLogic Enterprise Security products adhere to the following standards.

Table 1-2 BEA WebLogic 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.
JSSE	Protect infrastructure network connections for establishment of mutual trust.
JCE	Integrate cryptographic libraries.
JAAS	Provide authentication service implementations.
Miscellaneous Standards	**Used to**
X.509	Validate the identity of infrastructure components through digital certificates; supported as proof of identity for customer use.

Table 1-2 BEA WebLogic Enterprise Security Standards (Continued)

XML Standard	Used to
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.

CHAPTER 2

BEA WebLogic Enterprise Security Architecture

Architectural Overview

This section describes the general architecture of the BEA WebLogic Enterprise Security services, providers and service modules. Each Security Service Module you purchase is shipped 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 WebLogic 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 Application 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, discussed in the following sections:

- Administration Server
- Service Control Manager
- Security Service Module

Administration Server

BEA WebLogic Enterprise Security lets you control all of your security policies and configuration data from a single console. The Administration Application gives you the enterprise-wide visibility you need to analyze security policies and ensure that applications and resources are properly protected. BEA WebLogic 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 Application features, see Chapter 4, "Security Administration." The Administration Server consists of several components (as shown in Figure 2-1), including:

Policy Distributor–Ensures that the correct policies are provided to the correct Security Service Modules and maintains policy synchronization.

Policy Database–Maintains security policy data managed by the Administration Application 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.

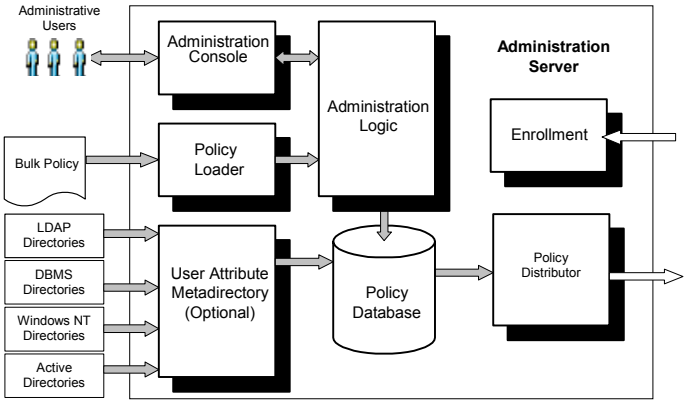
Policy Importer–Imports policy data from an external file, generated in another system, exported from another instance of an Administrative Application, or manually coded. For additional information on how to use the Policy Loader, see the *Policy Managers Guide*.

Administrative 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 WebLogic 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 3-14. BEA WebLogic 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 2-1 Administration Server Architecture



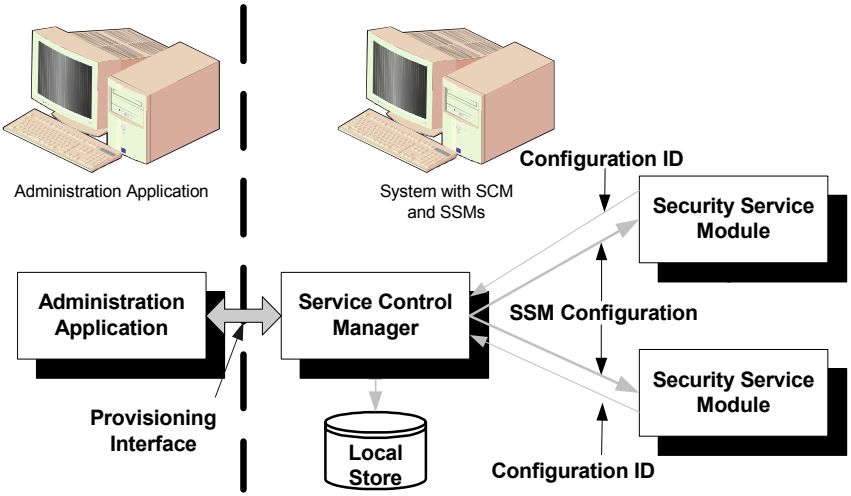
Service Control Manager

BEA WebLogic 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 2-2). To facilitate the management of a potentially large number of distributed Security Service Modules, the Administration Application uses a remote administration mechanism to distribute appropriate configuration and policy data to each Security Service Module.

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 Application, 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 Application. 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 Application goes down or is intentionally unavailable, such as in occasionally connected computing environments.

Figure 2-2 Service Control Manager



Security Service Module

BEA WebLogic Enterprise Security supports a variety of Security Service Modules that you integrate with the security framework and provision as needed. 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 WebLogic Enterprise Security services, see Chapter 3, "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 Application as shown in Figure 2-2. The distributed nature of the architecture allows you to configure, manage and distribute policy throughout the enterprise.

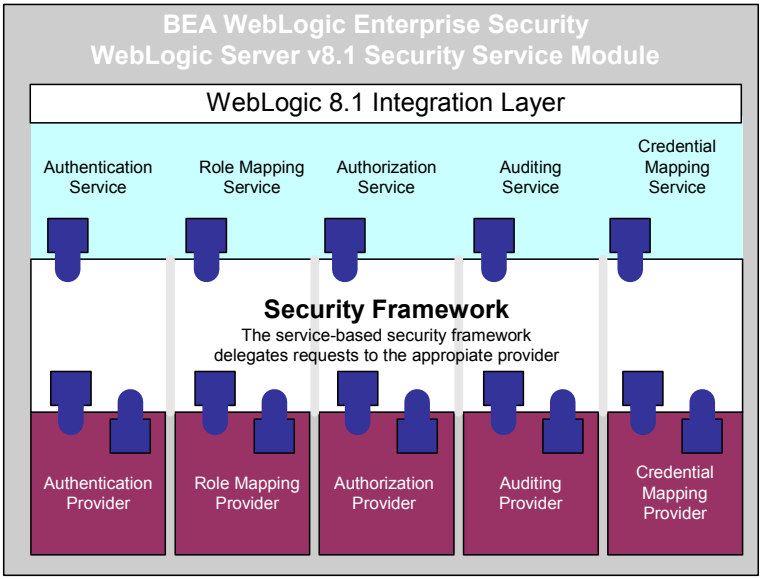
Configuration data for each Security Service Module is maintained within each machine and handled by a Service Control Manager. One additional benefit of this architecture is that even if the administration server goes down (either for maintenance or due to failure), there is no impact on the applications 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 2-5
- "Web Server Security Service Module" on page 2-5
- "Web Services Security Service Module" on page 2-7

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 Application so that all application server administrative security activities are performed through the Administration Application. 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 2-3 shows the major components of the WebLogic Server 8.1 Security Service Module.

Figure 2-3 WebLogic Server 8.1 Security Service Module Architecture



Web Server Security Service Module

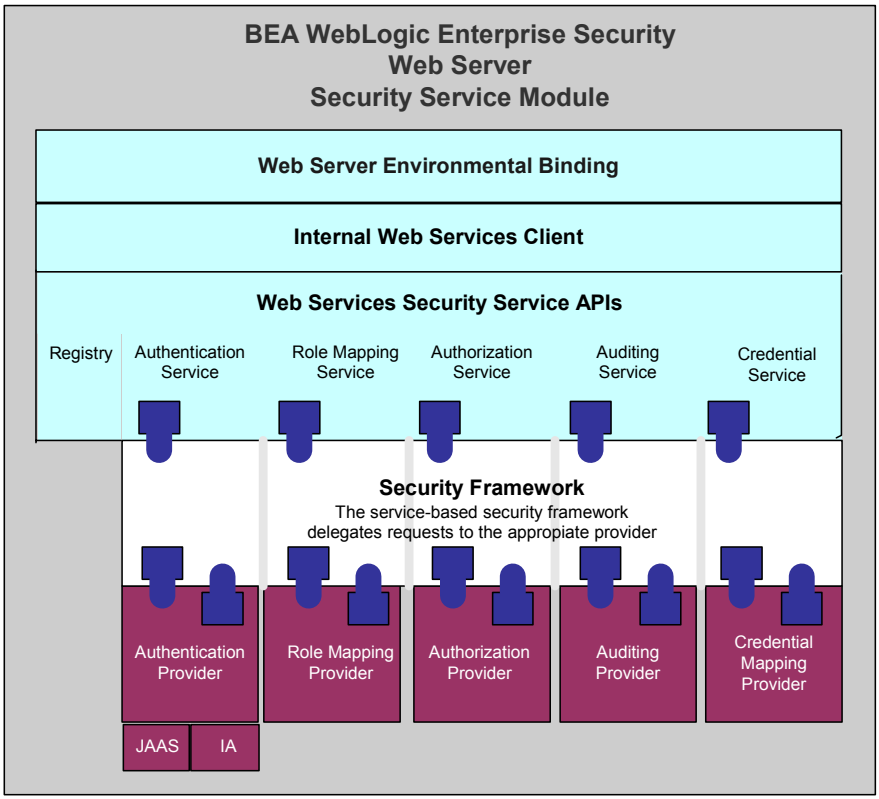
The ProductName Security Service Module (SSM) provides an environmental binding between the WebLogic Enterprise Security infrastructure and IIS and Apache web servers. The SSM consists of three components, a Web Server Environmental Binding, an Internal Web Services

Client, and the Web Services SSM (which includes the Security Service APIs, Security Framework and security providers) (See Figure 2-4). The WebLogic 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 WebLogic 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 2-4 Web Server SSM Components



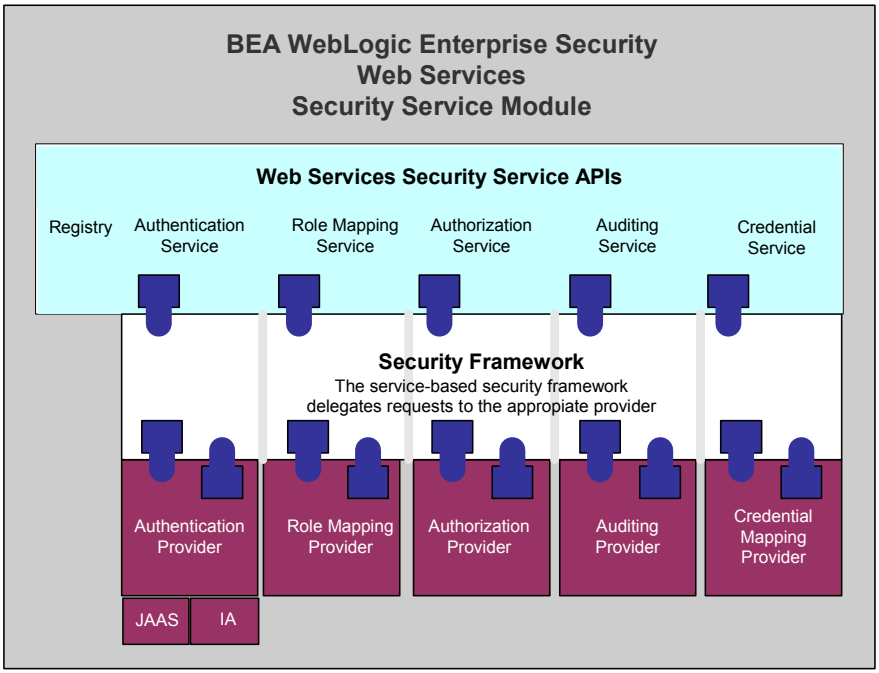
Web Services Security Service Module

The Web Services Security Service Module (SSM) provides provides six security service APIs: Registry, Authentication, Authorization, Auditing, Role Mapping, and Credential Mapping (see Figure 2-5). These APIs can be used to developed web services clients to access the WebLogic Enterprise Security infrastructure and use it to make access control decisions for users attempting to access web server application resources. Once the web services client is implemented, it uses the Web Services SSM (which incorporates the Security Services APIs, the Security Framework, and the configured security providers) to make access control decisions for the web server to which it is connected. You use the WebLogic Enterprise Security Administration Application to configured and deploy a security configuration to protect the web server application resources.

Thus, the Web Services SSM enables security administrators and web developers to perform security tasks for applications running on a web server. Additionally, you can use the Web Services SSM to add information provided by the Security Framework (such as roles and response attributes) to the HTTP requests handled by the protected web server applications.

Figure 2-5 shows the components of the Web Services SSM. These components are the same as corresponding components in the Web Server SSM.

Figure 2-5 Web Services SSM Components

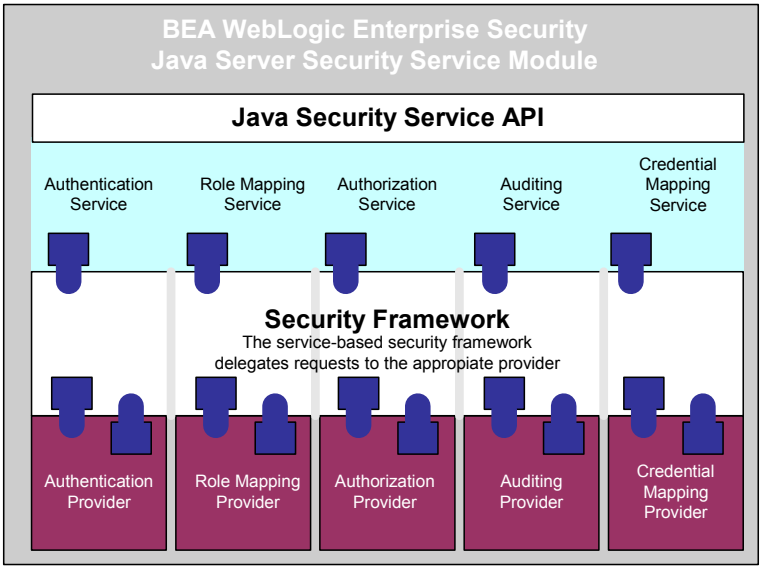


Java Security Service Module

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 3, "Security Services." Figure 2-6 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 2-6 Java Security Service Module Architecture



CHAPTER 3

Security Services

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

- Authentication and Identity
- Role Mapping
- Authorization
- Auditing
- Credential Mapping
- Security Service Providers

Authentication and Identity

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 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. During the authentication process, a Principal Validation provider supports additional security protections for the principals

(users and groups) contained within the subject, by first signing and later verifying the authenticity of those principals for each use.

Authentication Service

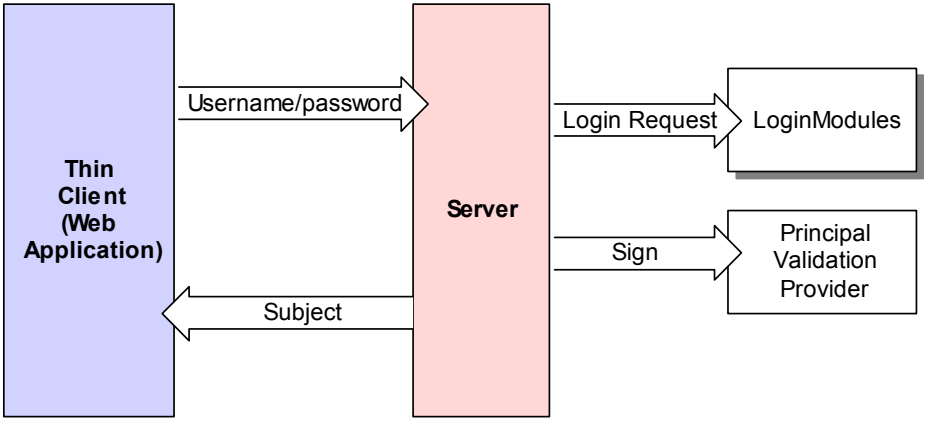
The Authentication providers support numerous methods of authentication services, including user username/password and identity assertion. In addition, a set of Authentication providers is supplied that access various types of user directories and identity tokens.

Figure 3-1 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 and a Principal Validation provider, as described in "Identity Assertion" on page 3-9.

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 WebLogic Enterprise Security supports any number of LDAP servers or a database.

Figure 3-1 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 3-3
- "Web Single Sign-On" on page 3-3
- "Perimeter Authentication" on page 3-5

You can use one of the Authentication providers provided as part of BEA WebLogic 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 checks the information and, if it is trustworthy, grants access to the protected resource.

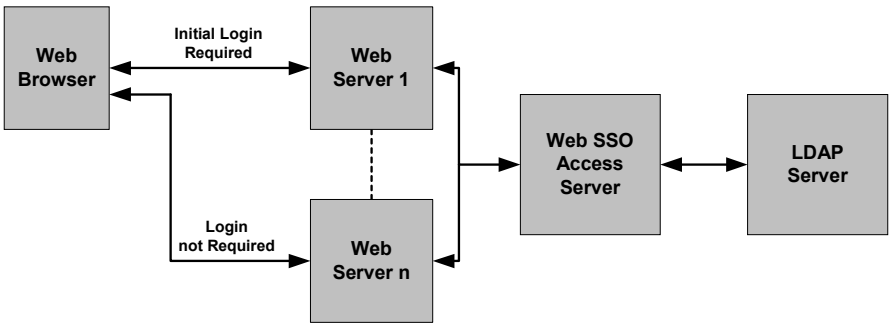
You can use Secure Sockets Layer (SSL) or Hyper-Text Transfer Protocol (HTTPS) to provide an additional level of security to username and password authentication. Because SSL encrypts the data transferred between the client and the Security Framework, the username and password are encrypted and not transferred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

Web Single Sign-On

Web single sign-on is an access-control mechanism that enables users to log on to one web server and gain access to other web servers in the same domain or other domains without supplying

log-in credentials again, even though the other web servers may have their own authentication schemes or requirements. Figure 3-2 shows the basic components of a web single sign-on service.

Figure 3-2 Web Single Sign-on



While web single sign-on removes the requirement for users to log on every time they access a different web server and thus makes user access easier, it does not improve security. In fact, you should take security requirements into consideration when implementing a web single sign-on solution.

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.

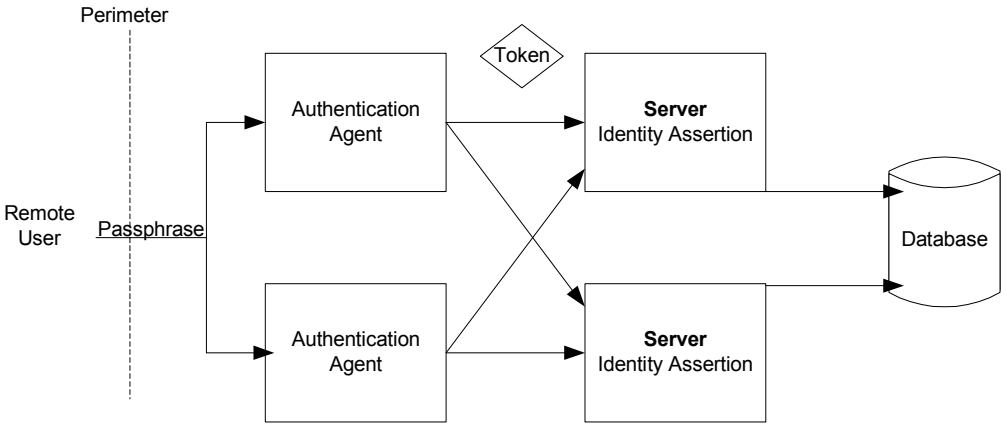
How is Perimeter Authentication Accomplished?

BEA WebLogic Enterprise Security services are designed to extend the single sign-on concept all the way to the perimeter by using identity assertion (see Figure 3-3). 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 3-3 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.

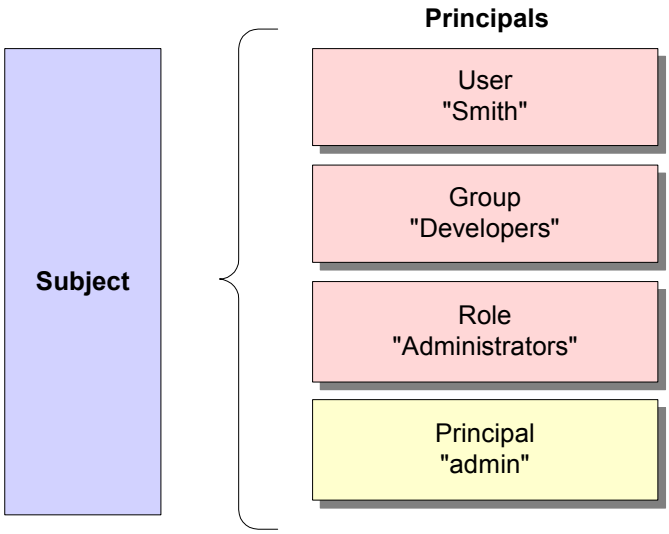
The following sections describe additional authentication concepts and functionality.

- Principal Validation
- Identity Assertion

Principal Validation

Subjects and principals are closely related. A principal is an identity assigned to a user or group as a result of authentication. Both users and groups can be used as principals by application servers. The Java Authentication and Authorization Service (JAAS standard) requires that subjects be used as containers for authentication information, including principals. Figure 3-4 illustrates the relationships between users, groups, principals, and subjects.

Figure 3-4 Relationships Between Users, Groups, Principals and Subjects



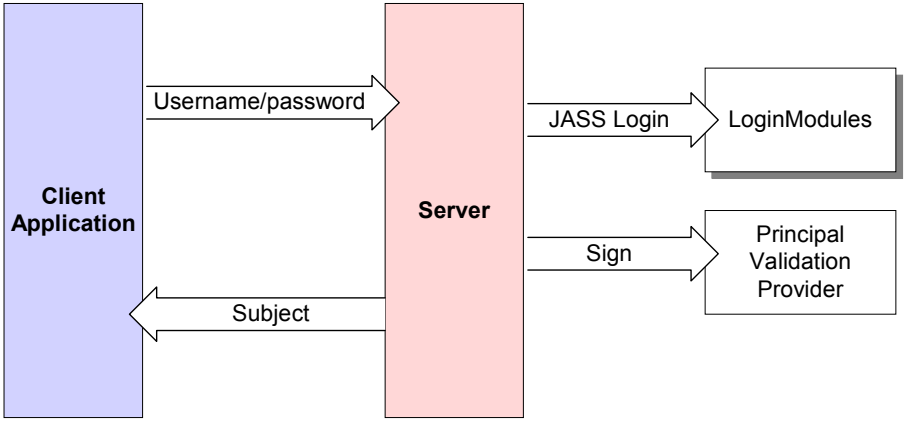
Principal Validation Service

Much as an Identity Assertion provider supports a specific type of token, a Principal Validation provider both signs and verifies the authenticity of a principal that represents a user or group. This provides an additional layer of protection for the user identity within the service module.

Note: You do not use the Administration Console to configure Principal Validation providers directly. BEA WebLogic Enterprise Security provides the required Principal Validation providers for you when you configure your Authentication provider.

As shown in Figure 3-5, a user attempts to log into a system using a username and password combination. After the authenticity of the user is verified, the principal validation service signs the principal. The principal validation process then passes the response to the specified Principal Validation provider, which signs the principals and then returns them to the client application. Whenever the principals stored within the subject are required for other security operations, the same Principal Validation provider verifies that the principals stored within the subject were not modified since they were signed.

Figure 3-5 Principal Validation Service



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 policy 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 WebLogic 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 3-6), a token from outside of the policy 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.

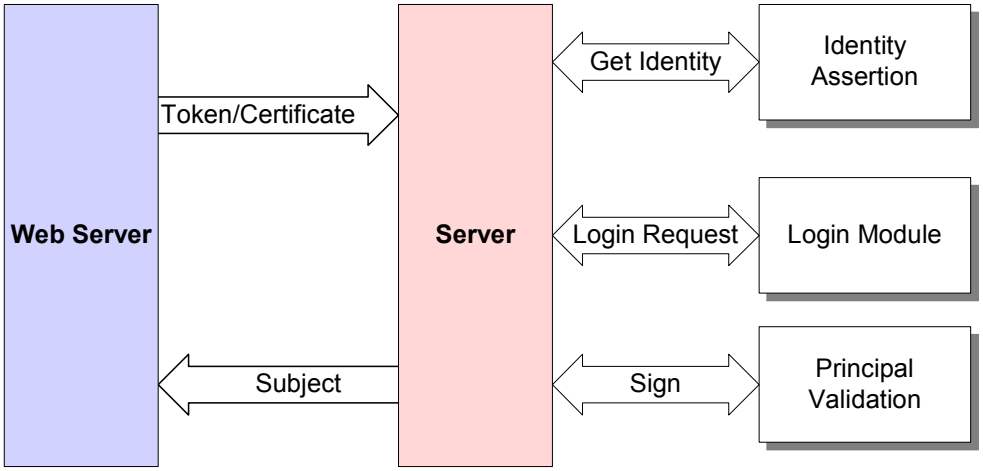
You can configure multiple Identity Assertion providers to support different types of tokens within the same policy domain, but you do not need one unless you are supporting perimeter authentication. Two Identity Assertion providers are supplied: one supports X.509 and IIOP-CSIv2 and one supports SAML. 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 3-6 Identity Assertion Example



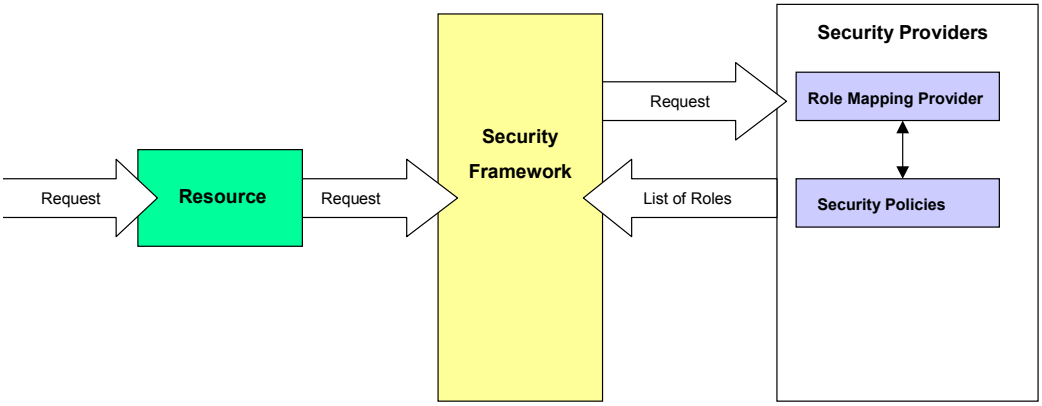
Role Mapping

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 3-7 shows how the Role Mapping provider interacts with the Security Framework to create dynamic role associations.

Figure 3-7 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 a security 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.

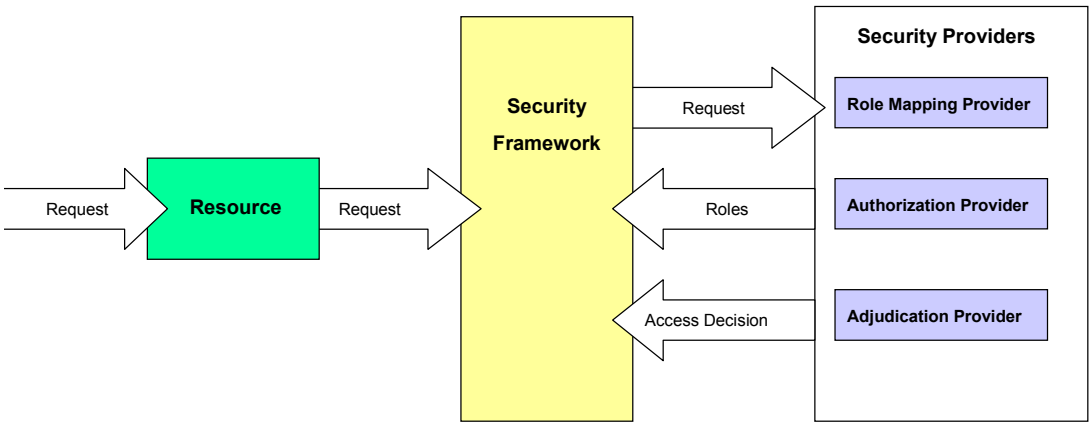
Authorization

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 indicate, *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 security role and the security policy assigned to the requested resource. Figure 3-8 illustrates how an Authorization provider, and the associated Adjudication and Role Mapping providers, interact with the Security Framework during the authorization process.

Figure 3-8 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 role data–that is, the Authorization provider makes an access decision.

The following sections describe authorization concepts and functionality:

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

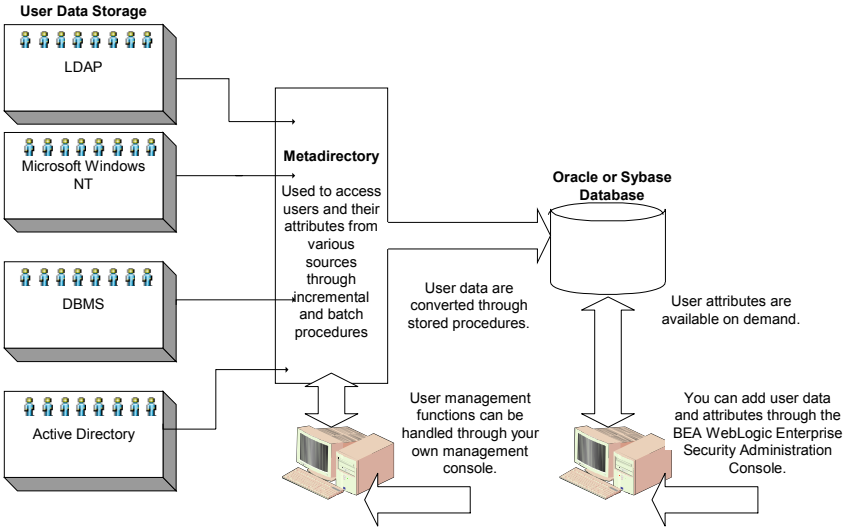
User Directories

BEA WebLogic 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 WebLogic 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 3-9 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

Authorization Policies and Role Mapping

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 WebLogic Enterprise Security authorization policies replace ACLs through the use of role-based access control. 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 rules and the permissions 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. By using role-based access control (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 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 policy to a type of resource, all new instances of that resource inherit that policy.

To use a user or group to create a security policy, the user or group must be defined for the Authentication 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 security policy, the role must be defined for the Role Mapping provider that is configured for that service module. All security policy data are stored in the policy database. When the Authorization provider is configured, its configuration is also stored in the policy database.

Note: Although you have the option of basing a security policy on a user or group, BEA recommends basing security policies on roles. Basing security policies on security 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 policy 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 WebLogic 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.

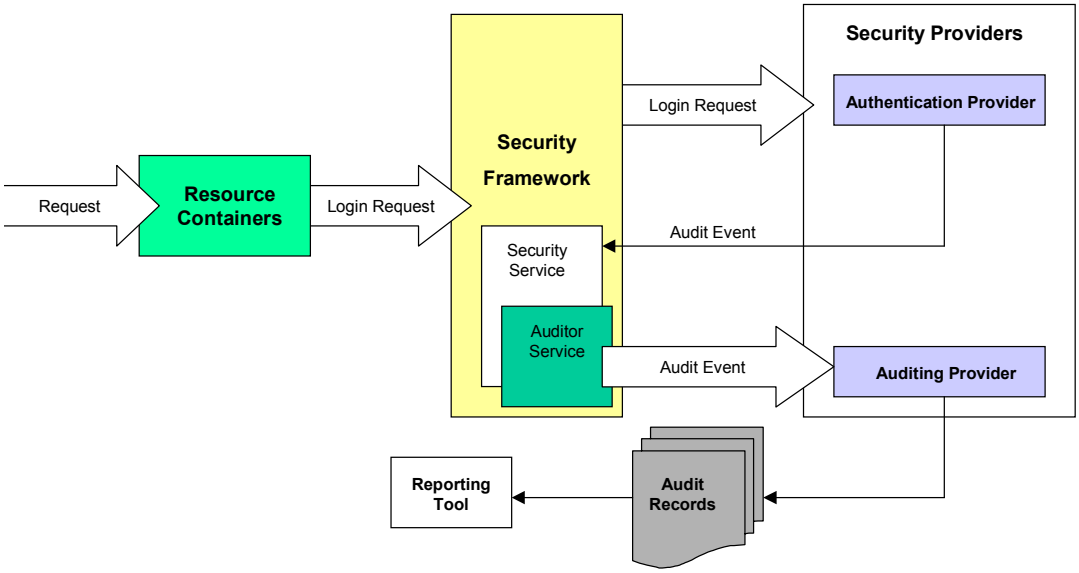
Auditing Service

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 3-10 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 3-10 Auditing Service Example for Authentication



Credential Mapping

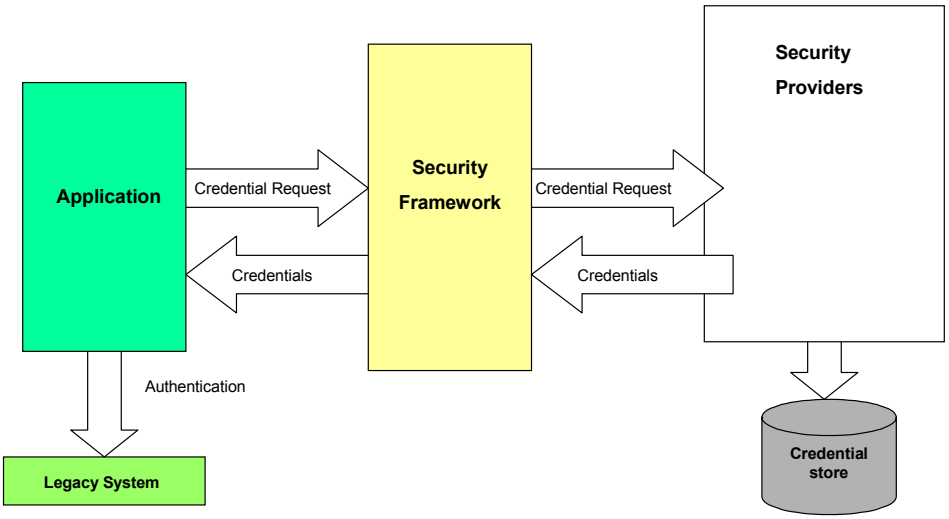
Credential mapping is the process of looking up the authentication credentials associated with a user for a given resource.

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 3-11 illustrates how the Credential Mapping provider interacts with the Security Framework to support the credential mapping service.

Figure 3-11 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 WebLogic Enterprise Security. SSPIs are available for Authorization, Role Mapping and Adjudication; Auditing; Authentication, Principal Validation and Identity Assertion; and Credential Mapping (for a description of providers, see Table 3-1). 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 3-1 Security Providers

Provider Type	Provider Name	Description
WebLogic Enterprise Security Providers		
Authentication	Windows NT Authentication Provider	Supports Microsoft Windows NT authentication.
	Active Directory Authentication Provider	Supports external LDAP authentication.

Table 3-1 Security Providers (Continued)

Provider Type	Provider Name	Description
	Netscape Authentication Provider	Supports external LDAP authentication.
	Novell Authentication Provider	Supports external LDAP authentication.
	OpenLDAP Authentication Provider	Supports external LDAP authentication.
	Database Authentication Provider	Supports authentication through a relational database.
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 v1.0.
	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.
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.
	Resource Deployment Audit Provider	Publishes WebLogic resources to the Administration Application.

Table 3-1 Security Providers (Continued)

Provider Type	Provider Name	Description
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 security policy with which it is used. The Role Mapping Provider returns a set or roles granted to a user on a protected resource.
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 security policy in WebLogic Server. The Authorization Provider returns an access decision that determines which resources are protected and if a particular user is allowed access to a resource.
	WebLogic Role Mapping Provider	Provides assignment to roles based on the security 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: 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 WebLogic Enterprise Security*.

CHAPTER 4

Security Administration

Managing Security

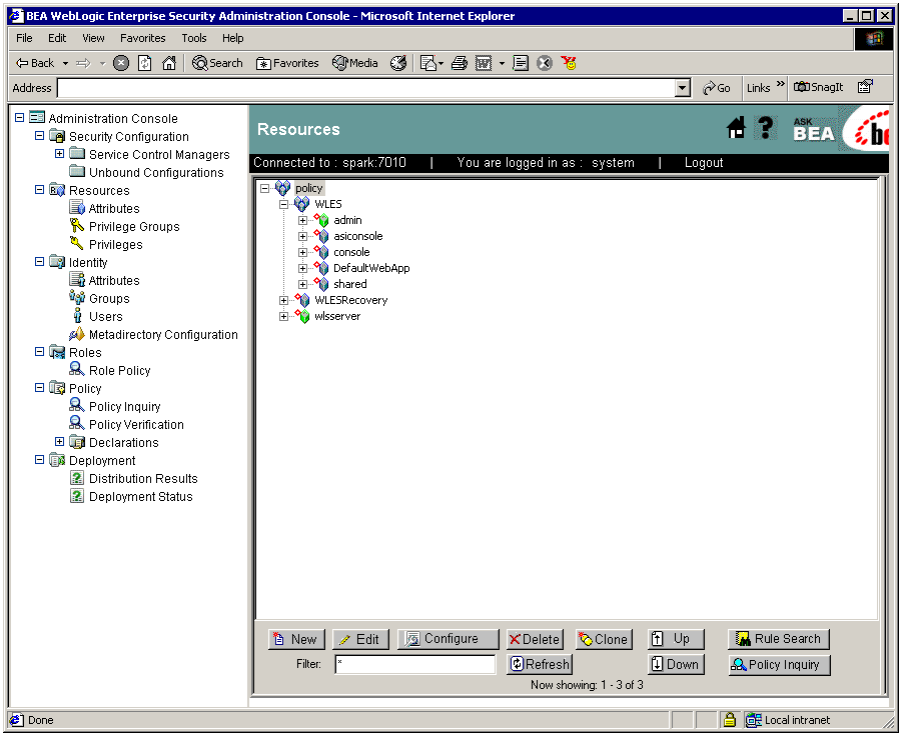
You manage security by using the Administration Console to configure Service Control Managers, Security Service Modules and their providers, and to design, edit, and distribute security policies. A policy is composed of a set of rules, modeled on your organizational business requirements that state who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

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

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

Figure 4-1 shows how the Administration Console represents the various policy elements and resources. The resources shown here represent the initial policy settings used to protect the console.

Figure 4-1 Administration Console Resource Representation



Security Configuration

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

Table 4-1 Security Providers and their Representations

	Adjudication		Authorization
	Auditing		Credential Mapping
	Authentication		Role Mapping

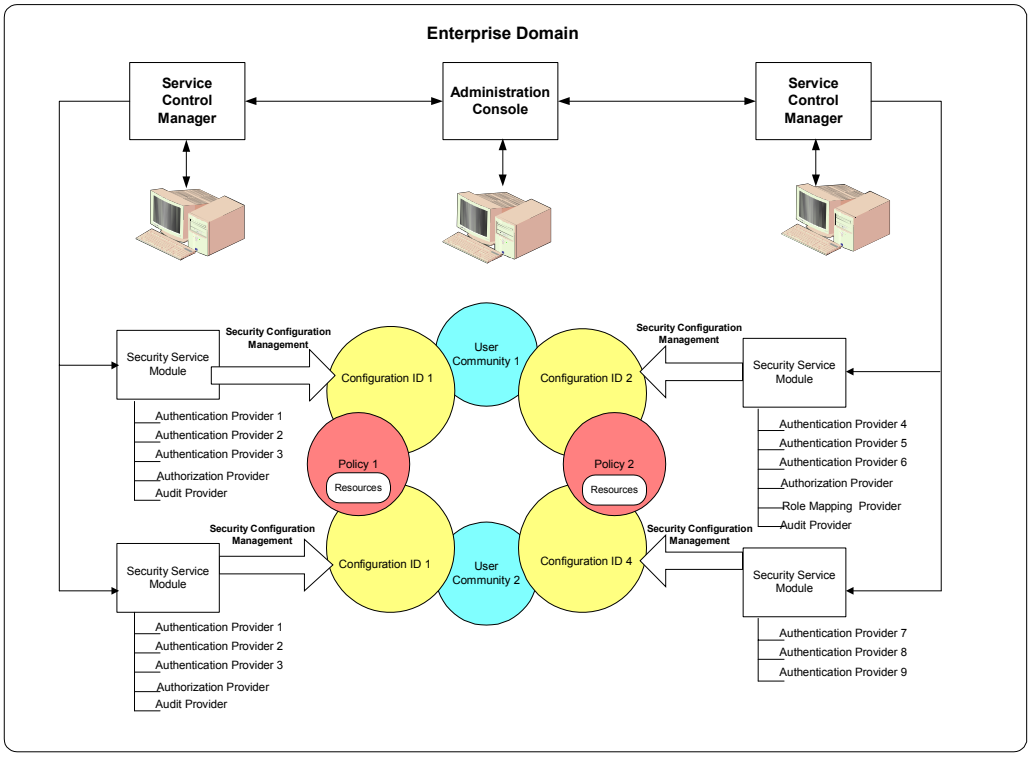
To learn more about security providers and their services, see "BEA WebLogic Enterprise Security Architecture" on page 2-1. For information on how to develop custom security providers, see *Developing Security Providers for WebLogic Enterprise Security*.

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

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

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

Figure 4-2 Policies and Security Configuration



Resources

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

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

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

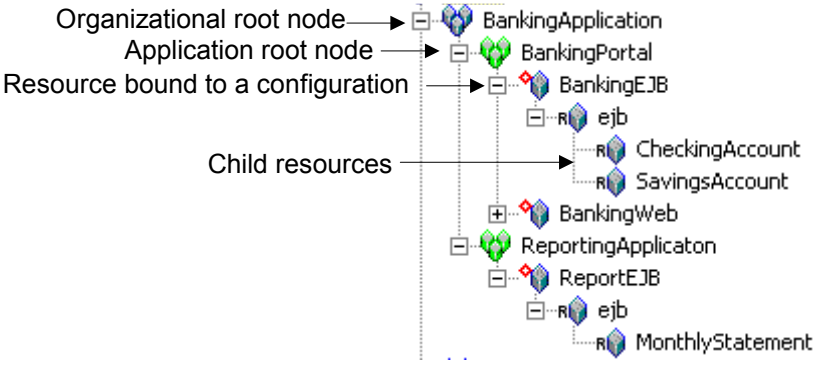
Another type of organizational node is an application node represented by a green resource icon  . This icon depicts a collection of resources that provide a specific set of services. Applications are considered organizational because they often consist of multiple, distributed components, such as an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

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

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

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

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



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

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

Resource Attribute

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

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

Privilege

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

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

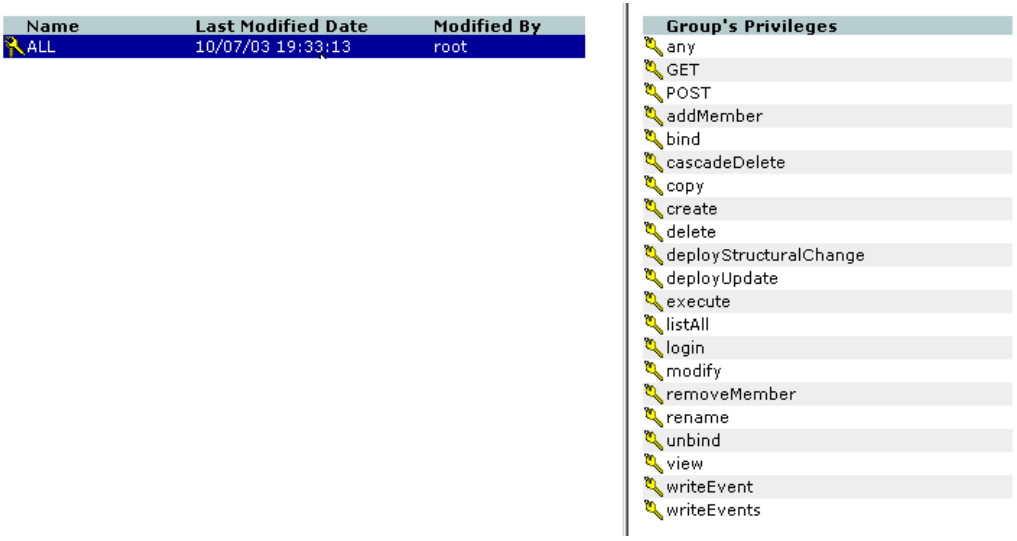
Figure 4-4 Privilege Representation in the Administration Console

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

Privilege Group

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

Figure 4-5 Privilege Group Representation in the Administration Console



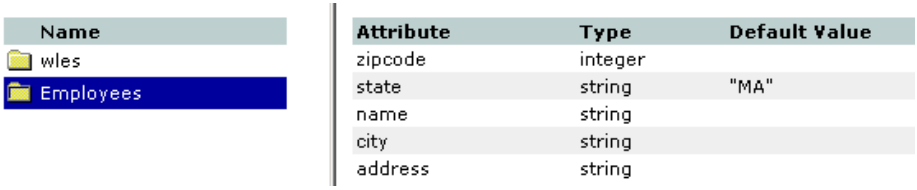
Identity

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

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

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

Figure 4-6 Directory Representation in the Administration Console



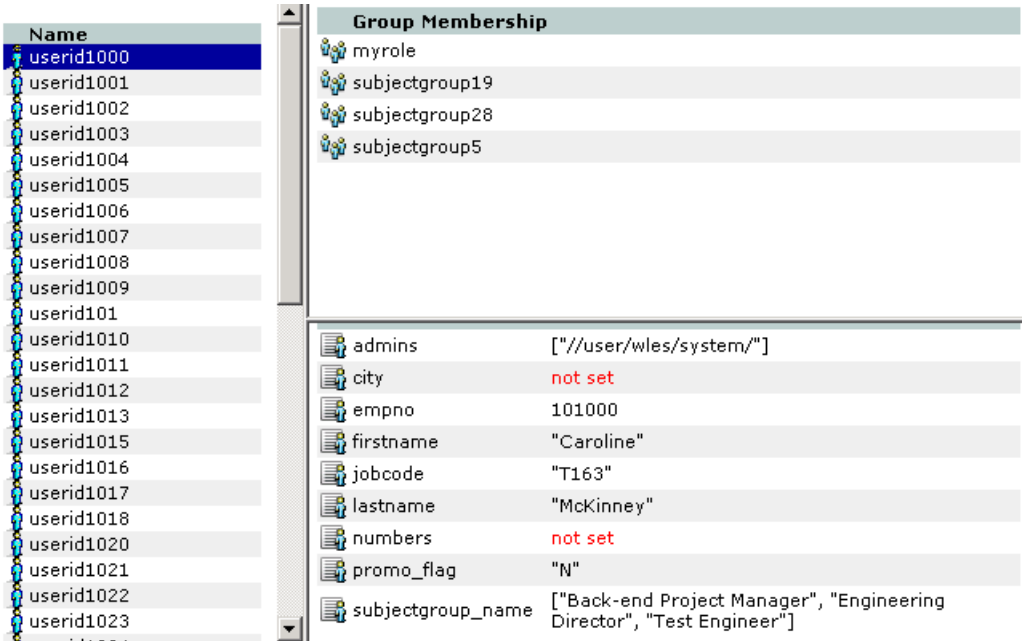
The number of directories you have depends on the level of granularity needed to separate your user community. You may want to have one global directory containing all users. In this case, you can populate a single directory using multiple external repositories. Or, you might want separate directories—one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

User

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

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

Figure 4-7 User Representation in the Administration Console



Group

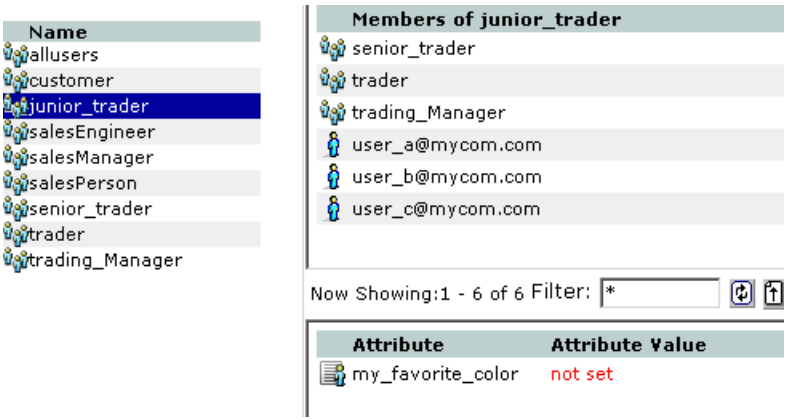
A group is represented by the  icon in the Administration Console and is a logical collection of users that share some common characteristics, such as department, job function, or job title. For example, a company may separate its sales staff into two groups, Sales Representatives and Sales Managers,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

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

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

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

Figure 4-8 Group Representation in the Administration Console



Identity Attribute

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

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

Role

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

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

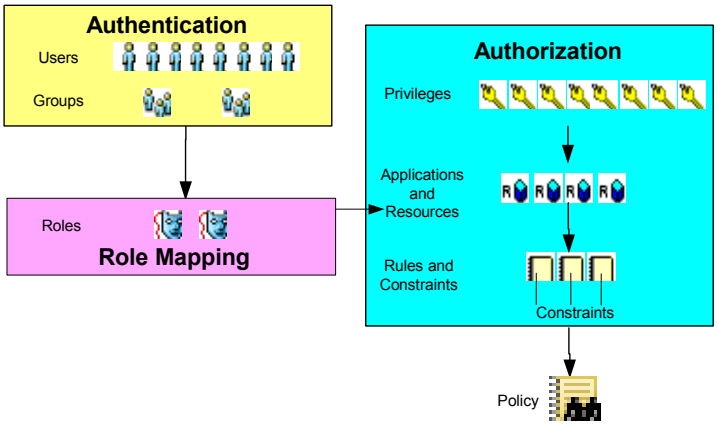
Role Policy

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

Policy

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

Figure 4-9 Role-based Authorization Policy



Policy Rule

A policy rule is represented by the blue check mark icon in the Administration Console. A policy rule defines the privileges, resources, subjects (users, groups, and roles), and constraints that apply to the rule. BEA recommends that authorization rules be written to grant privileges to roles, rather then granting them directly to users or groups.

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

Policy Inquiry

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

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

Policy Verification

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

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

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

Declarations

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

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

What's Next?

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

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

Index

E

F

G

H

I

J

L

M

N

O

P

R

S

T

U

W

X