

Entrust® Application Server Runtimes 7.0

Administration Guide

Document issue: 1.0

Date: June 2003

© 2003 Entrust. All rights reserved.

Entrust is a trademark or a registered trademark of Entrust, Inc. in certain countries. All Entrust product names and logos are trademarks or registered trademarks of Entrust, Inc. in certain countries. All other company and product names and logos are trademarks or registered trademarks of their respective owners in certain countries.

This information is subject to change as Entrust reserves the right to, without notice, make changes to its products as progress in engineering or manufacturing methods or circumstances may warrant.

Export and/or import of cryptographic products may be restricted by various regulations in various countries. Licenses may be required.

Table of contents

CHAPTER 1

About this document	1
Related documents	2
Entrust Identification Server 7.0	2
Entrust GetAccess 7.0	2
Typographic conventions	3
Note and Attention text	3
Getting help	4
Technical Support	4
Telephone numbers	4
E-mail address	5
Online	5
Professional Services	5

CHAPTER 2

About Entrust Application Server Runtimes 7.0	7
Overview	8
Web application servers	9
J2EE application environment	10
EJBs	10
Development roles	10
EJB security	12
Declarative security model	12
Programmatic security model	12
Security roles	13
Application Server Runtime for WebLogic	15
WebLogic Server domains	15
WebLogic Server security realms	15
Entrust security realm	15
Authentication providers	16
Servlet filters	17

Role mapping providers	17
Using a Java client to access protected resources	18
Using a Web browser to access protected resources	19
Application Server Runtime for WebSphere	21
WebSphere Application Server security model	21
User Registry	21
Trust Association Interceptor	21
Authorization Table	22
Using a Java client to access protected EJBs	22
Using a Web browser to access protected resources	24

CHAPTER 3

Installing Application Server Runtimes37

Preparing to install Application Server Runtimes	28
Dependencies	28
Downloading the installer application	29
Installing Application Server Runtimes	30
Installing on a Windows operating system	30
To install using the GUI	30
To install from the command line	32
Installing on a Solaris operating system	32
To install using the GUI	33
To run a command line installation.	35
Performing post-installation tasks	35

CHAPTER 4

Performing post-installation tasks37

Overview	38
Post installation steps — WebLogic Server	39
Creating an Entrust security realm	39
To create an Entrust realm using the createEntrustRealm script	39
Configuring a domain to use the Entrust Realm	41
To configure an Entrust realm in a WebLogic domain.	41
Setting up Application Server Runtime software in other WebLogic domains	41

To set up Application Server Runtime software in additional WebLogic domains	41
Setting up the Application Server Runtime in advanced WebLogic Server configurations	42
On the computer that hosts the WebLogic domain Administration Server	42
On computers that host the WebLogic domain Managed Servers	43
Post installation steps — WebSphere Application Server	44
Setting up Entrust GetAccess roles and users	44
To set up Entrust GetAccess users and roles.	45
Configuring WebSphere Application Server security	46
To configure an Entrust security realm	46
To configure Lightweight Third Party Authentication.	48
To configure Global Security settings for WebSphere Application Server	48
To stop WebSphere Application Server when security is enabled	49
Command console users	50
To add a console user with the addConsoleUser script.	51
Setting up the Application Server Runtime in advanced WebSphere configurations	52
On the computer that hosts the WebSphere Deployment Manager	52
On computers that host the WebSphere Application Server Network Deployment nodes	53
Disabling WebSphere sample applications	53
To disable WebSphere sample applications	54
To prevent sample applications from restarting when WebSphere Application Server restarts	54
Configuration	55
Configuration file	55
Editing the configuration file	55
Configuration settings	55
Confirming that Application Server Runtime software is working	61

CHAPTER 5

Verifying installation63

Overview	64
Related documentation	64
WebLogic Server	64
WebSphere Application Server	64
Testing WebLogic samples without EJB protection	65
Launching WebLogic Examples Server	65
To launch the WebLogic Examples Server	65
Running a Java client-EJB example	65
To gain access to a basic EJB from a Java client	65
Running a JSP-to-EJB example	66
To test JSP access to the EJB using WebLogic's built-in Web server	
67	
To test JSP access to the EJB using an external Web server.....	67
Testing WebLogic samples with EJB protection	69
Enabling Entrust security	69
To enable Entrust security for WebLogic Examples Server	69
Configuring EJB permissions	70
To configure EJB permissions using the Administration Console .	70
To configure EJB permissions by editing the EJB deployment	
descriptor.....	71
Gaining access to an EJB from a Java client	72
To gain access to a protected EJB from a Java client	72
Protecting an EJB with Entrust GetAccess roles	73
To protect a JSP using the Entrust GetAccess Runtime	73
Configuring a Web application's servlet filter	74
To configure a servlet filter using the WebLogic Administration	
Console	74
To configure a servlet filter by editing the deployment descriptor	75
To configure EJB permissions using the Administration Console .	76
To configure EJB permissions by editing the EJB deployment	
descriptor.....	77
Gaining access to an EJB from a browser through a JSP	78
To gain access to protected EJBs from a browser through a JSP .	78

Testing WebSphere samples without EJB protection	79
Running WebSphere Application Server samples	79
To test the Basic Calculator sample with a Java client.	79
To test the Basic Calculator sample with a Web browser	80
Testing WebSphere samples with EJB protection	81
Creating Entrust GetAccess users and resources	81
To set up Entrust GetAccess users for WebSphere Technology Samples.	81
To set up Entrust GetAccess resources for WebSphere Technology Samples.	82
Protecting EJBs using Application Server Runtimes	82
To configure EJB permissions	82
To add the Entrust GetAccess user security constraint to the Web application	83
To update the WebSphere Technology Samples.	84
Gaining access to a protected EJB from a Java client	85
To gain access to a protected EJB from a Java client.	85
Gaining access to an EJB through a JSP	87
To gain access to protected EJBs from a browser through a JSP	87

CHAPTER 6

Securing EJBs programmatically 89

EJB programmatic security model	90
Using getCallerPrincipal	90
Using isCallerInRole	91
Using servlet security methods	92

CHAPTER 7

Configuring Application Server Runtimes communications 95

Connecting to Entrust GetAccess	96
Configuring SSL	97
Preparing Application Server Runtimes to use SSL	97
To prepare Application Server Runtime for WebLogic to use server-authenticated SSL	97

To prepare Application Server Runtime for WebLogic to use client-authenticated SSL	98
To prepare Application Server Runtime for WebSphere to use server-authenticated SSL	100
To prepare Application Server Runtime for WebSphere to use client-authenticated SSL	101

CHAPTER 8

Uninstalling Application Server Runtimes105

Preparing to uninstall Application Server Runtimes	106
To prepare to uninstall Application Server Runtime for WebLogic.	106
To prepare to uninstall Application Server Runtime for WebSphere	107
Uninstalling on Windows	109
To uninstall on Windows	109
Uninstalling on Solaris	110
To uninstall on Solaris	110

CHAPTER 9

Troubleshooting Application Server Runtimes111

Overview	112
Troubleshooting installation problems	113
Removing remnants of a previous installation	113
To recover from a failed installation on Windows	113
To recover from a failed installation on Solaris — root user	113
To recover from a failed installation on Solaris — non root user	114
Redirecting temporary installation files	114
Error logging	115
Customizing the log file entries	115
Log file format	116
Log file header	116
Log file entries	116

Error messages	118
Installer error messages	118
Application Server Runtime error messages	120
Application Server Runtime for WebLogic error messages	120
Application Server Runtime for WebSphere error messages	121
createEntrustRealm script error messages	123
Web application server error messages	123

Chapter 1

About this document

This document is the Administration Guide for Entrust® Application Server Runtimes 7.0. It provides an overview of the product and instructions about how to install, configure, and use the Application Server Runtimes.

This chapter of the guide tells you where you can find Entrust information and documents related to the Application Server Runtimes, explains the document's typographical conventions, and describes how to obtain technical support.

Topics in this chapter:

- [“Related documents” on page 2](#)
- [“Typographic conventions” on page 3](#)
- [“Getting help” on page 4](#)

Related documents

Entrust Application Server Runtimes 7.0 depends upon Entrust Identification Server 7.0 running in an Entrust GetAccess™ 7.0 installation. This section lists, and briefly describes, the documentation for these products.

You can find the latest documentation on the Entrust Customer Support Web site at <https://www.entrust.com/support/documentation/index.cfm>.

Entrust Identification Server 7.0

Entrust Identification Server 7.0 and Entrust Entitlements Server 7.0 Administration Guide

This guide describes the Entrust Identification Server and Entrust Entitlements Server. It provides information about how to install the products and how to integrate them with Entrust GetAccess.

Entrust GetAccess 7.0

- *Entrust GetAccess 7.0 Product Guide*

This guide is an overview of Entrust GetAccess. It describes the product's features and components, and explains how they work together.

- *Entrust GetAccess 7.0 Planning and Installation Guide*

This guide shows the system architectures that Entrust GetAccess supports and explains how to deploy them. It also provides details about how to install and run the Entrust GetAccess server components and an Entrust GetAccess Runtime Service on a front-end computer. This guide is the hub of the Entrust GetAccess documentation set. Refer to it to find all the information related to installing and configuring your chosen architecture.

- *Entrust GetAccess 7.0 Business Administration Guide*

This guide describes the administrative tasks necessary to protect resources with Entrust GetAccess. The guide helps administrators and department managers to protect resources that are accessible to various groups of users within an organization.

- *Entrust GetAccess 7.0 System Administration Guide*

This guide describes how to configure and administer your Entrust GetAccess system. It also provides details about the installation of components that are not installed with the Entrust GetAccess server or Runtime Service installers.

- *Entrust GetAccess 7.0 Troubleshooting Guide*

This is the Entrust GetAccess troubleshooting guide. Refer to it if you encounter problems when you are installing or running Entrust GetAccess.

Typographic conventions

The following typographic conventions are used throughout this guide:

Convention	Purpose	Example
Bold text (other than headings)	Indicates graphical user interface elements and wizards	Click Next .
<i>Italicized text</i>	Used for book or document titles	<i>Entrust® Identification Server 7.0 Administration Guide</i>
Blue text	Used for hyperlinks to other sections in the document	Refer to “ Related documents ” on page 2.
<u>Underlined blue text</u>	Used for Web links	For more information, visit our Web site at www.entrust.com .
Courier type	Indicates installation paths, file names, Windows registry keys, commands, and text you must type	Locate and double-click the executable file called <code>AppServerRuntimes_setupwin32.exe</code> .
Angle brackets < >	Indicates variables (text you must replace with your organization's correct values)	Navigate to <code><install_path>\Tools\dvt</code> .
Square brackets []	Indicates optional parameters	<code>dsa password [-ldap]</code>

Note and Attention text

Throughout this guide you will see paragraphs that have ruled lines above and below the text. These paragraphs provide key information with two levels of importance, as shown below.

Note: Hints, tips, and information that must be emphasized to help you get the best from your software.



Attention: Issues that, if ignored, may seriously affect performance, security, or the operation of your software.

Getting help

Entrust recognizes the importance of providing quick and easy access to our support resources. The following subsections provide details about the technical support and professional services available to you.

Technical Support

Entrust offers a variety of technical support programs to help you keep Entrust Application Server Runtimes up and running. To learn more about the full range of Entrust technical support services, visit our Web site at:

<http://www.entrust.com/support/>

If you are registered in our support program, you can use our Web-based support services at:

<https://www.entrust.com/support/contact/index.htm>

The Customer Support Extranet Web site offers technical resources including online versions of Entrust product documentation, white papers and technical notes, and a comprehensive Knowledge Base.

Note: You must have an account to use the Customer Support Extranet Web site. To create an account, go to

https://www.entrust.com/members/user_create.htm

When you contact Entrust Customer Support, please provide as much of the following information as possible:

- Your contact information
- Product name, version, and operating system information
- Your deployment scenario
- Description of the problem
- Copy of log files containing error messages
- Description of conditions under which the error occurred
- Description of troubleshooting activities you have already performed

Telephone numbers

For telephone assistance between 8:00 AM and 8:00 PM Eastern Standard Time (EST), Monday to Friday, call one of the numbers below:

- 1-866-267-9297 in North America
- 1-613-270-2680 outside North America

E-mail address

The e-mail address for Customer Support is:

customer.support@entrust.com

Online

To submit a question online, go to the following Web address:

<https://www.entrust.com/support/supportinfo/index.htm>

Professional Services

The Entrust team assists e-businesses around the world to deploy and maintain secure transactions and communications with their partners, customers, suppliers and employees. We offer a full range of professional services to deploy our e-business solutions successfully for wired and wireless networks, including planning and design, installation, system integration, deployment support, and custom software development.

Whether you choose to operate your Entrust solution in-house or subscribe to hosted services, Entrust Professional Services will design and implement the right solution for your e-business needs. For more information about Entrust Professional Services please visit our Web site at:

<http://www.entrust.com>

Chapter 2

About Entrust Application Server Runtimes 7.0

This chapter provides an overview of Entrust Application Server Runtimes 7.0. The sections in this chapter briefly describe how the product integrates with BEA® WebLogic Server™, IBM® WebSphere® Application Server, and Entrust GetAccess.

Topics in this chapter:

- [“Overview” on page 8](#)
- [“Web application servers” on page 9](#)
- [“J2EE application environment” on page 10](#)
- [“EJB security” on page 12](#)
- [“Application Server Runtime for WebLogic” on page 15](#)
- [“Application Server Runtime for WebSphere” on page 21](#)

Overview

Entrust Application Server Runtimes 7.0 comprises two security extensions, one for each of the following Web application servers:

- BEA WebLogic Server 7.0
- IBM WebSphere 5.0 Application Server

These security extensions provide Entrust security for authenticating and authorizing access to Enterprise JavaBean™ (EJB) components and their methods in a distributed Java™ 2 Platform Enterprise Edition (J2EE) environment.

The Application Server Runtimes make use of Entrust GetAccess 7.0 as the underlying authentication and authorization security provider through WebLogic Server's Security Service Provider Interfaces (SSPIs) and WebSphere Application Server's User Registry, Trust Association Interceptor, and Authorization Table application programming interfaces (APIs).

An Entrust GetAccess add-on, Entrust Identification Server 7.0, allows the Application Server Runtimes to communicate with Entrust GetAccess. Entrust Identification Server provides Web services and application programming interfaces (APIs) for identification (authentication) services. You can think of the Application Server Runtime applications as clients of this server. Refer to the *Entrust Identification Server 7.0 and Entrust Entitlements Server 7.0 Administration Guide* for information about this product.

Note: The Application Server Runtimes perform authorization checks by comparing a user's roles with roles required for access to protect resources. The Entrust GetAccess Entitlements Service policy rules, therefore, do not apply.

The Entrust GetAccess Runtime software protects HTTP resources and the Application Server Runtimes extend and enhance Entrust GetAccess to protect EJBs by enforcing the security levels specified in the EJB deployment descriptor.

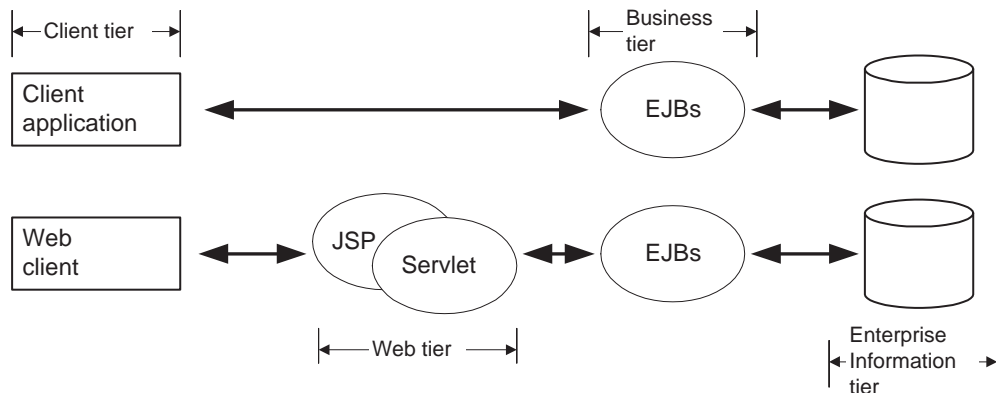
This document describes how to set up and configure the Application Server Runtimes to protect EJBs deployed on WebLogic Server and WebSphere Application Server as part of a J2EE distributed application.

Web application servers

A J2EE enterprise application is a distributed set of software components installed on separate computers in a multi-tiered system. The business tier of the application consists primarily of an application server, which provides three important functions within the enterprise application:

- Communication with the application and Web clients in the client tier
- Communication with back-end systems and databases in the data, or enterprise information, tier
- Management of EJBs, which provide the core business logic capabilities of the application, in an EJB container

Figure 1: Typical J2EE application



The Entrust Application Server Runtimes integrate the EJB container security model, which is responsible for enforcing security in the operational environment, with Entrust GetAccess. As the EJB security provider, Entrust GetAccess is the underlying authentication infrastructure for the EJB containers in WebLogic Server and in WebSphere Application Server.

J2EE application environment

An enterprise application is a complex and distributed set of related software programs that perform functions critical to the proper operation of an organization's business. The organization, in the case of an enterprise application, is usually an entire corporation, comprising all of its affiliates, branches, and divisions.

The J2EE from Sun™ Microsystems provides a development environment and multi-tiered application model for the design, assembly, and deployment of enterprise applications. Distributed J2EE applications divide application logic into components corresponding to their function. A J2EE environment comprises these components and the APIs, containers, roles, and protocols used to develop and manage J2EE enterprise applications.

Refer to the J2EE Web site (<http://java.sun.com/j2ee/>) for information about J2EE technologies and for the J2EE specification.

EJBs

EJBs are core components of the J2EE. They are software modules that form the heart of an enterprise application — it's business logic. EJBs provide the scalability required of a J2EE enterprise application and reside in an EJB container on the business tier. The EJB container provides and manages the runtime environment for EJBs, whose business logic is invoked at runtime either directly by a client application, or by way of a Web browser and Java Server Pages (JSPs) or servlets.

Refer to the Enterprise JavaBeans Technology Web site (<http://java.sun.com/products/ejb/>) for detailed information about EJBs and for the EJB specification.

Development roles

The component-based architecture of a J2EE application, and its reusable software modules, allow application development and deployment activities to be separated into roles that each describe a part of the development process. The important roles are:

- Enterprise Bean developer — who writes and compiles the code and deployment descriptor into an EJB jar file
- Application assembler — who collects the application component jar files, transaction, and security information, and assembles them to create an Enterprise Archive (EAR) file containing the J2EE application
- Application deployer — who configures the J2EE application and deploys it into the production environment

The J2EE and EJB specifications define other roles including those of EJB server provider and EJB container provider. In the context of the Entrust Application Server Runtimes, these two roles are filled by the WebSphere Application Server and WebLogic Server products, both of which incorporate J2EE servers and EJB containers.

EJB security

The EJB specification (<http://java.sun.com/products/ejb/docs.html>) discusses two models for implementing security in EJBs: a declarative model, and a programmatic model. The specification, however, does not dictate how to implement security and EJB server vendors are free to specify their own mechanisms for authenticating users and authorizing their access to EJBs.

Declarative security model

The EJB specification favors the declarative security model recommending that the EJB developer separates security code from business logic in an EJB. Enterprise application providers declare the security requirements of their applications in an XML deployment descriptor.

The application assembler, or the application deployer, configures EJB component security according to the deployment descriptor, when the application is deployed. Using container-specific tools, the application deployer maps security roles to EJB methods. Typically, the application deployer also associates roles with principals (identities assigned to a user, or group of users, during authentication) in the operational environment. With the Application Server Runtimes, Entrust GetAccess maintains this user-role mapping centrally. Each security role represents a specific type of user that shares the same access privileges as the EJB methods. At runtime, the EJB container enforces access control.

Programmatic security model

The EJB specification also describes a programmatic security model. EJBs that use this model determine the identity of users and their associated security roles programmatically using methods that allow access to the caller's security context (the container-provided runtime instance of an EJB). These methods are:

- `java.ejb.EJBContext.getCallerPrincipal()` — returns an instance of `java.security.Principal`
- `javax.ejb.EJBContext.isCallerInRole(String rolename)` — returns a `Boolean`

The programmatic security model supplements declarative security enabling security-aware applications to make decisions based on such variables as time-of-day, for example. Refer to "[Securing EJBs programmatically](#)" on page 89 for information about how to use programmatic security.

Security roles

EJB containers enforce security. In the context of the Application Server Runtimes, either WebLogic Server or WebSphere Application Server provides the EJB container.

EJB containers enforce access to EJBs by defining security roles according to the security defined in the deployment descriptor. The application assembler maps these roles to EJB methods when he or she creates the application's EAR file. The association between roles and methods, which is a many-to-many relationship — a role can be mapped to many methods and a method can be mapped to many roles — is called a *security view*. If one of the security roles defined in the security view is associated with a user, the user can execute the EJB method associated with that security role.

The XML deployment descriptor, which the application assembler includes in the application's EAR file, specifies security roles. The following XML fragment is an example of a deployment descriptor for an EJB called `BasicCalculator` with a protected method called `makeSum`. The fragment shows a typical role name in `role-category.role-id` format and shows how the `<method-permission>` and `<security-role>` elements are related.

```
<assembly-descriptor>
  <security-role>
    <role-name>calculator.sum</role-name>
  </security-role>
  <method-permission>
    <role-name>calculator.sum</role-name>
    <method>
      <ejb-name>BasicCalculator</ejb-name>
      <method-name>makeSum</method-name>
      <method-params>
        <method-param>double</method-param>
        <method-param>double</method-param>
      </method-params>
    </method>
  </method-permission>
  <container-transaction>
    <method>
      <ejb-name>BasicCalculator</ejb-name>
      <method-name>*</method-name>
    </method>
    <trans-attribute>Never</trans-attribute>
  </container-transaction>
</assembly-descriptor>
```

In this example, to execute the `makeSum` method, a user must have the role `sum`, which belongs to the role category `calculator`.

WebLogic Server and WebSphere Application Server permit or deny a user access to EJBs according to whether the user is associated with the relevant security roles. The application servers perform the following steps to make this determination:

- Obtain roles required to gain access to the requested resource or method (specified in the security constraints or method permission tables)
- Obtain roles associated with the user making the request
- Grant user access if there is a match between the resource or method roles and the roles associated with the user
- Throw a security exception if there is no match

Application Server Runtime for WebLogic

BEA WebLogic Server 7.0 implements the J2EE Platform providing a security framework within which application deployers can set security roles for EJB resources. The WebLogic security framework also includes the means for independent software vendors, such as Entrust, to create security providers.

Security providers are the software modules that deliver security services for WebLogic Server resources. In the case of the Entrust Application Server Runtime for WebLogic, the security provider is Entrust GetAccess, which uses WebLogic Server's Security Service Provider Interfaces (SSPI) to provide the authentication and role mapping services that protect WebLogic Server resources.

WebLogic Server domains

A domain is a WebLogic Server administrative unit comprising logically related resources managed as a single entity by a WebLogic Administration Server. One or more WebLogic Servers, server clusters, and applications can make up a single domain.

For detailed information about WebLogic Server domains, refer to the WebLogic Server 7.0 Administration Guide (<http://e-docs.bea.com/wls/docs70/adminguide>).

WebLogic Server security realms

The collection of operations and processes that protect WebLogic Server resources make up a WebLogic Server security realm. The security realm comprises security providers, users, groups, and security roles.

You can set up more than one security realm in a WebLogic Server domain, but only one realm can be the active, or default, realm. Until you configure another one, the default security realm in a WebLogic Server installation, containing WebLogic security providers, is called `myrealm`. WebLogic Server authenticates users defined in the active security realm before they can gain access to the resources in the realm.

For detailed information about WebLogic Server security realms, refer to the WebLogic document entitled WebLogic Server 7.0 Introduction to WebLogic Security (<http://e-docs.bea.com/wls/docs70/secintro/>).

Entrust security realm

In a WebLogic domain, the Entrust security realm comprises the following components:

- Default WebLogic Server authentication provider
- Entrust Application Server Runtime authentication provider (SSPI plug-in)
- Default WebLogic Server role mapping provider
- Entrust Application Server Runtime role mapping provider (SSPI plug-in)
- Default WebLogic Server authorization provider
- Default WebLogic Server adjudication provider
- Default WebLogic Server credential mapping provider
- Default WebLogic Server key store

The default WebLogic Server and the Entrust Application Server Runtime authentication providers are configured so that either provider can authenticate a user. For routine administrative tasks and resources, the WebLogic Server authentication provider handles authentication decisions, while the Application Server Runtime authentication provider protects the EJBs.

To use Entrust GetAccess security services to protect the resources in a WebLogic security realm, configure a new Entrust security realm and then set the Entrust security realm as the default.

For information about the other types of providers, such as the adjudication and credential mapping providers, in a WebLogic Server security realm, refer to “Types of security providers” in the WebLogic document entitled WebLogic Server 7.0 Introduction to WebLogic Security_ (<http://e-docs.bea.com/wls/docs70/secintro/>).

Authentication providers

WebLogic Server establishes trust by identifying and validating a user using an authentication provider. A security realm can have one or more authentication providers, each of which can perform a different type of authentication — username/password authentication, certificate-based authentication, and so on.

When a user requests access to protected resources from a Web browser by way of a Web server running an Entrust GetAccess runtime service, the runtime service coordinates the authentication task. In this case, the Entrust GetAccess runtime accepts any authentication type (Entrust GetAccess username/password, certificate-based, LDAP, SecureID, Windows, Entrust TruePass, and so on) and the authentication provider then validates the Entrust GetAccess session ID.

When a user requests access to protected resources directly, using an EJB client application to communicate with the EJB container for example, the Application Server Runtime performs the authentication operation. In this case, the Application Server Runtime accepts only the Entrust GetAccess username/password authentication type. If the user requesting access to an EJB is not an Entrust GetAccess user, the Application Server Runtime does nothing

and allows the default WebLogic Server authentication provider to process the authentication tasks.

Servlet filters

A servlet filter intercepts requests to a servlet or JSP in a Web application. The servlet filter object can modify request and response messages as they pass to and from a Web application's resources, and are therefore useful components of a user authentication mechanism.

The Application Server Runtime servlet filter is used when requests for access to protected resources originate in a Web browser client. The servlet filter intercepts such requests and processes them using Entrust security to authenticate the user before allowing access to EJB components and methods.

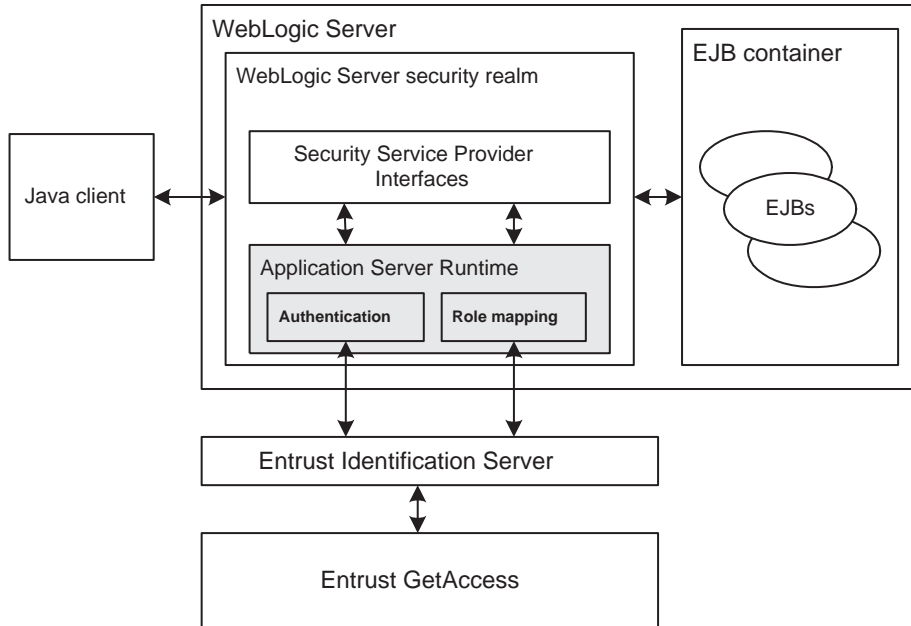
Role mapping providers

The Application Server Runtime role mapping provider returns Entrust GetAccess roles (associated with the user making the request for access to a protected resource) to the default WebLogic Server authorization provider, which performs all authorization tasks. The default WebLogic authorization provider uses this role information to determine whether or not to grant the user access to the resource.

Using a Java client to access protected resources

This subsection describes a simple client-server architecture in which a Java client requests direct access to a protected EJB.

Figure 2: Application Server Runtime for WebLogic in a Java client-server architecture



The client provides the user ID and password of an Entrust GetAccess user associated with roles giving the user access privileges to the EJB.

WebLogic Server forwards the user's request to the security framework, which calls the Application Server Runtime for WebLogic by way of the Security Services Provider Interface (SSPI). The Application Server Runtime software takes the user ID and password and authenticates the user through the Entrust Identification Server notifying the WebLogic SSPI of success by returning a session ID.

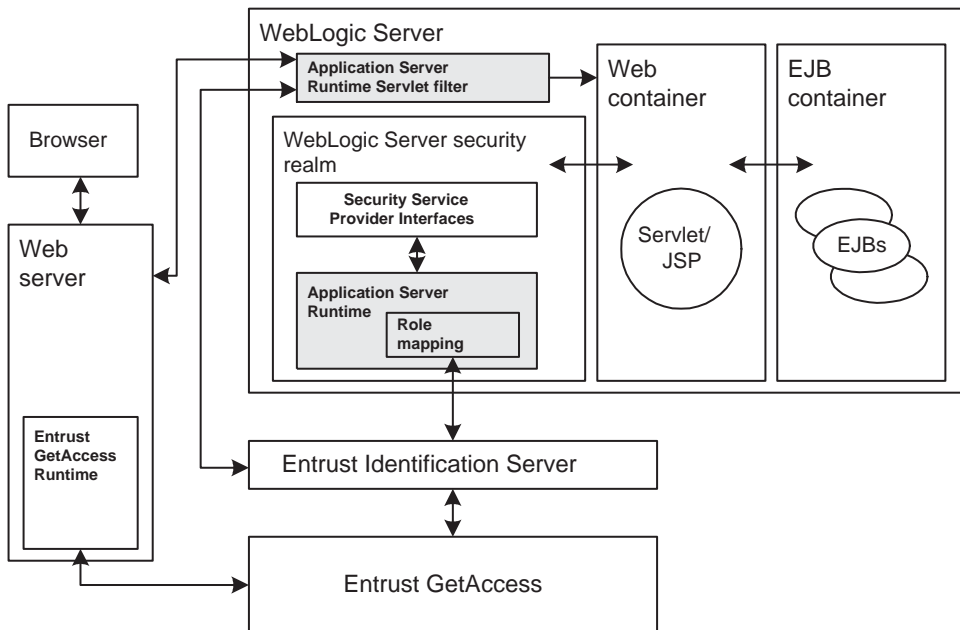
The WebLogic security framework then invokes the Application Server Runtime role mapping provider, which uses the session ID to obtain Entrust GetAccess roles from Entrust Identification Server and returns them to the WebLogic SSPI. If the session ID is not valid, the role mapping provider returns an empty list to the SSPI. Using the roles it receives from the Application Server Runtime, the

default WebLogic authorization provider determines whether or not to grant access to the EJB resources requested by the user.

Using a Web browser to access protected resources

This subsection describes a client-server architecture in which a user requests access to protected EJB resources indirectly, by way of a servlet or JSP, using a Web browser.

Figure 3: Application Server Runtime for WebLogic in a Web browser client-server architecture



In this case, the user must be authenticated by Entrust GetAccess to obtain a session ID before being granted access to the protected resources on WebLogic Server. One way to achieve this is to protect the JSP or servlet URL with Entrust GetAccess.

The user's browser sends an HTTP request to the servlet/JSP by way of the Web server. The Entrust GetAccess Runtime software on the Web server intercepts the request and, if the client has not already been authenticated, sends the request to Entrust GetAccess, which returns a login page. After the user logs in successfully, Entrust GetAccess sends a session ID as an HTTP cookie to the

browser, which then forwards the now authenticated Entrust GetAccess user's request to the servlet/JSP on WebLogic Server.

The Application Server Runtime servlet filter (specified in the Web application's deployment descriptor) intercepts the request to the servlet/JSP and extracts the Entrust GetAccess session ID to perform validation using Entrust Identification Server.

When the servlet/JSP tries to gain access to the protected EJBs, the WebLogic SSPI invokes the Application Server Runtime role mapping software, which extracts the session ID to obtain corresponding Entrust GetAccess roles. The role mapping software returns the roles to the WebLogic SSPI and the default WebLogic authorization provider grants or denies access to the protected resources.

Application Server Runtime for WebSphere

IBM WebSphere Application Server 5.0 provides a security infrastructure that protects Web and EJB resources in accordance with the J2EE specification. The architecture of WebSphere Application Server allows for the integration of third-party software components, such as Entrust Application Server Runtime for WebSphere, to provide security for the protection of resources.

WebSphere Application Server security model

WebSphere Application Server provides three APIs that allow third-party software vendors to extend and customize native WebSphere security:

- User Registry
- Trust Association Interceptor (TAI)
- Authorization Table (AT)

Refer to the WebSphere Application Server InfoCenter

(<http://publib7b.boulder.ibm.com/webapp/wasinfo1/index.jsp?deployment=ApplicationServer&lang=en>) for detailed information about the WebSphere security model.

User Registry

The user registry contains information about users and groups in a WebSphere Application Server environment. The WebSphere User Registry interface enables customers to implement their own custom user registry, which WebSphere Application Server then uses to determine who can, and who cannot, gain access to protected resources. The implementation of the API by Application Server Runtime for WebSphere that plugs into WebSphere Application Server when security is enabled is the `com.entrust.stp.asr.websphere.GaCustomRegistry` class.

Trust Association Interceptor

Setting up a trust relationship, or trust association, allows WebSphere Application Server to delegate the authentication of incoming requests to a third party. The Application Server Runtime for WebSphere class, `com.entrust.stp.asr.websphere.GaTrustAssociationInterceptor`, implements the Trust Association Interceptor interface.

Authorization Table

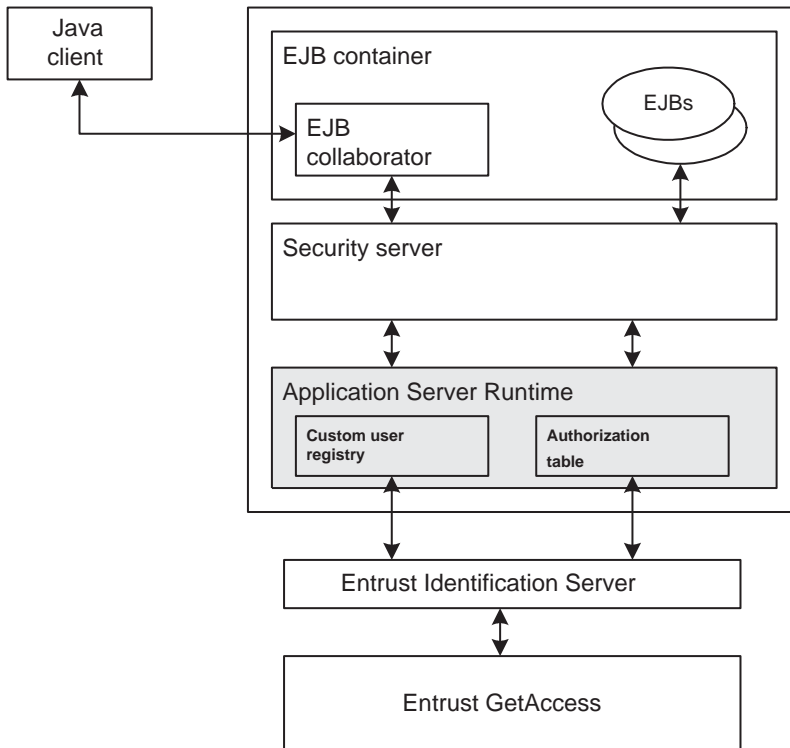
WebSphere Application Server uses an Authorization Table to determine user roles for gaining access to protected resources. The Application Server Runtime for WebSphere provides the

`com.entrust.stp.asr.websphere.GaAuthorizationTable` class for this purpose.

Using a Java client to access protected EJBs

This subsection describes a simple client-server architecture in which a Java client requests direct access to protected EJBs on a WebSphere Application Server. In this case, the EJB container in the WebSphere Application Server controls access to protected EJBs.

Figure 4: Application Server Runtime for WebSphere in a Java client-server architecture



The client requests access to protected EJBs on a WebSphere Application Server, providing the user ID and password of an Entrust GetAccess user. To authenticate the user, the WebSphere EJB Collaborator and Security Server

forward the user ID and password to Entrust Identification Server by way of the Application Server Runtime.

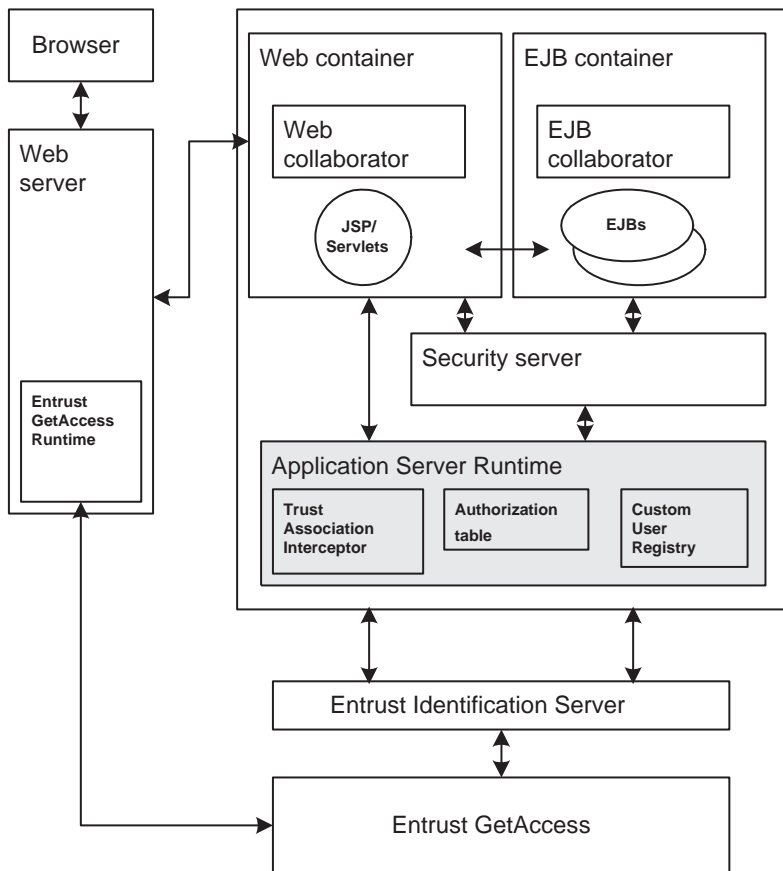
To obtain the Entrust GetAccess roles for the user, the WebSphere Security Server forwards the user ID to the Application Server Runtimes, where the Authorization Table determines, for the WebSphere Security Server, whether the user has the roles necessary to access to the EJBs.

Note: When Java clients access a protected EJB directly, the Application Server Runtime can look up only a user's static Entrust GetAccess roles, because the session is maintained by WebSphere Application Server.

Using a Web browser to access protected resources

This subsection describes a client-server architecture in which a user requests access to protected EJB resources indirectly, by way of a servlet or JSP, using a Web browser. In this case, the Web container in the WebSphere Application Server controls access to protected EJB resources.

Figure 5: Application Server Runtime for WebSphere in a Web browser client-server architecture



The user must be authenticated by Entrust GetAccess to obtain a session ID before being granted access to the protected resources on the WebSphere Application Server.

The user's browser sends an HTTP request to the servlet/JSP by way of the Web server. The Entrust GetAccess Runtime on the Web server intercepts the request

and, if the client has not already been authenticated, sends the request to Entrust GetAccess, which returns a login page. After the user logs in successfully, Entrust GetAccess sends a session ID as an HTTP cookie to the browser, which then forwards the now authenticated Entrust GetAccess user's request to the servlet/JSP on the WebSphere EJB container.

The Application Server Runtime Trust Association Interceptor intercepts the HTTP request and validates the session ID through Entrust Identification Server. WebSphere Security Server then contacts the Application Server Runtime to obtain Entrust GetAccess roles corresponding to the session ID and uses the role information to determine the user's access privileges.

Chapter 3

Installing Application Server Runtimes

This chapter describes how to prepare for, and install Entrust Application Server Runtimes.

Topics in this section:

- [“Preparing to install Application Server Runtimes” on page 28](#)
- [“Installing Application Server Runtimes” on page 30](#)

Preparing to install Application Server Runtimes

The Application Server Runtimes are security extensions for the BEA WebLogic Server and the IBM WebSphere Application Server. Refer to the Application Server Runtimes Readme document for specific information about the platform, operating systems, and Web application server versions that support the Application Server Runtimes.

Dependencies

To install and use the Application Server Runtimes, you must have network access to an Entrust GetAccess 7.0 installation that is running Entrust Identification Server 7.0. Refer to the section called [“Related documents” on page 2](#) for references to the relevant documentation.

Refer to your application server documentation for information about installing and configuring a Web application server on your computer.

The Application Server Runtime installer requires a Java Runtime Environment (JRE). The installer searches for a JRE in the default locations for Sun, IBM, and other well-known JREs. If the computer on which you are installing the Application Server Runtime software does not have a JRE, or if the JRE is not in a default location, you can invoke the installer at a command prompt using a command line option to tell the installer to use the JRE bundled with your Web application server. For example:

On a Windows operating system

- Using the WebSphere Application Server default location

```
AppServerRuntimes_setupwin32.exe -is:javahome "C:\Program Files\WebSphere\AppServer\java"
```

- Using the WebLogic Server default location

```
AppServerRuntimes_setupwin32.exe -is:javahome "C:\bea\jdk131_06"
```

On a Solaris operating system

- Using the WebSphere Application Server default location

```
AppServerRuntimes_setupsolarisSparc.bin -is:javahome "/opt/WebSphere/AppServer/java"
```

- Using the WebLogic Server default location

```
AppServerRuntimes_setupsolarisSparc.bin -is:javahome "/opt/bea/jdk131_06"
```

Downloading the installer application

When your organization orders the Application Server Runtimes, Entrust's Software Distribution department sends an email letter that contains instructions about how to obtain the installer software and the user ID and password you will need. Follow the instructions that explain how to use the user ID and password to gain access to the download page and download the installer executable file for your operating system to a convenient temporary location.

- For a Windows operating system, download `AppServerRuntimes_setupwin32.exe`
- For a Solaris operating system, download `AppServerRuntimes_setupsolarisSparc.bin`

Installing Application Server Runtimes

Having downloaded the appropriate installer application for your operating system (refer to ["Downloading the installer application" on page 29](#)), follow the steps in this section to install the Application Server Runtime software.

Installing on a Windows operating system

You can choose to install the Application Server Runtime software using either a graphical user interface (GUI) or from the command line.

To install using the GUI

- 1 Locate and double-click the executable file called `AppServerRuntimes_setupwin32.exe` in the temporary folder you used when downloading it.

The Welcome dialog box appears.

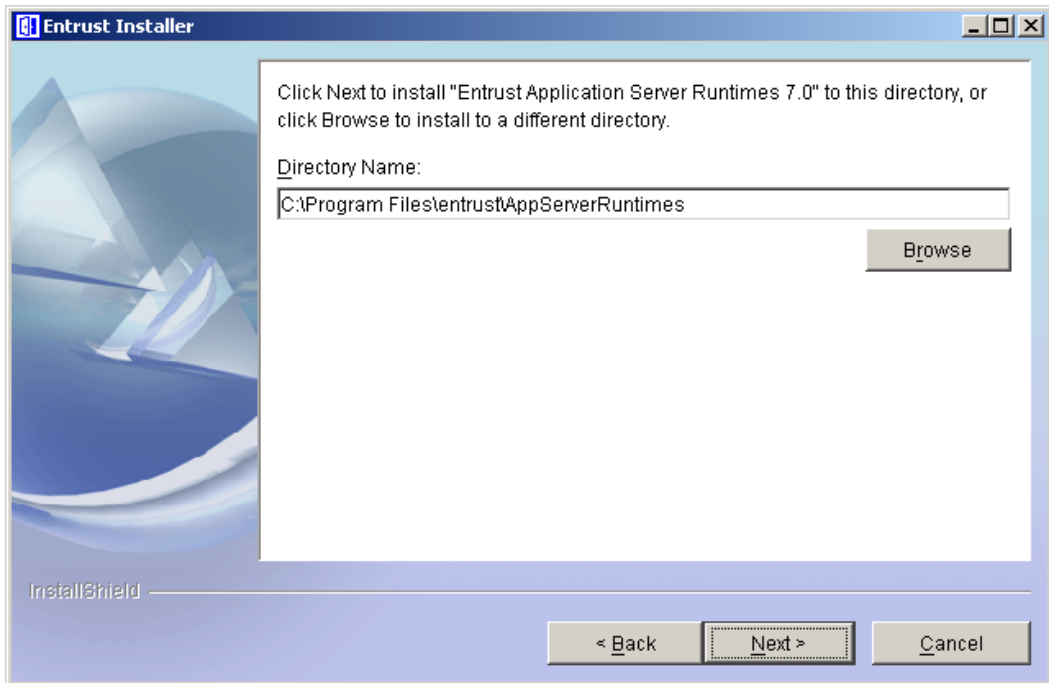
Click **Next**.

- 2 Read the license agreement carefully. If you accept the terms, select the appropriate radio button and click **Next** to continue with the installation.

The install location dialog box appears.

If you do not accept the terms of the licence agreement, select the appropriate radio button and click **Cancel** to terminate the installer application.

- 3 Click **Browse** to install the Application Server Runtime software in a directory other than the default directory shown in the dialog box, or click **Next** to accept the default location.



The server setup dialog box appears.

- 4 Select the protocol you wish to use to communicate with the Entrust GetAccess 7.0 Infrastructure — HTTP or HTTPS.

Type the host name and port number of the computer that the Application Server Runtime software will use to communicate with Entrust GetAccess 7.0 Infrastructure.

Click **Next** — the Web application server selection dialog box appears.

- 5 Select the Web Application Server you are running — either BEA WebLogic Server or IBM WebSphere Application Server.

Click **Next** — the Web server setup dialog box appears.

- 6 Type the path to, or click **Browse** to select, the location of your Web Application Server installation directory.

For WebLogic Server, the default installation directory is

c:\bea\weblogic700.

For WebSphere Application Server, the installer application detects the WebSphere Application Server installation directory.

Note: If you are installing the Application Server Runtime for use with the Deployment Manager in a WebSphere Application Server Network Deployment distributed environment, the installer application detects the installation directory of the Deployment Manager.

You must also specify a WebLogic domain for the Application Server Runtimes to configure. Refer to [“WebLogic Server domains” on page 15](#) for more information.

Type the path to, or click **Browse** to select, the WebLogic domain you want to configure, usually `c:\bea\user_projects\mydomain`.

Click **Next** — the installation details dialog box appears.

- 7 Read the Application Server Runtime installation details.

Click **Next** to proceed with the installation.

- 8 When the installation is complete, a successful completion dialog box appears.

Click **Finish** to close the installer application.

Refer to the chapter entitled [“Performing post-installation tasks” on page 37](#) for information and procedures that tell you how to setup and configure your Web Application Server and the Entrust Application Server Runtimes.

To install from the command line

- 1 Open a command prompt window.
- 2 Change to the directory that contains the Application Server Runtimes executable file (the installer application), `AppServerRuntimes_setupwin32.exe`.
- 3 Type `AppServerRuntimes_setupwin32.exe -console` and press **Enter**.
The installer application opens a new command prompt window and displays information and instructions as the installation progresses.
- 4 Follow the instructions in the command prompt window.

Refer to the chapter entitled [“Performing post-installation tasks” on page 37](#) for information and procedures that tell you how to setup and configure your Web Application Server and the Entrust Application Server Runtimes.

Installing on a Solaris operating system

You can choose to install the Application Server Runtime software using either a graphical user interface (GUI) or from the command line.

To install using the GUI

- 1 Locate the file called `AppServerRuntimes_setupsolarisSparc.bin` in the temporary directory you used when downloading it.
- 2 Use the `chmod` command to change the permissions of the `AppServerRuntimes_setupsolarisSparc.bin` file to make it executable.

```
chmod u+x AppServerRuntimes_setupsolarisSparc.bin
```

- 3 Run the executable file.
The Welcome dialog box appears.

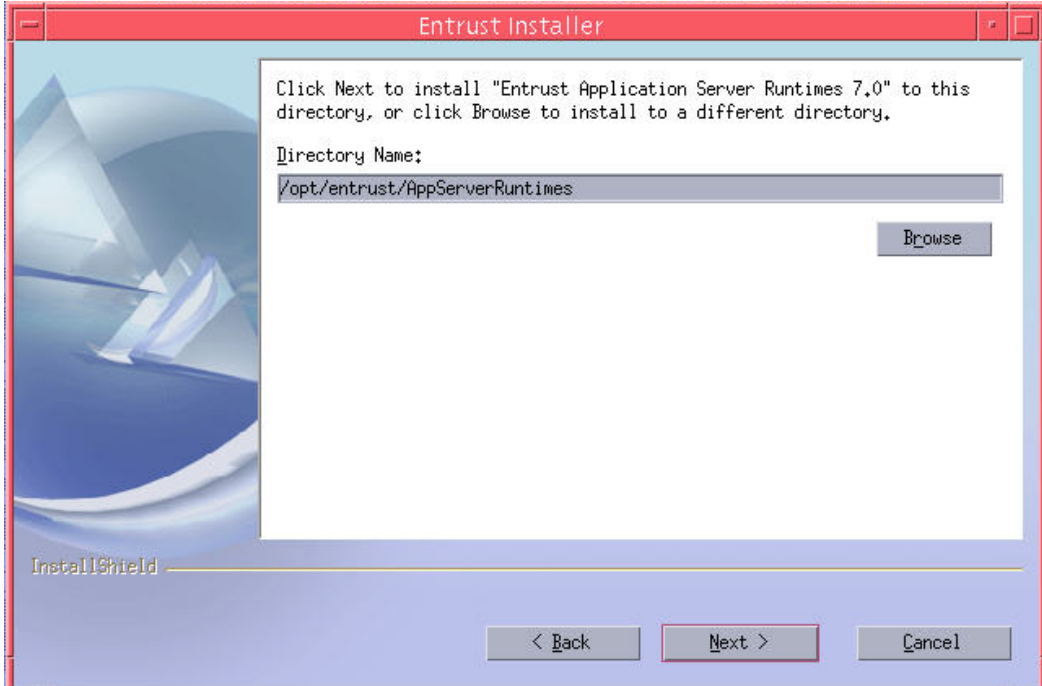
Click **Next**.

- 4 Read the license agreement carefully. If you accept the terms, select the appropriate radio button and click **Next** to continue with the installation.

The install location dialog box appears.

If you do not accept the terms of the licence agreement, select the appropriate radio button and click **Cancel** to terminate the installer application.

- 5 Click **Browse** to install the Application Server Runtime software in a directory other than the default directory shown in the dialog box, or click **Next** to accept the default location.



If you install Application Server Runtimes in a directory other than the default, make sure the directory name does not contain a space.

The server setup dialog box appears.

- 6 Select the protocol you wish to use to communicate with the Entrust GetAccess 7.0 Infrastructure — HTTP or HTTPS.

Type the host name and port number of the computer that the Application Server Runtime software will use to communicate with Entrust GetAccess 7.0 Infrastructure.

Click **Next** — the Web server selection dialog box appears.

- 7 Select the Web Application Server you are running — either BEA WebLogic Server or IBM WebSphere Application Server.

Click **Next** — the Web server setup dialog box appears.

- 8 Type the path to, or click **Browse** to select, the location of your Web Application Server installation directory.

For WebLogic Server, the default installation directory is
`/opt/bea/weblogic700.`

For WebSphere Application Server, the default installation directory is
`/opt/WebSphere/AppServer.`

Note: If you are installing the Application Server Runtime for use with the Deployment Manager in a WebSphere Application Server Network Deployment distributed environment, click **Browse** to select the path to the Deployment Manager. For example, the default path is:

`/opt/WebSphere/DeploymentManager`

You must also specify a WebLogic domain for the Application Server Runtimes to configure. Refer to [“WebLogic Server domains” on page 15](#) for more information.

Type the path to, or click **Browse** to select, the WebLogic domain you want to configure, usually `/opt/bea/user_projects/mydomain.`

Click **Next** — the installation details dialog box appears.

- 9 Read the Application Server Runtime installation details.

Click **Next** to proceed with the installation.

- 10 When the installation is complete a successful completion dialog box appears.

Click **Finish** to close the installer application.

Refer to the section entitled [“Performing post-installation tasks” on page 37](#) for information and procedures that tell you how to setup and configure your Web Application Server and the Entrust Application Server Runtimes.

To run a command line installation

- 1 Change to the directory that contains the Application Server Runtimes installer, `AppServerRuntimes_setupsolarisSparc.bin`.

This should be the temporary folder you used when downloading the installer.

Use the `chmod` command to change the permissions of the file to make it executable.

```
chmod u+x AppServerRuntimes_setupsolarisSparc.bin
```

- 2 Type `AppServerRuntimes_setupsolarisSparc.bin -console` and press **Enter**.
- 3 Follow the installer application's instructions to complete the installation.
Refer to the section entitled "[Performing post-installation tasks](#)" on page 37 for information and procedures that tell you how to setup and configure your Web Application Server and the Entrust Application Server Runtimes.

Performing post-installation tasks

Once you have installed the Application Server Runtime software, there are a number of post-installation procedures you must perform. These steps ensure that your Entrust GetAccess installation and Web application server are configured correctly to operate with the Application Server Runtime software.

Refer to the chapter entitled "[Performing post-installation tasks](#)" on page 37 for more information.

Chapter 4

Performing post-installation tasks

This chapter describes how to perform post-installation tasks that set up and configure the Application Server Runtimes for your Web application server.

Topics in this chapter:

- [“Overview” on page 38](#)
- [“Post installation steps — WebLogic Server” on page 39](#)
- [“Post installation steps — WebSphere Application Server” on page 44](#)
- [“Configuration” on page 55](#)

Overview

This chapter contains information about the post installation tasks you should perform to set up the Application Server Runtime software for the Web application server you chose during installation — WebLogic Server or WebSphere Application Server.

The Application Server Runtime installation directories contain script files that carry out many of the configuration steps automatically. This chapter describes how to use the script files in combination with necessary manual steps in the WebLogic Administration Console and the WebSphere Administrative Console.

Post installation steps — WebLogic Server

If you have installed the Application Server Runtime software for BEA WebLogic Server, follow the procedures in this section to perform post installation tasks that configure the Application Server Runtime software for a WebLogic domain. The section called “[Application Server Runtime for WebLogic](#)” on page 15 contains background information on these tasks.

Note: Refer to the BEA document called Overview of WebLogic System Administration (<http://e-docs.bea.com/wls/docs70/adminguide/overview.html#1034986>) for definitions and explanations of terms used in WebLogic server administration.

Creating an Entrust security realm

Refer to the sections entitled “[WebLogic Server security realms](#)” on page 15 and “[Entrust security realm](#)” on page 15 for information about the WebLogic security realm and the Entrust security realm. The following procedure describes how to run the `createEntrustRealm` script. The script creates an Entrust security realm in the active domain. Repeat the procedure for other Weblogic domains as required.

To create an Entrust realm using the `createEntrustRealm` script

- 1 In the WebLogic domain you are working with, ensure the WebLogic Administration Server is running.

Note: If a WebLogic domain contains just one instance of WebLogic Server, it is the Administration Server for that domain.

If you have just completed the Application Server Runtime installation, restart the Administration Server.

- 2 Locate the `createEntrustRealm` script file in your Application Server Runtimes installation directory.

```
<App Server Runtimes root>\bin\weblogic\createEntrustRealm.cmd
```

```
<App Server Runtimes root>/bin/weblogic/createEntrustRealm.sh
```

- 3 Open the script file in a text editor and amend or confirm the values for `WEBLOGIC_HOME` and `ADMIN_URL`.

Save the file if you changed these values.

Note: You will not need to make changes to the script file if you are using WebLogic Server defaults.

Do not change `t3://`. It is a special protocol used for WebLogic Server administration.

The script file assumes that the Java executable is specified in your computer's `PATH` environment variable. If this is not the case, edit the script file to specify the path to the Java executable. For example, change

```
java -classpath wlsasrinst.jar
```

to

```
c:\bea\jdk131_06\jre\bin\java -classpath wlsasrinst.jar
```

- 4 At a command prompt, change to the directory containing the `createEntrustRealm` script file and run the script.

Type `createEntrustRealm.cmd` on a Windows operating system or `createEntrustRealm.sh` on a Solaris operating system.

- 5 In response to the prompts, type the administrator user name and password for your WebLogic domain.

```
WebLogic home: c:\bea\weblogic700
Domain administration server listen address:
t3://localhost:7001
Enter username for domain admin: ASRforWLS
Enter password for domain admin: *****
```

```
This will install the Entrust Realm within your WebLogic
domain.
```

```
This action may take a few minutes. Are you sure you want to
continue (Y/N)?
```

- 6 Type `Y` to create the Entrust security realm.

The script creates the Entrust realm and displays the following output:

```
Creating EntrustRealm...
Configuring AuthenticationProviders.....
Configuring Authorizers.....
Configuring the Adjudicator.....
Configuring RoleMappers.....
Configuring CredentialMappers.....
Configuring KeyStores.....
```

Done!

You have created an Entrust security realm in the active domain.

Configuring a domain to use the Entrust Realm

Once you have created an Entrust security realm, you must configure the WebLogic domain to use the security provided by the Application Server Runtime. The WebLogic domain must recognize the Entrust realm as the default security realm. To do this, follow the procedure in this section.

To configure an Entrust realm in a WebLogic domain

- 1 Open the WebLogic Administration console.
- 2 In the left pane, click the name of the WebLogic domain you want to configure. The name of the domain (`mydomain`, for example) is usually the root node of the tree.
- 3 In the right pane, click the **Security** tab.
- 4 Under the **Security** tab, in the **Default Realm** drop-down list, select **EntrustRealm**, then click **Apply**.
- 5 Restart the WebLogic server.

Setting up Application Server Runtime software in other WebLogic domains

When you installed the Application Server Runtime for WebLogic Server, you selected a WebLogic domain for the installer to configure. The installer application modified the startup script for the Administration Server in that domain by inserting code that sets the classpath, Java options, and the installation directory of the Application Server Runtime software.

Follow the procedure in this subsection to set up the Application Server Runtime in a WebLogic domain other than the domain you selected when you installed the Application Server Runtime software.

To set up Application Server Runtime software in additional WebLogic domains

- 1 Locate the startup script (`startWebLogic.cmd` on Windows, and `startWebLogic.sh` on Solaris) for the Administration Server of the domain modified by the Application Server Runtime installer application.

- 2 Open the Administration Server startup script in a text editor.
- 3 Locate and copy the following lines of code.

```
@rem ASR *****
@rem start config line for Application Server Runtimes
set ENTRUST_ASR=<path to App Server Runtimes installation>
set CLASSPATH=<classpath> ...
set JAVA_OPTIONS=<Java options> ...
@rem end config line for Entrust Application Server Runtimes
@rem ASR *****
```

- 4 Locate the startup script for WebLogic Server you want to modify.
If a WebLogic domain contains just one instance of WebLogic Server, it is the Administration Server for that domain.
- 5 Open the startup script in a text editor and locate the line containing
`@rem Call Weblogic Server.`
- 6 Paste the lines of code you copied in [Step 3](#) before the line located in [Step 5](#) (just before the call to start the WebLogic Server).
- 7 Save the startup scripts you have modified.

If you uninstall the Application Server Runtime, you must edit the startup scripts to remove the lines of code you inserted by following this procedure.

Setting up the Application Server Runtime in advanced WebLogic Server configurations

In advanced WebLogic Server configurations, such as a WebLogic domain with multiple WebLogic server instances on several computers, you must perform further configuration tasks to protect your resources with the Application Server Runtime for WebLogic Server. This subsection describes these additional steps.

When you installed the Application Server Runtime for WebLogic Server, you selected a WebLogic domain for the installer to configure. The installer application modified the startup script for the Administration Server in that domain by inserting code that sets the classpath, options, and the installation directory of the Application Server Runtime.

You must now modify the startup scripts for all instances of WebLogic server (managed servers) in the domain whose EJB resources you want to protect with the Application Server Runtime.

On the computer that hosts the WebLogic domain Administration Server

- 1 Install the Application Server Runtime for WebLogic Server.
Refer to [“Installing Application Server Runtimes”](#) on page 30.

- 2 Run the script to create an Entrust security realm, `createEntrustRealm.cmd` or `createEntrustRealm.sh`.
Refer to [“Creating an Entrust security realm” on page 39](#).

On computers that host the WebLogic domain Managed Servers

- 1 Install the Application Server Runtime for WebLogic Server.
Refer to [“Installing Application Server Runtimes” on page 30](#).
- 2 Follow the procedure called [“To set up Application Server Runtime software in additional WebLogic domains” on page 41](#).

Note: The startup script referred to in steps 4 to 7 of this procedure are those that start each Managed Server you want to modify (`startManagedWebLogic.cmd` on Windows, and `startManagedWebLogic.sh` on Solaris).

- 3 Restart the WebLogic server.

Post installation steps — WebSphere Application Server

If you have installed the Application Server Runtime software for IBM WebSphere Application Server, follow the procedures in this section to perform post-installation steps.

Once you have configured WebSphere Application Server to use Entrust security and Global Security has been turned on, the WebSphere Administrative Console will recognize only Entrust GetAccess users associated with one of the following roles:

- WebSphere Administrator
- WebSphere Operator
- WebSphere Configurator
- WebSphere Monitor

Assign one of these roles to Entrust GetAccess users who need to use the WebSphere Application Server Administrative console. You can do this after you have set up the appropriate Entrust GetAccess roles and users as described in the procedure entitled “[To set up Entrust GetAccess users and roles](#)” on page 45.

For detailed instructions, procedures, and information about WebSphere Application Server, refer to the product's online documentation (<http://www-3.ibm.com/software/webservers/appserv/infocenter.html>).

Note: Before performing the procedures in this section, backup your WebSphere Application Server configuration files using the script in the bin directory of your WebSphere installation. The script file is called `backupConfig` and its default locations are

`C:\Program Files\WebSphere\AppServer\bin\backupConfig.bat` on Windows, and `/opt/WebSphere/AppServer/bin/backupConfig.sh` on Solaris.

Setting up Entrust GetAccess roles and users

To prepare WebSphere Application Server to take advantage of the security services provided by Entrust GetAccess by way of the Application Server Runtime, set up Entrust GetAccess users and roles to correspond to WebSphere security roles.

To set up Entrust GetAccess users and roles

- 1 Create an Entrust GetAccess role category with the following role category information:
 - ID — webspHERE
 - Name — webspHERE
 - Description — webspHERE

Refer to the *Entrust GetAccess 7.0 Business Administration Guide* for information about how to perform this task.

- 2 Create the Entrust GetAccess roles listed in [Table 1](#).

Table 1: Entrust GetAccess roles

Role Category	Role ID	Role Name	Description
webspHERE	administrator	WebSphere Administrator	WebSphere Administrator
webspHERE	operator	WebSphere Operator	WebSphere Operator
webspHERE	configurator	WebSphere Configurator	WebSphere Configurator
webspHERE	monitor	WebSphere Monitor	WebSphere Monitor

Refer to the *Entrust GetAccess 7.0 Business Administration Guide* for information about how to perform this task.

The role IDs in the table correspond to WebSphere Application Server administrative roles with the same names.

Depending on your WebSphere administrative needs, the operator, configurator, and monitor roles are optional.

To use a role category ID other than webspHERE, you must change the default configuration setting in the `configuration.global.xml` file. Refer to the section called ["Configuration" on page 55](#) for details about working with configuration settings.

- 3 Create a user called `wasadmin` and assign the user the WebSphere Administrator role.

Refer to the *Entrust GetAccess 7.0 Business Administration Guide* for information about how to perform this task.
- 4 Create an Entrust GetAccess user with the following attributes:
 - First Name — UserRegistry

- Last Name — Websphere
- Login — wasuserreg

The user does not require Entrust GetAccess roles.

The values you choose for these attributes are not important, but you must remember them so that you can use them when you create a custom user registry.

Refer to the *Entrust GetAccess 7.0 Business Administration Guide* for information about how to perform this task.

Configuring WebSphere Application Server security

You can configure WebSphere Application Server security semi-automatically using script files and the WebSphere Administrative Console. This section describes how to use script files and the browser-based Administrative Console to enable the use of Entrust security by WebSphere Application Server.

To configure an Entrust security realm

- 1 Locate the script called `configureEntrustRealm` in your Application Server Runtimes installation directory.

On Windows

```
<App Server Runtimes root>\bin\websphere\configureEntrustRealm.cmd
```

On Solaris

```
<App Server Runtimes root>/bin/websphere/configureEntrustRealm.sh
```

- 2 At a command prompt, change to the directory containing the script file and type `configureEntrustRealm.cmd` on Windows, or `configureEntrustRealm.sh` on Solaris.
- 3 Respond to the prompts as the script executes.
- 4 When the script displays the prompt `Enter GetAccess userid to add as WebSphere Console User`, type the userID of the WebSphere Administrator (`wasadmin`) you created in [Step 3](#) of the procedure called ["To set up Entrust GetAccess users and roles"](#) on page 45.
- 5 Type `a` to select administrator.

This user (`wasadmin`) will now be able to run all WebSphere command line tools (such as `wsadmin`) and scripts (such as `stopServer`).

You should see script prompts and responses similar to the following sample output when the script successfully configures an Entrust security realm:


```

C:\Program
Files\Entrust\AppServerRuntimes\bin\websphere>configureEntrustRealm

WAS_HOME=C:\Program Files\WebSphere\AppServer
ASR_HOME=C:\Program Files\entrust\AppServerRuntimes

Found cell: <host name>
The following files MAY be modified:
Found: C:\Program Files\WebSphere\AppServer\config\cells\<host
name>\security.xml
Found: C:\Program Files\WebSphere\AppServer\config\cells\<host
name>\admin-authz.xml

Continue and modify these configuration files (y/n) [y]: y

Configure Entrust Custom Registry... done
Set Active User Registry to Entrust... done
Set Active Authentication Mechanism to LPTA... done
Configure Entrust Trust Association Interceptor... done
Enable Trust Association... done
Configure Entrust Authorization Table... done

Enter GetAccess userid to add as WebSphere Console User: wasadmin
Select role:
  (a)administrator
  (o)perator
  (c)onfigurator
  (m)onitor
Enter a, o, c, m or (q)uit: a
Add "wasadmin" as administrator... done

Updated: C:\Program Files\WebSphere\AppServer\config\cells\<host
name>\security.xml
Updated: C:\Program Files\WebSphere\AppServer\config\cells\<host
name>\admin-authz.xml

** Operations completed successfully. **
The Entrust Application Server Runtime is configured.
You may now enable global security in WebSphere.
Press any key to continue . . .

```

- 6** Restart the WebSphere Application Server and open the **Administrative Console**.
- 7** In the left navigation pane, click **Security > User Registries > Custom**.
The **Custom User Registry** page opens.
- 8** Under the **Configuration** tab, **General Properties**, type a valid Entrust GetAccess user name in the **Server User ID** field.

Note: Type the name of the user (`wasuserreg`) you created in [Step 4](#) in the procedure entitled [“To set up Entrust GetAccess users and roles”](#) on page 45.

The Server User ID field specifies the user name under which the server runs for security purposes.

- 9** In the **Server User Password** field, type the user password corresponding to the Server User ID.
- 10** Click **Apply**.
- 11** Follow the entire procedure called [“To configure Lightweight Third Party Authentication”](#) on page 48 to set the password for Lightweight Third-Party Authentication (LTPA).
- 12** Follow the entire procedure called [“To configure Global Security settings for WebSphere Application Server”](#) on page 48 to enable Global Security.
- 13** Restart WebSphere Application server to allow your changes to take effect.

You have configured an Entrust security realm. WebSphere Application Server now uses the Application Server Runtime software for authentication and authorization.

To configure Lightweight Third Party Authentication

- 1** Start the WebSphere Application Server and open the **Administrative Console**.
- 2** In the left navigation pane, click **Security > Authentication Mechanisms > LTPA**.
The **LTPA** page opens.
- 3** Under the **Configuration** tab, **General Properties**, type a password and confirm it in the **Password** and **Confirm Password** fields.
WebSphere Application Server uses this password to encrypt and decrypt the LTPA keys when they are imported and exported. Remember the password — you will need to use it when you export the LTPA keys to another cell (a logical group of one or more nodes in a WebSphere Application Server distributed network).
- 4** Click **Apply** to complete the LTPA configuration.
- 5** In the **Administrative Console** menu, click **Save** to complete the procedure.

To configure Global Security settings for WebSphere Application Server

- 1** Start the WebSphere Application Server and open the **Administrative Console**.
- 2** In the left navigation pane, click **Security > Global Security**.
The **Global Security** page opens.
- 3** Under the **Configuration** tab, **General Properties**, select the **Enabled** checkbox.

Note: Java 2 Security is disabled by default in WebSphere Application Server, but it is automatically enabled when you turn on Global Security in the Administrative Console. Clear the **Enforce Java 2 Security** checkbox if you do not want to enable Java 2 Security.

Refer to the WebSphere Application Server InfoCenter (<http://publib7b.boulder.ibm.com/webapp/wasinfo1/index.jsp?deployment=ApplicationServer&lang=en>) and in the left pane click **Security > Managing Security > Configuring Java 2 Security** for information about securing your applications when Java 2 Security is enforced.

4 Click **OK**.

Note: When you turn on Global Security, the WebSphere Administrative Console displays a warning saying that “The domain name for Single Sign On is not defined.”

You can safely ignore this warning because you will be using the Entrust GetAccess single sign-on mechanism. If you want to disable the warning, refer to the WebSphere InfoCenter documentation (<http://www-3.ibm.com/software/webservers/appserv/infocenter.html>) for Single Sign-on settings and in the left pane click **Security > Securing applications and their environment > Managing security > Configuring authentication mechanisms > Single Sign-on settings**.

5 Restart the server for changes to the security settings to take effect.

When you log in, after you have complete this step and restarted the server, WebSphere Application Server checks your user ID and password against the custom user registry configured in the procedure “[To configure an Entrust security realm](#)” on page 46. Validation will fail if the user ID or password you use to log in are incorrect, or if the custom user registry has not been set up correctly.

To stop WebSphere Application Server when security is enabled

You cannot stop WebSphere Application Server using **Start > Programs > IBM WebSphere > Application Server v5.0 > Stop the Server**. This action invokes the `stopServer.bat` script without using a user ID and password. To stop the server you must run `stopServer.bat` from a command line prompt. For example:

```
stopServer server1 -username wasadmin -password <your_password>
```

The minimum privileges that allow a user to stop WebSphere Application Server are those associated with the WebSphere Command Console role.

Note: If you performed a default WebSphere Application Server installation, your error log will contain exceptions after you turn on Entrust security. Refer to the section entitled [“Disabling WebSphere sample applications” on page 53](#) for information about how to prevent the exceptions.

Command console users

Command console users have the authority to administer WebSphere Application Server using either the Administrative Console or command line tools (such as `wsadmin`) and scripts (such as `stopServer`). The `wsadmin` tool has the capability to execute scripts using the Bean Scripting Framework (BSF) and you can use it to configure and control your WebSphere Application Server installation.

When you run the `createEntrustRealm` script it prompts you to add a command console user. You can add additional command console users in the Administrative Console (if you are already a command console administrator with the `websphere.administrator` role) or using the `addConsoleUser` script.

Add additional users after you have turned on Global Security and after you have restarted the server. Select one of the following WebSphere console user administrative roles for the console users you are adding:

- `monitor`
This is the least privileged role and allows the user to view product configuration and current state.
- `configurator`
This user has the same privileges as a user with the `monitor` role and the right to change the product configuration.
- `operator`
This user has the same privileges as a user with the `monitor` role and the right to restart services.
- `administrator`
Users with the `administrator` role have the same privileges as users with the `configurator` and `operator` roles and the right to access sensitive configuration data including server passwords, LTPA passwords and keys, and so on.

These WebSphere console user administrative roles are not compared with Entrust GetAccess roles when scripts, such as `stopserver`, execute.

To add a console user with the addConsoleUser script

- 1 Locate the script called `addConsoleUser` in your Application Server Runtimes installation directory.

On Windows

```
<App Server Runtimes root>\bin\websphere\addConsoleUser.bat
```

On Solaris

```
<App Server Runtimes root>/bin/websphere/addConsoleUser.sh
```

- 2 Run the script and respond to its prompts.

You should see script prompts and responses similar to the following sample output:

```
C:\Program Files\Entrust\AppServerRuntimes\bin\websphere>addConsoleUser

WAS_HOME=C:\Program Files\WebSphere\AppServer
ASR_HOME=C:\Program Files\entrust\AppServerRuntimes

Found cell: jsmith
The following files MAY be modified:
Found: C:\Program Files\WebSphere\AppServer\config\cells\jsmith\security.xml
Found: C:\Program Files\WebSphere\AppServer\config\cells\jsmith\admin-authz.xml

Continue and modify these configuration files (y/n) [y]: y

Enter GetAccess userid to add as WebSphere Console User: user1
Select role:
  (a) dministrator
  (o) perator
  (c) onfigurator
  (m) onitor
Enter a, o, c, m or (q)uit: a
Add "user1" as administrator... done

Updated: C:\Program
Files\WebSphere\AppServer\config\cells\jsmith\admin-authz.xml

** Operations completed successfully. **
Press any key to continue . . .
```

- 3 Add additional command console users as required.

Note: Additional command console users will not be recognized until you restart WebSphere Application Server.

Setting up the Application Server Runtime in advanced WebSphere configurations

This subsection describes how to set up the Application Server Runtime for WebSphere if you are using WebSphere Application Server Network Deployment in advanced distributed configurations.

The WebSphere Application Server concepts of cells and nodes is the same in both the base and Network Deployment versions of the product. Nodes are groups of managed server and cells are groups of one or more nodes in a distributed network. In WebSphere Application Server base configuration, the correspondence between cells and nodes is one-to-one. A node is typically a single computer, although you can have more than one server in a node.

When you are using WebSphere Application Server Network Deployment, cells can have more than one node by federating, or incorporating, a node into a cell. Refer to the WebSphere InfoCenter for Network Deployment (<http://publib7b.boulder.ibm.com/webapp/wasinfo1/index.jsp?deployment=NetworkDeployment&lang=en>) for more information.

If you install WebSphere Application Server Network Deployment as a stand-alone application server in its own cell, the installation and configuration of the Application Server Runtime does not differ from that described in earlier chapters and sections of this guide. If you have installed WebSphere Application Server Network Deployment to manage more than one node federated under one cell, you must follow the steps described in this subsection.

On the computer that hosts the WebSphere Deployment Manager

- 1** Install the Application Server Runtime for WebSphere.
Refer to the installation instructions, “[Installing Application Server Runtimes](#)” on [page 30](#).
- 2** Run the script to create an Entrust security realm,
`configureEntrustRealm.cmd` or `configureEntrustRealm.sh`.
Refer to the procedure called “[To configure an Entrust security realm](#)” on [page 46](#).
- 3** Restart the Deployment manager.

Note: A Deployment Manager is a WebSphere administrative agent that centralizes the management of all cells in a node. A Deployment Manager is a discrete component, separate from the application servers, that is available when you install WebSphere Application Server Network Deployment.

Refer to the WebSphere InfoCenter for Network Deployment (<http://publib7b.boulder.ibm.com/webapp/wasinfo1/index.jsp?deployment=NetworkDeployment&lang=en>) and in the left pane click **System Administration > Administrative agents > Managing administrative agents > Deployment Managers**.

The Deployment Manager propagates the security configuration information into federated nodes when it restarts and when the nodes restart.

On computers that host the WebSphere Application Server Network Deployment nodes

- 1 Install the Application Server Runtime for WebSphere.
Refer to the installation instructions, “[Installing Application Server Runtimes](#)” on page 30.
- 2 Restart the WebSphere node and servers running on that node.

Note: You can also configure security manually using the Deployment Manager Administrative Console. Refer to the WebSphere InfoCenter for Network Deployment (<http://publib7b.boulder.ibm.com/webapp/wasinfo1/index.jsp?deployment=NetworkDeployment&lang=en>) and in the left pane click **System Administration > Administrative agents > Managing administrative agents > Managing nodes**.

Disabling WebSphere sample applications

This subsection describes how you can disable the WebSphere Application Server sample applications. Some of these applications do not support Global Security and generate exceptions in the WebSphere Application Server `SystemOut.log` file when you turn on Global Security.

The sample applications demonstrate various aspects of the application server, but are not all suitable for testing security. You can use the Technology Samples to verify your Application Server Runtime for WebSphere installation as described in the chapter called “[Verifying installation](#)” on page 63.

For more details, refer to the WebSphere Application Server online documentation at the following URLs:

http://publib7b.boulder.ibm.com/wasinfo1/en/info/aes/ae/covr_samples.html
http://publib7b.boulder.ibm.com/wasinfo1/en/info/aes/ae/rtrb_secprobs.html#msgs0508e

To disable WebSphere sample applications

- 1** Start the WebSphere Application Server and open the **Administrative Console**.
- 2** In the left navigation pane, click **Applications > Enterprise Applications**.
The **Enterprise Applications** page opens.
- 3** Select the checkbox next to each sample application you wish to stop and click **Stop**.

To prevent sample applications from restarting when WebSphere Application Server restarts

- 1** Follow the steps in the procedure entitled "[To disable WebSphere sample applications](#)" on page 54.
- 2** Click the application name.
The sample application page opens.
- 3** Under the **Configuration** tab, **Additional Properties**, click **Target Mappings**.
- 4** Click the target (for example, `server1`) you want to disable.
The target page opens.
- 5** Under the **Configuration** tab, **General Properties**, clear the **Enabled** checkbox and click **Apply**.
- 6** Restart WebSphere Application Server.

Configuration

You can use configuration settings to customize your Application Server Runtime environment.

Configuration file

The `configuration.global.xml` file contains configuration settings for the Application Server Runtimes. Each installation of Application Server Runtimes has its own configuration file.

The default location of the configuration file is:

- On Windows — `C:\Program Files\Entrust\AppServerRuntimes\config\configuration.global.xml`
- On Solaris — `/opt/entrust/AppServerRuntimes/config/configuration.global.xml`

Editing the configuration file

The configuration file contains settings that apply to both the Application Server Runtime for WebLogic and the Application Server Runtime for WebSphere. The file also contains service-specific settings, which apply only to the Application Server Runtime for WebSphere. There are no configuration settings that are specific to the Application Server Runtime for WebLogic.

The configuration file is a well-formed XML document. Make a backup of `configuration.global.xml` before you start and use a text editor to edit the file, but do not use non-ASCII characters.

To allow your configuration changes to take effect, save the file and restart your Application Server Runtime.

Configuration settings

The following sample shows the available configuration settings highlighted in bold. [Table 2 on page 58](#) explains the settings.

```
<ent-configuration>
  <global>
    <gaIdentificationServer>
      <credentialsCollectorUrl>
        http://jsmith.entrust.com:50002/ies/CredentialCollector
      </credentialsCollectorUrl>

      <authenticationAuthorityUrl>
        http://jsmith.entrust.com:50002/ies/AuthenticationAuthority
      </authenticationAuthorityUrl>
    </gaIdentificationServer>
  </global>
</ent-configuration>
```

```

    <attributeAuthorityUrl>
      http://jsmith.entrust.com:50002/ies/AttributeAuthority
    </attributeAuthorityUrl>
  </gaIdentificationServer>

  <asrLogging>
    <logFile>
      C:/Program Files/entrust/AppServerRuntimes/logs/asr.log
    </logFile>

    <!-- The possible values for the log level:
      TRACE
      DEBUG
      INFO
      WARNING
      ERROR
      ALERT
      FATAL
    -->
    <logLevel>INFO</logLevel>
  </asrLogging>

  <asrSessionIdCache>
    <enableCaching>true</enableCaching>
    <cacheProperties>
      <maxSize>1000</maxSize>
      <timeToLive>120</timeToLive>
    </cacheProperties>
  </asrSessionIdCache>

  <asrUserRolesCache>
    <enableCaching>true</enableCaching>
    <cacheProperties>
      <maxSize>1000</maxSize>
      <timeToLive>120</timeToLive>
    </cacheProperties>
  </asrUserRolesCache>

  <gaSessionCookieName>AUTH_SESSION_ID</gaSessionCookieName>
</global>

<services>
  <WebLogicASR>
    <global></global>
  </WebLogicASR>

  <WebSphereASR>
    <global>

```

```
<!-- Name of the Realm returned by the Custom Registry -->
<WebSphereSecurityRealm>Entrust Realm</WebSphereSecurityRealm>
<WebSphereRoleCategory>
  <!-- If "ignore" is true, for roles belonging to the specified
        "roleCategoryID", the Authorization Table will ignore the role
        CategoryID in the GA role when comparing it to the J2EE role.
        If "ignore" is false, the J2EE role must match a
        <GARoleCategoryID>.<GARoleCategory> from the GA backend and
        the"roleCategoryID" setting has no effect.-->
  <ignore>true</ignore>
  <roleCategoryID>websphere</roleCategoryID>
</WebSphereRoleCategory>
</global>
</WebSphereASR>
</services>
</ent-configuration>
```

Table 2: Configuration settings

Configuration setting	Description
<pre><gaIdentificationServer> <credentialsCollectorUrl> <authenticationAuthorityUrl> <attributeAuthorityUrl></pre>	<p>The <gaIdentificationServer> element contains three child elements whose contents specify the URLs of the three components that comprise the Entrust Identification Server. Refer to the <i>Entrust Identification Server and Entrust Entitlements Server 7.0 Administration Guide</i> for information about these components.</p> <p><credentialsCollectorUrl> — specifies the URL of the Credentials Collector servlet.</p> <p><authenticationAuthorityUrl> — specifies the URL of the SAML Authentication Authority.</p> <p><attributeAuthorityUrl> — specifies the URL of the SAML Attribute Authority.</p>
<pre><asrLogging> <logFile> <logLevel></pre>	<p>The contents of the two child elements of <asrLogging> specify the location of the Application Server Runtimes log file and the logging level. Refer to the section called “Error logging” on page 115 for information about error logging.</p> <p><logFile> — specifies the path to the Application Server Runtimes error log file. The default location of the log file is:</p> <p>On Windows — C:\Program Files\Entrust\AppServerRuntimes\logs\asr.log</p> <p>On Solaris — /opt/entrust/AppServerRuntimes/logs/asr.log</p> <p><logLevel> — contains the error logging level.</p> <p>Default setting: INFO.</p>

Table 2: Configuration settings

Configuration setting	Description
<pre><asrSessionIdCache> <enableCaching> <cacheProperties> <maxSize> <timeToLive></pre>	<p>The <code><asrSessionIdCache></code> element contains settings that allow you to enable or disable session ID caching and to adjust cache properties.</p> <p>The settings you can change are:</p> <ul style="list-style-type: none"><code><enableCaching></code> — set to true or false. <p>Default setting: true</p> <ul style="list-style-type: none"><code><cacheProperties></code> — contains two child elements<code><maxSize></code> — sets the maximum size of the session ID cache. <p>Default setting: 1000 entries.</p> <ul style="list-style-type: none"><code><timeToLive></code> — sets the time in seconds for which the user's session remains valid after revocation. Entrust recommends you set this time to match the value of the Entrust GetAccess configuration setting called <code><gaStandardSessionDecisionCache></code>. The valid range for this setting is 1 second or greater. <p>Default setting: 120 seconds.</p>
<pre><asrUserRolesCache> <enableCaching> <cacheProperties> <maxSize> <timeToLive></pre>	<p>The <code><asrUserRolesCache></code> element contains settings that allow you to enable or disable user role caching and to adjust cache properties. The settings you can change are:</p> <ul style="list-style-type: none"><code><enableCaching></code> — set to true or false. <p>Default setting: true</p> <ul style="list-style-type: none"><code><cacheProperties></code> — contains two child elements<code><maxSize></code> — sets the maximum size of the cache. The default size is 1000 entries.<code><timeToLive></code> — sets the time interval in seconds between checks for changes to the user's list of roles. The valid range for this setting is 1 second or greater. <p>Default setting: 120 seconds.</p>

Table 2: Configuration settings

Configuration setting	Description
<gaSessionCookieName>	The name of the Entrust GetAccess session cookie. The value of this setting must match the <gaSessionCookieName> in the Entrust GetAccess configuration file. Default setting: AUTH_SESSION_ID.
<WebLogicASR>	There are no configuration settings that are specific to the Application Server Runtime for WebLogic.
<WebSphereASR> <WebSphereSecurityRealm> <WebSphereRoleCategory> <ignore> <roleCategoryID>	<WebSphereSecurityRealm> — short description of the currently defined security realm. <WebSphereRoleCategory> — contains child elements that specify whether the Authorization Table ignores a specific Entrust GetAccess role category when determining a user's privileges. <ignore> — set to true or false. The default true setting allows the WebSphere Administrative Console application to run without redeployment. Do not change this setting to false, unless you intend to modify the roles required by the Administrative Console application in its deployment descriptor. Default setting: true <roleCategoryID> — specifies the role category ID that is ignored by the Authorization Table when <ignore> is set to true. If <ignore> is set to false, this setting is not applicable. Default setting: websphere

Note: The Application Server Runtime for WebSphere uses the <asrSessionIdCache> settings only when it is protecting resources being accessed through a servlet or JSP. In this case, Entrust GetAccess manages the session.

When Java clients gain access to protected resources directly, the session is managed by the WebSphere Application Server and the configuration settings for the session cache have no effect.

Confirming that Application Server Runtime software is working

When you have completed the post installation tasks, you can confirm that the Application Server Runtime software is working by examining the log file in `<Application Server Runtime installation directory>\logs\asr.log`.

When you have enabled Entrust security for WebLogic Server, the log file should contain `Provider Initialized` logs.

If you are using WebLogic Server and have added an Entrust servlet filter to intercept requests to a JSP, the log file should contain an `Identity Verification servlet filter initialized` log.

Chapter 5

Verifying installation

This chapter describes how to confirm that the Entrust Application Server Runtimes are properly installed and configured to protect the EJB resources of your Web application server.

Topics in this chapter:

- [“Overview” on page 64](#)
- [“Testing WebLogic samples without EJB protection” on page 65](#)
- [“Testing WebLogic samples with EJB protection” on page 69](#)
- [“Testing WebSphere samples without EJB protection” on page 79](#)
- [“Testing WebSphere samples with EJB protection” on page 81](#)

Overview

When you have installed the Application Server Runtimes and configured your Web application server to use Entrust security, you can confirm that the software is operating correctly by performing the tasks described in this section of the guide. These tasks use the WebLogic Examples Server and the WebSphere Technology Samples and verify their operation both before and after you turn on Entrust security.

Related documentation

This subsection lists documentation related to the procedures in this section.

WebLogic Server

- WebLogic e-docs — <http://e-docs.bea.com/wls/docs70/>
- WebLogic samples and tutorials — <http://e-docs.bea.com/wls/docs70/samples.html>
- Programming WebLogic Enterprise JavaBeans — <http://e-docs.bea.com/wls/docs70/ejb/>
- Using Web server plug-ins with WebLogic Server — <http://e-docs.bea.com/wls/docs70/plugins/>

WebSphere Application Server

- WebSphere Application Server InfoCenter (home page) — <http://www-3.ibm.com/software/webservers/appserv/infocenter.html>
- WebSphere Application Server InfoCenter (for Version 5.0) — <http://publib7b.boulder.ibm.com/webapp/wasinfo1/index.jsp?deployment=ApplicationServer&lang=en>

Testing WebLogic samples without EJB protection

The procedures in this section use the WebLogic Examples Server to confirm that your WebLogic Server installation is working correctly. If you chose not to install the WebLogic Server examples when you installed WebLogic Server, you must install them before you perform the procedures in this section.

Complete the tasks in the order they appear.

Launching WebLogic Examples Server

The procedure in this subsection describes how to start the WebLogic Examples Server and confirm that it is running.

To launch the WebLogic Examples Server

- 1 Launch the Examples server.

On Windows, from the **Start** menu, click **Programs > BEA WebLogic Platform 7.0 > WebLogic Server 7.0 > Server Tour and Examples > Launch Examples Server**.

On Solaris, run the `startExamplesServer.sh` script file

```
<SAMPLES_HOME>/server/config/examples/startExamplesServer.sh
```

where `SAMPLES_HOME` is `<location of WebLogic installation>/samples`

- 2 Open the URL `http://<hostname>:7001/examplesWebApp`, where `hostname` is the name of the computer on which WebLogic Server is running and assuming that the default WebLogic Server listen port is 7001.

This step confirms that the Examples Server is running.

Running a Java client-EJB example

Before you protect an EJB using Entrust security, perform this procedure to check that you can gain access to an EJB from a basic Java client.

To gain access to a basic EJB from a Java client

Note: The `SAMPLES_HOME\server\src\examples\security\jaas\package-summary.html` document describes this example and its operation.

- 1 Build the example.

Open `SAMPLES_HOME\server\src\examples\security\jaas\package-summary.html` and follow steps 1 to 4 in the subsection entitled "Build the Example".

This step uses the Apache Ant build tool to build the example and sets up the Java client to use username/password authentication.

2 Run the example.

Open `SAMPLES_HOME\server\src\examples\security\jaas\package-summary.html` and follow steps 1 and 2 in the subsection entitled "Run the Example".

The example writes output similar to the following when it runs successfully:

```
C:\bea\weblogic700\samples\server\src\examples\security\jaas>ant
run
Buildfile: build.xml
```

```
run:
```

```
[java] username: weblogic
[java] password: *****
[java] URL: t3://localhost:7001
[java] Creating a trader
[java] Buying 100 shares of BEAS.
[java] Buying 200 shares of MSFT.
[java] Buying 300 shares of AMZN.
[java] Buying 400 shares of HWP.
[java] Selling 100 shares of BEAS.
[java] Selling 200 shares of MSFT.
[java] Selling 300 shares of AMZN.
[java] Selling 400 shares of HWP.
[java] Removing the trader
```

```
BUILD SUCCESSFUL
```

```
Total time: 2 seconds
```

Running a JSP-to-EJB example

The file `SAMPLES_HOME\server\src\examples\jsp\EJBManagedClient.jsp` demonstrates a JSP that accesses an EJB using WebLogic's built-in Web server. The `package-summary.html` file in the same folder describes the example.

Running the test using an external Web server prepares the ground for using the Application Server Runtime to protect an EJB that is accessed by a browser through a JSP. To protect the EJB in this configuration, the Application Server Runtime relies on the interception of requests by Entrust GetAccess runtime software on an external Web server.

You can use any Web server that supports Entrust GetAccess Runtime software and Web server plug-ins for WebLogic Server, for example the Sun One Web Server or the Microsoft Internet Information Server. Refer to the Entrust Customer Support Platform Support and Integration Center (<https://www.entrust.com/support/psic/index.cfm>) for information about supported software.

To test JSP access to the EJB using WebLogic's built-in Web server

- 1** Open the URL
`http://<hostname>:7001/examplesWebApp/EJBeanManagedClient.jsp`, where `<hostname>` is the name of the computer where WebLogic Server is running and assuming that the default WebLogic Server listen port is 7001.
- 2** Ensure that you see a page with the heading "JSP Example using EJB-managed persistence" and no errors. Disregard the exception in "Part B".

To test JSP access to the EJB using an external Web server

- 1** Install the WebLogic Web server plug-in for the Web server you want to use. Refer to the BEA e-docs that describe how to use Web server plug-ins with WebLogic Server — <http://e-docs.bea.com/wls/docs70/plugins/index.html>.
- 2** Set up the plug-in to act as a proxy based on the URL of the requests to WebLogic Server.
Follow the examples in the WebLogic documentation to set up a proxy for WebLogic Server, for all relative URLs starting with `/examplesWebApp`.
- 3** Restart your Web server.
- 4** Confirm that your plug-in is working.
Open the URL `http://<hostname:port_number>/examplesWebApp` where `hostname:port_number` is the host name and port number of your Web server.
- 5** Modify the deployed `EJBeanManagedClient.jsp` file (written originally for use with WebLogic's built-in Web server) to work with the external Web server.
In a text editor, open

```
SAMPLES_HOME\server\stage\examples\examplesWebApp\EJBeanManagedClient.jsp
```

Change the line

```
String url = "http://" + request.getServername() + ":" +  
request.getServerPort();  
to  
String url = "http://localhost:7001";
```

Save the JSP file.

- 6 Confirm that the JSP is working.

Open the URL

```
http://<hostname:port_number>/examplesWebApp/  
EJBManagedClient.jsp
```

where `hostname:port_number` is the host name and port number of your Web server.

Testing WebLogic samples with EJB protection

The procedures in this section enable Entrust security to protect the WebLogic Examples Server samples described in the previous section.

Complete the tasks in the order they appear.

Enabling Entrust security

This subsection contains a procedure that describes how to turn on Entrust security for the WebLogic Examples Server.

To enable Entrust security for WebLogic Examples Server

Note: You do not need to perform this procedure if you selected the WebLogic Examples Server as the domain to configure when you installed the Application Server Runtime

- 1 Follow the procedure called [“To set up Application Server Runtime software in additional WebLogic domains”](#) on page 41.
-

Note: The startup script referred to in steps 4 to 7 of this procedure is the script that starts the Examples Server (`startExamplesServer.cmd` on Windows, and `startExamplesServer.sh` on Solaris).

- 2 Restart the Examples Server.
- 3 Create an Entrust realm in the Examples Server.
Follow the procedure entitled [“To create an Entrust realm using the createEntrustRealm script”](#) on page 39.
- 4 Set the Entrust realm as the default security realm for the Examples Server.
Refer to the procedure called [“To configure an Entrust realm in a WebLogic domain”](#) on page 41.
- 5 Restart the Examples Server.
- 6 Confirm that the Application Server Runtime software is operating correctly.
Refer to the subsection entitled [“Confirming that Application Server Runtime software is working”](#) on page 61.

Configuring EJB permissions

This subsection contains procedures that describe how to configure EJB permissions for the WebLogic Server examples.

To configure EJB permissions using the Administration Console

Note: This procedure adds method permissions to the EJB tested in the procedure called [“To gain access to a basic EJB from a Java client”](#) on page 65.

- 1** Ensure the Examples Server is running.
- 2** Open the WebLogic Administration Console.
- 3** In the left pane, click **Deployments > EJB > ejb20_basic_statelessSession.jar**.
- 4** In the right pane, click **Edit EJB Descriptor**.
A new browser window opens.
- 5** In the left pane, click **Assembly Descriptor > Security Roles**.
- 6** In the right pane, click **Configure a new Security Role**.
A new window opens in the right pane.
- 7** Under the **Configuration** tab, type a description for the new role in the **Description** field (optional).
Type the role name you want to use in the **Role Name** field.
The role name must be an Entrust GetAccess role in the format `categoryID.RoleID` — `excacc.marketing`, for example.
Click **Create**.
- 8** In the left pane, click **Method Permissions**.
A new window opens in the right pane.
- 9** In the right pane, click **Configure a new Method Permission**.
Type the role name you specified in [Step 7](#).
Type a description for the method permission (optional).
- 10** In the left pane, click **Method Permissions > <method permission name> > Methods**.
A new window opens in the right pane.
- 11** In the right pane, click **Configure a new Method**.
- 12** Under the **Configuration** tab, in the **Description** field, type a description (optional).
In the **EJB Name** field, type `statelessSession`.
In the **Method Name** field, type `buy`.

Click **Create**.

- 13** Repeat [Step 12](#), but in the **Method Name** field, type `sell`.
- 14** In the left pane click the root node of the tree, `ejb20_basic_statelessSession.jar`. In the right pane, under the **Configuration** tab, click **Persist** to complete the procedure.
You have now configured EJB method protection.

To configure EJB permissions by editing the EJB deployment descriptor

Note: In the WebLogic Examples Server domain, the EJB descriptor for this example is in the following location:

```
<SAMPLES_HOME>\server\config\examples\applications\  
ejb20_basic_statelessSession.ear\ejb20_basic_statelessSession.jar\  
ejb-jar.xml
```

-
- 1** Open the deployment descriptor `ejb-jar.xml` file for editing in a text editor.
 - 2** Make the changes shown in the following code fragment to the EJB deployment descriptor.

The code fragment shows an example of the `<assembly-descriptor>` element of an EJB deployment descriptor, with child elements whose contents match the changes made in the Administration Console configuration procedure described earlier:

```
<assembly-descriptor>  
  <security-role>  
    <role-name>excacc.marketing</role-name>  
  </security-role>  
  <method-permission>  
    <role-name>excacc.marketing</role-name>  
    <method>  
      <ejb-name>statelessSession</ejb-name>  
      <method-name>buy</method-name>  
    </method>  
    <method>  
      <ejb-name>statelessSession</ejb-name>  
      <method-name>sell</method-name>  
    </method>  
  </method-permission>  
  ...  
</assembly-descriptor>
```

Gaining access to an EJB from a Java client

The procedure in this subsection describes how to run a sample that demonstrates both unsuccessful and successful access from a Java client to a protected EJB on WebLogic Server.

To gain access to a protected EJB from a Java client

- 1 Refer to the procedure called [“To gain access to a basic EJB from a Java client” on page 65](#) and run the example in

```
<SAMPLES_HOME>\server\src\examples\security\jaas.
```

The example should fail because the user for the script, `weblogic`, does not have the appropriate permissions associated with the EJB.

For example:

```
<SAMPLES_HOME>\server\src\examples\security\jaas\ant run
Buildfile: build.xml

run:
    [java] username: weblogic
    [java] password: *****
    [java] URL: t3://localhost:7001
    [java] Creating a trader
    [java] Buying 100 shares of BEAS.
    [java] java.rmi.AccessException: Security Violation: User:
    'weblogic' has insufficient permission to access EJB:
    type=<ejb>,
    application=_appsdir_ejb20_basic_statelessSession_ear,
    module=ejb20_basic_statelessSession.jar, ejb=statelessSession,
    signature=[java.lang.String.int}
```

- 2 Edit the Apache Ant build file,
`<SAMPLES_HOME>\server\src\examples\security\jaas\build.xml`,
to specify a valid Entrust GetAccess username and password.

The username and password are on the following line in `build.xml`:

```
<arg line="ts://localhost:${PORT} weblogic weblogic"/>
```

Change `weblogic weblogic` to the username and password of your choice and save the file.

Make sure the user is associated with the role name you configured in [Step 7 on page 70](#), `excacc.marketing`, for example.

- 3 Run the example again.

```
<SAMPLES_HOME>\server\src\examples\security\jaas\ant run
Buildfile: build.xml

run:
    [java] username: <Entrust GetAccess user>
```

```
[java] password: *****
[java] URL: t3://localhost:7001
[java] Creating a trader
[java] Buying 100 shares of BEAS.
[java] Buying 200 shares of MSFT.
[java] Buying 300 shares of AMZN.
[java] Buying 400 shares of HWP.
[java] Selling 100 shares of BEAS.
[java] Selling 200 shares of MSFT.
[java] Selling 300 shares of AMZN.
[java] Selling 400 shares of HWP.
[java] Removing the trader
```

BUILD SUCCESSFUL

Total time: 6 seconds

Protecting an EJB with Entrust GetAccess roles

This procedure adds method permissions to the EJB tested in the procedure called [“To test JSP access to the EJB using an external Web server” on page 67](#).

This procedure requires that the JSP be a protected Entrust GetAccess resource.

To protect a JSP using the Entrust GetAccess Runtime

- 1 If you have not already done so, install the Entrust GetAccess Runtime on the Web server you used in the procedure called [“To test JSP access to the EJB using an external Web server” on page 67](#).

Refer to the *Entrust GetAccess 7.0 Planning and Installation Guide* for installation instructions.

- 2 Restart the Web server.
- 3 Open the Entrust GetAccess Administration Console and add the JSP on the Web server as an Entrust GetAccess resource.

Refer to the *Entrust GetAccess 7.0 Business Administration Guide* for instructions about creating resources.

The relative URL for the JSP is `/examplesWebApp/EJBManagedClient.jsp`. Add the relative URL as an item to protect.

Make the resource available to all users (otherwise you must assign a role to the resource) and leave other settings at their defaults.

Update the server configuration in Entrust GetAccess Administration Console and restart the Web server.

4 Open the URL

`http://<hostname>/examplesWebApp/EJBManagedClient.jsp` to confirm that the JSP is protected by Entrust GetAccess

where `hostname` is the computer on which the Web server is running.

You should see a login page.

Login to Entrust GetAccess with a valid username and password and confirm that you can see the JSP page.

Refer to the *Entrust GetAccess 7.0 Troubleshooting Guide* if you encounter problems using the Entrust GetAccess Runtime software.

Configuring a Web application's servlet filter

To protect your Web application's EJB resources when browser-based clients request access to them indirectly through a servlet or JSP that is protected by Entrust GetAccess runtime software on a Web server, you must set up the `EntrustGetAccessIdentityVerificationServletFilter` servlet filter. You must perform this task for every Web application containing EJB resources you want to protect with Entrust security.

To configure a servlet filter using the WebLogic Administration Console

Note: This procedure uses the WebLogic `examplesWebApp` application to demonstrate configuration steps, but the sequence you should follow is the same for all other Web applications.

- 1 Ensure the WebLogic Server is running.
- 2 Open the Administration Console.
- 3 In the left pane, click **Deployments > Web Applications > examplesWebApp**.
A new window opens in the right pane.
- 4 In the right pane, click **Edit Web Application Deployment Descriptors**.
A new browser window opens.
- 5 In the left pane, click **Filters**.
A new window opens in the right pane.
Click **Configure a new Filter**.
- 6 Under the **Configuration** tab, in the **Filter Name** field, type **EntrustGetAccessIdentityVerificationServletFilter**.
In the **Filter Class** field, type

```
com.entrust.stp.asr.weblogic.EntrustGetAccessIdentityVerificationServletFilter
```

You can leave the other fields under the Configuration tab empty — they are optional.

Click **Create**.

- 7** In the left pane, click **Filter Mappings**.

A new window appears in the right pane.

Click **Configure a new Filter Mapping**.

- 8** Under the **Configuration** tab, from the **Filter** drop-down list box, select **EntrustGetAccessIdentityVerificationServletFilter**.

In the **Url Pattern** field, type `EJBeanManagedClient.jsp` (to intercept requests to the WebLogic EJBeanManagedClient JSP sample, for example)

Leave the **Servlet** drop-down list box at the default setting (none).

Click **Create**.

- 9** In the left pane click the root node of the tree.

- 10** In the right pane, under the **Configuration** tab, click **Persist**.

- 11** Restart the server.

- 12** Refer to the subsection entitled [“Confirming that Application Server Runtime software is working” on page 61](#) to find out how to confirm that the servlet filter is working.

To configure a servlet filter by editing the deployment descriptor

- 1** Open the Web application’s `web.xml` deployment descriptor in a text editor.

The deployment descriptor is in the `WEB-INF` directory under your Web application. For example

```
<bea home>\user_projects\mydomain\applications\MyWebApp\WEB-INF
```

- 2** Add the following filter declaration to the deployment descriptor to define the filter’s name and class file name.

```
<filter>
  <filter-name>
    EntrustGetAccessIdentityVerificationServletFilter
  </filter-name>
  <filter-class>
com.entrust.stp.asr.weblogic.EntrustGetAccessIdentityVerificationServletFilter
  </filter-class>
</filter>

<filter-mapping>
```

```
<filter-name>
    EntrustGetAccessIdentityVerificationServletFilter
</filter-name>
<url-pattern>EJBeanManagedClient.jsp</url-pattern>
</filter-mapping>
```

- 3** In the `<url-pattern>` element of the deployment descriptor, change the element's content to a pattern that specifies all the servlets and JSPs that provide access to EJBs protected by the Application Server Runtime for WebLogic.
If you use an asterisk (*) as the url-pattern, the servlet filter will be invoked for all of the resources within the Web application.
- 4** Save the amended `web.xml` deployment descriptor.
- 5** Refer to the subsection entitled "[Confirming that Application Server Runtime software is working](#)" on page 61 to find out how to confirm that the servlet filter is working.

Note: Refer to the BEA online documentation entitled *Assembling and Configuring Web applications* (<http://e-docs.bea.com/wls/docs70/webapp/index.html>) for more information about configuring servlet filters for WebLogic Server applications.

To configure EJB permissions using the Administration Console

- 1** Ensure the Examples Server is running.
- 2** Open the WebLogic Administration Console.
- 3** In the left pane, click **Deployments > EJB > ejb20_basic_beanManaged.jar**.
A new window appears in the right pane.
- 4** Click **Edit EJB descriptor**.
A new browser window opens.
- 5** In the left pane, click **Assembly Descriptor > Security Roles**.
A new window appears in the right pane.
- 6** Click **Configure a new Security Role**.
- 7** Under the **Configuration** tab, in the **Role Name** field, type an Entrust GetAccess role name.
The role name must be an Entrust GetAccess role in the format `categoryID.RoleID` — `excacc.marketing`, for example.
- 8** In the left pane, click **Method Permissions**.
A new window opens in the right pane.
- 9** In the right pane, click **Configure a new Method Permission**.

Type the role name you specified in [Step 7](#).

Type a description for the method permission (optional).

- 10** In the left pane, click **Method Permissions > MyMethod Permission > Methods**.

A new window opens in the right pane.

- 11** In the right pane, click **Configure a new Method**.

- 12** Under the **Configuration** tab, in the **Description** field, type a description (optional).

In the **EJB Name** field, type `beanManaged`.

In the **Method Name** field, type `deposit`.

Click **Create**.

- 13** Repeat [Step 12](#) to create methods called `withdraw` and `balance`.

You have now configured EJB method protection.

- 14** In the left pane click the root node of the tree, `ejb20_basic_beanManaged.jar`.

- 15** In the right pane, under the **Configuration** tab, click **Persist** to complete the procedure.

To configure EJB permissions by editing the EJB deployment descriptor

Note: In the WebLogic Examples Server domain, the EJB descriptor is in the following location:

```
<SAMPLES_HOME>\server\config\examples\applications\  
ejb20_basic_beanManaged.ear\ejb20_basic_beanManaged.jar\  
ejb-jar.xml
```

-
- 1** Open the deployment descriptor XML file for editing in a text editor.
 - 2** Make the changes shown in the following XML fragment to the EJB deployment descriptor.

The XML fragment shows an example of the `<assembly-descriptor>` element of an EJB deployment descriptor, with child elements whose contents match the changes made in the Administration Console configuration procedure described earlier:

```
<assembly-descriptor>  
  <security-role>  
    <role-name>excacc.marketing</role-name>  
  </security-role>  
  <method-permission>  
    <role-name>excacc.marketing</role-name>  
    <method>
```

```

        <ejb-name>beanManaged</ejb-name>
        <method-name>deposit</method-name>
    </method>
    <method>
        <ejb-name>statelessSession</ejb-name>
        <method-name>withdraw</method-name>
    </method>
    <method>
        <ejb-name>statelessSession</ejb-name>
        <method-name>balance</method-name>
    </method>
</method-permission>
...
</assembly-descriptor>

```

- 3 Save the file to complete the procedure.

Gaining access to an EJB from a browser through a JSP

At the end of the short procedure in this subsection you should see a JSP page with no errors. This confirms that an Entrust GetAccess user associated with appropriate roles can make requests to a JSP from a browser for access to protected EJBs on WebLogic Server

To gain access to protected EJBs from a browser through a JSP

- 1 Open the URL

`http://<hostname>/examplesWebApp/EJBeanManagedClient.jsp`

where `hostname` is the Web server on which the WebLogic Web server plug-in and the Entrust GetAccess Runtime are installed.

Login using Entrust GetAccess user who is not associated with the `excacc.marketing` role.

The JSP page should display an error.

- 2 Close your browser and reopen the URL

`http://<hostname>/examplesWebApp/EJBeanManagedClient.jsp`.

Login using an Entrust GetAccess username and password for a user associated with the role `excacc.marketing`.

The JSP page should appear with no errors.

Testing WebSphere samples without EJB protection

The WebSphere Samples Gallery contains Web application samples that demonstrate J2EE applications.

Running WebSphere Application Server samples

Run the WebSphere Technology Sample applications to confirm that your WebSphere Application Server installation is working correctly before you test them with the Application Server Runtime for WebSphere.

The steps in this subsection describe how to test the WebSphere Application Server Basic Calculator sample both with the WebSphere client application samples and using a Web browser.

To test the Basic Calculator sample with a Java client

- 1 Locate the `launchClient` script file.

For a J2EE client, the default location on Windows is:

```
C:\Program Files\WebSphere\AppServer\bin\
launchClient.bat
```

The default location on Solaris is:

```
/opt/WebSphere/AppServer/bin/launchClient.sh
```

- 2 Run the script file using the following command:

```
launchClient <WAS_HOME>\samples\lib\TechnologySamples\
TechnologySamples.ear -CCjar=BasicCalculatorClient.jar add 1 2
```

where `WAS_HOME` (default location) is `C:\Program Files\WebSphere\AppServer` on Windows operating systems, and `/opt/WebSphere/AppServer` on Solaris operating systems.

Note: If security is turned on, you will be prompted for a username and password even if you have not yet protected EJB methods. Type a valid Entrust GetAccess username and password (no roles necessary).

When the operation completes successfully, you should see output similar to the following:

```
C:\Program Files\WebSphere\AppClient\samples\bin\launchClient C:\Program Files\
```

```
WebSphere\samples\TechnologySamples\TechnologySamples.ear
-CCjar=BasicCalculatorClient.jar add 1 2
```

```
IBM WebSphere Application Server, Release 5.0
J2EE Application Client Tool
Copyright IBM Corp., 1997-2002
WSQL0012I: Processing command line arguments.
WSQL0013I: Initializing the J2EE Application Client Environment.
WSQL0035I: Initialization of the J2EE Application Client Environment
has completed.
WSQL0014I: Invoking the Application Client class
com.ibm.websphere.samples.technologysamples.basiccalclient.
BasicCalculatorClientJ2EEMain
--Creating InitialContext... Done.
--Looking-up Home... Done.
--Narrowing... Done.
--Creating Home... Done.
Result: 1+2 = 3
Press any key to continue . . .
```

To test the Basic Calculator sample with a Web browser

- 1** Locate and run the Technology Samples in the Samples Gallery
<http://<Web server with Entrust GetAccess Runtime>/WSsamples>.
- 2** In the left pane, click **Enterprise beans**.
The Enterprise beans page opens.
- 3** On the **Enterprise beans** page, under **Stateless Session - Basic Calculator**, click **Run**.
The **Stateless Session EJB - Basic Calculator** window opens allowing you to perform basic arithmetical operations.

Testing WebSphere samples with EJB protection

The procedures in this section describe how to use the Application Server Runtimes to protect the EJBs in the WebSphere Technology Samples you tested in the previous section.

Complete the tasks in the order they appear.

Creating Entrust GetAccess users and resources

When you have enabled Entrust security for the WebSphere Technology Samples, only Entrust GetAccess users associated with specific roles will have access to protected resources.

Note: When you have completed all the procedures in this section both `user1` and `user2` (which you create in the first procedure) will be able to open the JSP associated with the calculator sample. Only `user2` will have permission to use the `makeSum` method of the Basic Calculator EJB.

To set up Entrust GetAccess users for WebSphere Technology Samples

- 1 Ensure you have completed the relevant post-installation tasks in [“Performing post-installation tasks” on page 37](#).
- 2 Create an Entrust GetAccess role category with the following role category information:
 - ID — calculator
 - Name — calculator
 - Description — calculatorRefer to the *Entrust GetAccess 7.0 Business Administration Guide* for information about how to perform the tasks described in the following steps.
- 3 Create an Entrust GetAccess role in the `calculator` category with the Role ID of `sum`.
- 4 Create an Entrust GetAccess role in the `websphere` category with the Role ID of `GetAccessUser`.
- 5 Create an Entrust GetAccess user, `user1` for example, and assign the user the role of `websphere.GetAccessUser`.

- 6 Create an Entrust GetAccess user, `user2` for example, and assign the user the roles of `websphere.GetAccessUser` and `calculator.sum`.

To set up Entrust GetAccess resources for WebSphere Technology Samples

- 1 Install the Entrust GetAccess Runtime on a Web server between the client browser and WebSphere Application Server.

Refer to the *Entrust GetAccess 7.0 Planning and Installation Guide* for installation instructions.

- 2 Restart the Web server.

- 3 Open the Entrust GetAccess Administration Console and add the JSP on the Web server as an Entrust GetAccess resource.

Refer to the *Entrust GetAccess 7.0 Business Administration Guide* for instructions about creating resources.

The relative URL for the JSP is `/TechnologySamples/BasicCalculator/*`. Add the relative URL as an item to protect.

Add the URI `/TechnologySamples/BasicCalculator/*` item to protect and leave other settings at their defaults.

Update the server configuration.

Protecting EJBs using Application Server Runtimes

The procedures in this subsection describe the steps you should take in the WebSphere Application Assembly Tool to use the Application Server Runtime for WebSphere to protect the EJBs in the WebSphere Technology Samples.

To configure EJB permissions

Note: This procedure describes how to modify the Technology Samples EAR file. Make a backup copy of the EAR file before you begin.

- 1 Start the WebSphere Application Assembly Tool.

On Windows operating systems, from the **Start** menu, click **Programs > IBM WebSphere > Application Server v5.0 > Application Assembly Tool**.

On Solaris operating systems, run the `assembly.sh` script file,
`<WAS_HOME>\bin\assembly.sh`

where `WAS_HOME` is the location of your WebSphere Application Server installation.

- 2 In the Application Assembly Tool, open
`<WAS_HOME>\samples\lib\TechnologySamples\TechnologySamples.ear`
- 3 In the left pane, click **EJB Modules > Basic Calculator EJB Module** to expand the nodes.
- 4 Right-click **Security Roles** and from the pop-up menu, select **New**.
The **New Security Role** dialog box opens.
- 5 Under the **General** tab, in the **Name** field, type a name for the new role, `calculator.sum`, for example.
Click **OK**.
- 6 In the left pane, right-click **Method Permissions** and from the pop-up menu, select **New**.
The **New Method Permission** dialog box opens.
- 7 Under the **General** tab, click **Add** (next to methods list) to add a method.
The **Add Methods** dialog box opens.
- 8 Click **BasicCalculatorEJB.jar > BasicCalculator > All methods** to expand the nodes, and select `makeSum(double double)`.
Click **OK**.
- 9 Under the **General** tab, click **Add** (next to roles list) to add a role.
The **Application Assembly Tool Dialog** box opens.
- 10 Select `calculator.sum` and click **OK**.
- 11 In the **New Method Permission** dialog box, under the **General** tab, click **Apply** to close the dialog box.
You have now configured EJB permissions to protect the `makeSum` method.

To add the Entrust GetAccess user security constraint to the Web application

Note: This procedure is applicable only when clients gain access to the protected EJB by way of a servlet or JSP.

- 1 Ensure you have completed the procedure called [“To configure EJB permissions” on page 82](#).
- 2 In the Application Assembly Tool, open
`<WAS_HOME>\samples\lib\TechnologySamples\TechnologySamples.ear`
where `WAS_HOME` is the WebSphere Application Server installation directory.
- 3 In the left pane, click **Web Modules > Basic Calculator (Stateless Session)** to expand the nodes.

- 4** Right-click **Security Roles** and from the pop-up menu, select **New**.
The **New Security Role** dialog box opens.
- 5** Under the **General** tab, in the **Name** and **Description** fields, type `websphere.GetAccessUser`.
Click **OK**.
- 6** In the left pane, right-click **Security Constraints** and from the pop-up menu, select **New**.
The **New Security Constraint** dialog box opens.
- 7** Under the **General** tab, in the **Security constraint name** field, type `Basic Calculator`.
Click **Add** to add a role.
The **Application Assembly Tool Dialog** box opens.
- 8** Select `GetAccessUser` and click **OK**.

Note: In the New Security Constraint dialog box, in the User Data Constraint area, leave the Transport guarantee set to None. Do not set it to Integral or Confidential unless you have turned on SSL at the Web server.

- 9** In the **New Security Constraint** dialog box, under the **General** tab, click **OK** to close the dialog box.
- 10** In the left pane, click **Security Constraints > Basic Calculator** to expand the nodes.
- 11** Right-click **Web Resource Collections**, and from the pop-up menu, select **New**.
The **New Web Resource Collection** dialog box opens.
- 12** Under the **General** tab, in the **Web Resource Name** field, type, `Basic Calculator`.
Click **Add** to add a URL pattern.
The **Add URLs** dialog box opens.
- 13** In the **URL pattern** field, type `/*` and click **OK**.
- 14** In the **New Web Resource Collection** dialog box, under the **General** tab, click **OK** to close the dialog box.
- 15** Save your changes and close the Application Assembly Tool.
You have now added a security constraint to protect the Basic Calculator Web application.

To update the WebSphere Technology Samples

- 1** Start WebSphere Application Server and open the Administrative Console.

- 2** In the left pane, click **Applications > Enterprise Applications**.
The **Enterprise Applications** window (allowing you to view and manage your deployed enterprise applications) opens in the right pane.
- 3** In the **Enterprise Applications** window, select the **Technology Samples** checkbox and click **Update**.
A new window opens called, **Preparing for the application update**.
- 4** Under **Local path**, click **Browse** and select the `TechnologySamples.ear` file, `WAS_HOME\samples\lib\TechnologySamples\TechnologySamples.ear`.
Click **Next**.
A new window opens called **Preparing for the application update**.
- 5** Click **Next**.
A new window opens called, **Install New Application**.
Click **Step 14 Summary**, and click **Finish**.
- 6** Click **Save to Master Configuration** and, in the **Enterprise Applications** window, click **Save**.
Only `user2`, created in [“To set up Entrust GetAccess users for WebSphere Technology Samples” on page 81](#), can now gain access to the `makeSum` method of the Basic Calculator EJB.

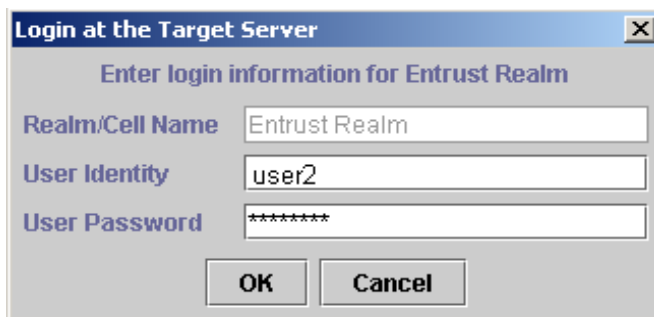
Gaining access to a protected EJB from a Java client

The procedure in this subsection describes how to run script files that demonstrate access to a method of a protected EJB (the Basic Calculator EJB) from a Java client application.

To gain access to a protected EJB from a Java client

- 1** Repeat [Step 1](#) of the procedure called [“To test the Basic Calculator sample with a Java client” on page 79](#).
- 2** Repeat [Step 2](#) of the procedure called [“To test the Basic Calculator sample with a Java client” on page 79](#).

When prompted to do so, type the Entrust GetAccess user ID and password of the user associated with the `calculator.sum` role, `user2`.



If the operation completes successfully, you should see output similar to the following:

```
C:\Program Files\WebSphere\AppClient\samples\bin\TechnologySamplesJ2EEClient>
BasicCalculator\basicCalculator
-----
Running BasicCalculatorClient from the TechnologySamples.ear
file with parameters: add 1 2
-----

IBM WebSphere Application Server, Release 5.0
J2EE Application Client Tool
Copyright IBM Corp., 1997-2002
WSCL0012I: Processing command line arguments.
WSCL0013I: Initializing the J2EE Application Client Environment.
WSCL0035I: Initialization of the J2EE Application Client Environment
has completed.
WSCL0014I: Invoking the Application Client class
com.ibm.websphere.samples.technologysamples.basiccalcclient.
BasicCalculatorClientJ2EEMain
--Creating InitialContext... Done.
--Looking-up Home... Done.
--Narrowing... Done.
--Creating Home... Done.
Result: 1+2 = 3
Press any key to continue . . .
```

If the operation does not succeed, you will see a stack trace and exceptions that indicate authorization has failed. This will happen if you respond to the login prompt with the user ID and password for `user1`. The following is a fragment of output from a failed authorization.

```
java.rmi.RemoteException: ; nested exception is:
com.ibm.websphere.csi.CSIException: SECJ0053E: Authorization
failed for Entrust Realm/user1 while invoking
```



```
(Bean)WSSamples/BasicCalculator makeSum(double,double):1
securityName: Entrust Realm/user1;accessID: user:Entrust
Realm/jsmith is not granted any of the required roles:
calculator.sum
```

Gaining access to an EJB through a JSP

The procedure in this subsection confirms that an Entrust GetAccess user associated with appropriate roles (`user2` with the `calculator.sum` role) can make requests to a JSP from a browser for access to a method of a protected EJB (the Basic Calculator EJB).

To gain access to protected EJBs from a browser through a JSP

- 1 Open the following URL:

```
http://<WebServerWithEntrustGetAccessRuntime>/TechnologySamples/
BasicCalculator/BasicCalculator.jsp
```

- 2 Respond to the login prompt using the user ID and password for `user2`.
The **Stateless Session EJB - Basic Calculator** page opens.
- 3 Perform some standard arithmetical operations, including addition (+), to verify that `user2` has access to the EJB's `makeSum` method.
The page displays the results of successful operations in the browser.
- 4 Repeat [Step 2](#), but respond to the login prompt with the user ID and password for `user1`.

Note: You might have to close all your browser windows to clear any session caches maintained by your browser.

- 5 Repeat [Step 3](#) and verify that `user1` can perform all arithmetical operations except addition.

When you attempt to perform an addition, the browser displays a page saying that the requested page cannot be displayed.

Chapter 6

Securing EJBs programmatically

This chapter briefly describes the EJB programmatic security model you can use to implement security in your EJBs.

This chapter has a single topic:

- [“EJB programmatic security model” on page 90](#)

EJB programmatic security model

If you need to express your security policies programmatically rather than declaratively, the EJB specification describes a programmatic security model. You can use the programmatic security model to determine the identity of users and their associated security roles with methods that allow access to the caller's security context (the container-provided runtime instance of an EJB). These methods are:

- `java.ejb.EJBContext.getCallerPrincipal()` — returns an instance of `java.security.Principal`
- `javax.ejb.EJBContext.isCallerInRole(String rolename)` — returns a `Boolean`

Refer to the EJB specification (<http://java.sun.com/products/ejb/docs.html>) for more information.

Using `getCallerPrincipal`

The `getCallerPrincipal` method allows EJB methods to obtain the name of the principal of the current caller. The EJB invokes the `getCallerPrincipal` method, which returns a `java.security.Principal` object that corresponds to the Entrust GetAccess user currently logged in. Calling `java.security.Principal.getName()` on the returned object provides the user ID of the current caller. For example:

```
EJBContext ejbContext;
...
// get the caller principal
java.security.Principal callerPrincipal =
    ejbContext.getCallerPrincipal();

// get the caller's name
String callerName = callerPrincipal.getName();
```

If you are using WebSphere Application Server, and requests for access to EJBs are being made by way of servlets or JSPs, the `getName` method returns the login name of the user and an appended Entrust GetAccess session ID (SID) in the following format:

```
<userid>+<GetAccess SID>

rcheng+SMS_eponine_9478d2::c49034bdae22b7769469de155d3eacfb
```

When Java clients request direct access to an EJB, the `getName` method returns just the user ID and not the Entrust GetAccess SID.

You can then use the Entrust GetAccess SID to make additional calls to the Identification Server and Entitlements Server as required.

If you are using WebLogic Server, the `getName` method returns the login name of the user without the Entrust GetAccess SID.

Using `isCallerInRole`

The `isCallerInRole` method allows you to perform programmatic security checks that might be in addition to those that have been defined declaratively in the EJB's deployment descriptor, or that might be easier to define programmatically. The EJB invokes the `isCallerInRole` method to determine whether the current caller is assigned a particular security role.

In a WebSphere Application Server environment, the role name argument specified in the method is a reference to a security role defined in the EJB's deployment descriptor. For example:

```
EJBContext ejbContext;  
...  
if (ejbContext.isCallerInRole("SomeRole"))  
    // Perform some function
```

The security role reference is at the level of the individual EJB and it refers to a security role at the EJB module level.

The following XML fragment is taken from the EJB deployment descriptor for the Basic Calculator EJB (`BasicCalculatorEJB.jar`) in the WebSphere Technology Samples EAR file (`AsrTechnologySamples.ear`).

```
<enterprise-beans>  
  <session id="Session_1">  
    <description>  
      Basic Calculator to add, subtract, multiply, divide  
    </description>  
    <display-name>  
      Basic Calculator Stateless Session  
    </display-name>  
    <ejb-name>BasicCalculator</ejb-name>  
    <home>  
      com.ibm.websphere.samples.technologysamples.ejb.stateless.  
      basiccalculatorejb.BasicCalculatorHome  
    </home>  
    <remote>  
      com.ibm.websphere.samples.technologysamples.ejb.stateless.  
      basiccalculatorejb.BasicCalculator  
    </remote>  
    <ejb-class>  
      com.ibm.websphere.samples.technologysamples.ejb.stateless.  
      basiccalculatorejb.BasicCalculatorBean
```

```

    </ejb-class>
    <session-type>Stateless</session-type>
    <transaction-type>Container</transaction-type>
    <security-role-ref id="SecurityRoleRef_1048108430151">
        <role-name>all</role-name>
        <role-link>calculator.all</role-link>
    </security-role-ref>
</session>
</enterprise-beans>

<assembly-descriptor id="AssemblyDescriptor_1">
    <security-role id="SecurityRole_1048105201038">
        <role-name>calculator.all</role-name>
    </security-role>
    ...
</assembly-descriptor>

```

The fragment defines only one role reference, `all`, and this is the only role name that is a valid argument for the `isCallerInRole(String rolename)` method. If you use another role name as the argument, `isCallerInRole` will return `false` and the `WebSphere SystemOut.log` will contain entries similar to the following:

```
[3/19/03 16:17:35:500 EST] 4980bcc8 SecurityColla E SECJ0155E:
Deployment descriptor configuration error. security-role-ref
calculator.all in ejb-jar.xml is not mapped to any security
role in bean BasicCalculator, module BasicCalculatorEJB.jar,
application TechnologySamples.
```

In a WebLogic Server environment, use the `rolename` argument to specify an Entrust GetAccess role in the format `role-categoryID.RoleID`. For example, using the Basic Calculator EJB deployment descriptor mentioned earlier, you would use `calculator.all` as the `rolename` argument.

Note: Roles associated with protected EJBs cannot be macro roles, which comprise one or more standard roles. In Entrust GetAccess, macro roles are assigned only to users, not to resources.

Using servlet security methods

The Java Servlet 2.3 Specification (<http://www.jcp.org/aboutJava/communityprocess/first/jsr053/>) defines three methods that provide programmatic access to the caller's security request information in `javax.servlet.http.HttpServletRequest`.

- `getRemoteUser()` — returns a `String` that represents the user name used by the client to log in
- `isUserInRole(String role)` — allows additional checks on authorization privileges
- `getUserPrincipal()` — returns a `java.security.Principal` object, from which you can extract the name of the current caller using `getName()`

The `javax.servlet.http.HttpServletRequest` methods

`getRemoteUser` and `getUserPrincipal` return null, even if the user is logged in, unless the servlet or JSP is secured.

If you are working with WebSphere Application Server, both

`getRemoteUser()` and `getUserPrincipal().getName()` return the user ID and the Entrust GetAccess session ID in the same format as that returned by the `java.ejb.EJBContext.getCallerPrincipal` method described earlier (refer to [“Using isCallerInRole” on page 91](#)).

If you are working with WebLogic Server, `getRemoteUser()` and `getUserPrincipal().getName()` return the user ID without the session ID of the logged-in Entrust GetAccess user.

Chapter 7

Configuring Application Server Runtimes communications

This chapter provides information about connecting to Entrust GetAccess and configuring Secure Sockets Layer (SSL) communications.

Topics in this chapter:

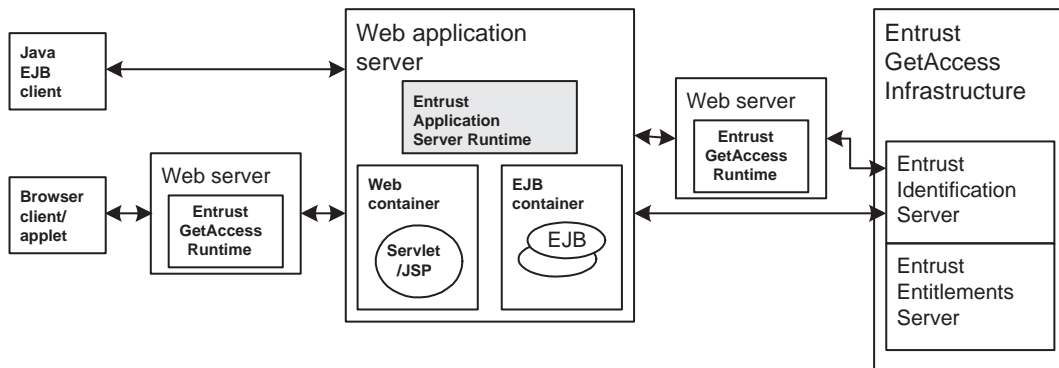
- [“Connecting to Entrust GetAccess” on page 96](#)
- [“Configuring SSL” on page 97](#)

Connecting to Entrust GetAccess

The Application Server Runtimes communicate with Entrust GetAccess by means of Entrust Identification Server, which comprises a set of servlets deployed on the Entrust GetAccess Infrastructure. This makes the Application Server Runtimes clients of Entrust Identification Server.

For information about installing and configuring Identification Server, refer to the *Entrust Identification Server 7.0 and Entrust Entitlements Server 7.0 Administration Guide*. Read the chapter called “Deploying Identification Server and Entitlements Server” for information about where the Application Server Runtimes fit in various deployment scenarios.

Figure 6: Entrust Application Server Runtimes deployment — logical view



You can design your network architecture so that the Application Server Runtimes communicate directly with the Entrust GetAccess Infrastructure through port number 50002 (the default port number), or through port 80 of a Web server running an Entrust GetAccess Runtime. In the latter case, the Entrust GetAccess Runtime is not being used to provide protection, but to forward HTTP requests from the Web server to Entrust GetAccess. You might choose to use a Web server for this purpose (see [Figure 6](#)) for a number of reasons including the following:

- Your network architecture makes it necessary.
- Entrust `gaServicesSSL` are turned on, but you do not want to use SSL between the Application Server Runtimes and Entrust Identification Server.
- Entrust `gaServicesSSL` are turned off, but you want to use SSL between the Application Server Runtimes and Entrust Identification Server.
- You want to use client-authenticated SSL between the Application Server Runtimes and Entrust Identification Server.

Configuring SSL

As clients of Identification Server, the Application Server Runtimes can use server-authenticated SSL to communicate directly through the Entrust GetAccess Infrastructure port (usually, 50002), or by way of a Web server using server-authenticated or client-authenticated SSL.

Refer to the “Configuring SSL” section in the *Entrust Identification Server 7.0 and Entrust Entitlements Server 7.0 Administration Guide* for more information.

Preparing Application Server Runtimes to use SSL

This subsection contains high-level procedures that describe how to prepare the Application Server Runtime for WebLogic Server and the Application Runtime for WebSphere Application Server to use SSL.

Where appropriate, the procedures mention Entrust and third-party tools and applications. The following references point to relevant documentation:

- For information about using the Sun ONE Web Server, refer to its online documentation at <http://docs.sun.com/source/816-5691-10/index.html>. (You can find information about SSL in Chapter 5, “Securing your Web server”.)
- For information about using Microsoft IIS, refer to the IIS help documentation, <http://localhost/iisHelp>.
- For information about using the Java Key and Certificate Management Tool, keytool, refer to <http://java.sun.com/j2se/1.3/docs/tooldocs/win32/keytool.html> or <http://java.sun.com/j2se/1.3/docs/tooldocs/solaris/keytool.html>
- For Entrust documentation, go to the Entrust Customer Support Web site at <https://www.entrust.com/support/documentation/index.cfm> (login required)

To prepare Application Server Runtime for WebLogic to use server-authenticated SSL

- 1 Download and unpack Sun Microsystems' JSSE distribution.
Version 1.0.3_01 of the JSSE is available at <http://java.sun.com/products/jsse/index-103.html>.
- 2 Put the JSSE jar file, `jsse.jar`, on your classpath.
- 3 Import the CA certificate into a key store using Java keytool. For example:

```
>keytool -import -alias <keystore alias> -file <CA cert file name>  
-keystore <keystore name> -storepass <keystorepassword>
```

```
Owner: OU=Entrust PKI Demonstration Certificates, O=Entrust, C=US
Issuer: OU=Entrust PKI Demonstration Certificates, O=Entrust, C=US
Serial number: 3b992f45
Valid from: Fri Sep 07 16:04:13 EDT 2001 until: Tue Sep 07
16:34:13 EDT 2021
Certificate fingerprints:
    MD5: 7D:29:12:AE:43:68:0D:17:03:3A:05:53:1A:5A:6F:03
    SHA1:
13:38:0A:7A:0E:65:56:07:D2:33:1B:37:82:36:B5:48:68:F3:0F:EC
Trust this certificate? [no]: yes
Certificate was added to keystore
```

The CA certificate you import must belong to the CA that issued the Web server certificate.

If you are using an Entrust PKI, you can use an administration tool, such as Entrust Authority™ Security Manager Administration Feature (formerly Entrust/RA) or Entrust/WebConnector™, to obtain the CA certificate.

- 4 Edit the configuration file, `configuration.global.xml`, to update the URLs that specify the three components comprising the Entrust Identification Server.

Refer to the section called ["Configuration" on page 55](#) for information about editing the configuration file. Under the `<gaIdentificationServer>` setting, change `http://...` to `https://...`

For example, change

```
http://hostname.entrust.com/ies/CredentialCollector
```

to

```
https://hostname.entrust.com/ies/CredentialCollector
```

To prepare Application Server Runtime for WebLogic to use client-authenticated SSL

- 1 Create a key pair using Java keytool.

Note: If you are using an Entrust PKI, add a user before moving to the next step. This will give you the reference number to use in the client name.

For example:

```
>keytool -genkey -keyalg RSA -keysize 1024 -keystore <keystore
name> -storepass <keystore password>
```

What is your first and last name?

```
[Unknown]: 34271903
```

```

What is the name of your organizational unit?
  [Unknown]:
What is the name of your organization?
  [Unknown]:
What is the name of your City or Locality?
  [Unknown]:
What is the name of your State or Province?
  [Unknown]:
What is the two-letter country code for this unit?
  [Unknown]:
Is <CN=34271903, OU=Unknown, O=Unknown, L=Unknown, ST=Unknown,
C=Unknown> correct?
  [no]: yes

Enter key password for <mykey>
      (RETURN if same as keystore password):

```

2 Generate a certificate request using Java keytool. For example:

```
>keytool -certreq -keystore <keystore name> -storepass <keystore
password>
```

```

-----BEGIN NEW CERTIFICATE REQUEST-----
MIIBrTCCARYCAQAwbTEQMA4GA1UEBhMHVW5rNm93bjEQMA4GA1UECBMHVW5rNm93bj
EQMA4GA1UEBxMHVW5rNm93bjEQMA4GA1UEChMHVW5rNm93bjEQMA4GA1UECXMHVW5r
Nm93bjJERMA8GA1UEAxMIMzQyNzE5MDMwgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAo
GBALIOqFm6jxZxtXkoh3phzLlhtmCjF99aKryMLSq2uEXO8tGp8vhJpAOEFgyXf502
fSZEtAgSqb0Ngj3nikJdZmV6A7rQrlxfRp68ChHdmv+ApESxJNQ6BMrFFLrF2YNQ6q
MxGg8NAuyeAa15oe+gir4sibLCMB16xvcpr9JsjCrtAgMBAAGgADANBgkqhkiG9w0B
AQQFAA0BgQAhKdHqAtmfUe6cozi8T/Extxp04gUATL6w/sf/lcevMNU//uJaA3SRYH
XJ+zJUD+4zE26+5uXEnQzoEZQBUJj5vMJiaL4SDHYN7stjan7VJ8SRyNkL8q10I1w5
AhF7EPLzT3U9cMIsZm84KQteNbWO/hLn5XVKiocAv34v1PQwCg==
-----END NEW CERTIFICATE REQUEST-----

```

3 Submit the certificate request to a CA.

4 In a text editor, open the WebLogic Server startup script for the domain you are working with and add the following values to the JAVA_OPTIONS environment variable:

```

-Djava.protocol.handler.pkgs=com.sun.net.ssl.internal.www.protocol
-Djavax.net.ssl.trustStore="c:/your_location/your_keystore.jks"
-Djavax.net.ssl.keyStore="c:/your_location/your_keystore.jks"
-Djavax.net.ssl.keyStorePassword=your_password

```

Note: The global system property `javax.net.ssl.keyStore` specifies the keystore. This ensures that all applications running in the JVM use the same keystore. If there is more than one client certificate in the keystore, the JSSE implementation selects an appropriate certificate to use when the session is established. There is no way to override this selection in the Application Server Runtimes, so Entrust recommends that you use a keystore with only one client certificate.

- 5 Add the following line to the WebLogic Server startup script:

```
set PRE_CLASSPATH=c:\jsse\lib\jsse.jar (assuming you have installed
the JSSE in c:\jsse, this line ensures that jsse.jar appears on the classpath
before weblogic.jar)
```

To prepare Application Server Runtime for WebSphere to use server-authenticated SSL

- 1 Download and unpack Sun Microsystems' JSSE distribution.

Version 1.0.3_01 of the JSSE is available at <http://java.sun.com/products/jsse/index-103.html>.

- 2 Copy `jsse.jar` from the JSSE `lib` directory to `<WAS_HOME>\lib`.

The default location of `WAS_HOME` is `C:\Program Files\WebSphere\AppServer` on Windows operating systems, and `/opt/WebSphere/AppServer` on Solaris operating systems.

Note: Do not copy the two other jar files (`jcert.jar` and `jnet.jar`) in the JSSE `lib` directory because they contain classes that interfere with WebSphere Application Server.

Using Java `keytool`, import the Web server CA certificate into a keystore. For example:

```
>keytool -import -alias <keystore alias> -file <CA cert file name>
-keystore <keystore name> -storepass <keystorepassword>
```

```
Owner: OU=Entrust PKI Demonstration Certificates, O=Entrust, C=US
Issuer: OU=Entrust PKI Demonstration Certificates, O=Entrust, C=US
Serial number: 3b992f45
Valid from: Fri Sep 07 16:04:13 EDT 2001 until: Tue Sep 07
16:34:13 EDT 2021
Certificate fingerprints:
    MD5: 7D:29:12:AE:43:68:0D:17:03:3A:05:53:1A:5A:6F:03
    SHA1:
13:38:0A:7A:0E:65:56:07:D2:33:1B:37:82:36:B5:48:68:F3:0F:EC
Trust this certificate? [no]: yes
```

Certificate was added to keystore

3 Set the following system properties:

```
-Djava.protocol.handler.pkgs=com.ibm.net.ssl.www.protocol
-Djavax.net.ssl.trustStore=c:/your_location/your_keystore.jks
-Djavax.net.ssl.keyStore=c:/your_location/your_keystore.jks
-Djavax.net.ssl.keyStorePassword=your_password
```

In the Administrative Console's left navigation pane, click **Application Servers** > **server1** > **Process Definition** > **Java Virtual Machine** and edit the field called **Generic JVM arguments** to set the system properties described earlier.

Note: Do not use double quotes when typing the system properties in the Generic JVM arguments field. For example,

```
-Djavax.net.ssl.trustStore=c:/your_location/your_keystore.jks
-Djavax.net.ssl.keyStore=c:/your_location/your_keystore.jks
```

The path to your key store should not contain spaces.

The global system property `javax.net.ssl.keyStore` specifies the keystore. This ensures that all applications running in the JVM use the same keystore. If there is more than one client certificate in the keystore, the JSSE implementation selects an appropriate certificate to use when the session is established. There is no way to override this selection in the Application Server Runtimes, so Entrust recommends that you use a keystore with only one client certificate.

4 Edit the configuration file, `configuration.global.xml`, to update the URLs that specify the three components comprising the Entrust Identification Server.

Refer to the section called ["Configuration" on page 55](#) for information about editing the configuration file. Under the `<gaIdentificationServer>` setting, change `http://...` to `https://...`

For example, change

```
http://hostname.entrust.com:50002/ies/CredentialCollector
```

to

```
https://hostname.entrust.com:50002/ies/CredentialCollector
```

To prepare Application Server Runtime for WebSphere to use client-authenticated SSL

1 Create a key pair using Java keytool.

Note: If you are using an Entrust PKI, add a user before moving to the next step. This will give you the reference number to use in the client name.

For example:

```
>keytool -genkey -keyalg RSA -keysize 1024 -keystore <keystore
name> -storepass <keystore password>
```

```
What is your first and last name?
```

```
[Unknown]: 34271903
```

```
What is the name of your organizational unit?
```

```
[Unknown]:
```

```
What is the name of your organization?
```

```
[Unknown]:
```

```
What is the name of your City or Locality?
```

```
[Unknown]:
```

```
What is the name of your State or Province?
```

```
[Unknown]:
```

```
What is the two-letter country code for this unit?
```

```
[Unknown]:
```

```
Is <CN=34271903, OU=Unknown, O=Unknown, L=Unknown, ST=Unknown,
C=Unknown> correct?
```

```
[no]: yes
```

```
Enter key password for <mykey>
```

```
(RETURN if same as keystore password):
```

2 Generate a certificate request using Java keytool. For example:

```
>keytool -certreq -keystore <keystore name> -storepass <keystore
password>
```

```
-----BEGIN NEW CERTIFICATE REQUEST-----
```

```
MIIBrTCCARYCAQAwTEQMA4GA1UEBhMHVW5rbm93bjEQMA4GA1UECBMHVW5rbm93bjEQMA4GA1UEBxMHVW5rbm93bjEQMA4GA1UEAxMIMzQyNzE5MDMwZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBALIOqFm6jxZxtXkoh3phzLlhtmCjF99aKryMLSq2uEXO8tGp8vhJpAOEFgyXf502fSZEtAgSqB0ngj3nikJdZmV6A7rQrlxfRp68ChHdmv+ApESxJNQ6BMrFFLrF2YNQ6qMxGg8NAuyeAa15oe+gir4sibLCMB16xvcpr9Js jCRtAgMBAAGgADANBgkqhkiG9w0BAQQFAAOBgQAhKdHqAtmfUe6cozi8T/Extxp04gUATL6w/sf/lcevMNU//uJaA3SRYHXJ+zJUd+4zE26+5uXEnQzoEZQBUJj5vMJiaL4SDHYN7sTjan7VJ8SRyNKL18q10Ilw5AhF7EPLzT3U9cMIsZm84KQteNbWO/hLn5XVKiocAv34v1PQwCg==
```

```
-----END NEW CERTIFICATE REQUEST-----
```

3 Submit the certificate request to a CA.

4 Set the following JVM system properties:

```
-Djavax.net.ssl.keyStore=c:/your_location/your_keystore.jks
```

```
-Djavax.net.ssl.keyStorePassword=your_password
```


The path to your key store should not contain spaces

- 5** Add the following line to the WebLogic Server startup script:

```
set PRE_CLASSPATH=c:\jsse\lib\jsse.jar (assuming you have installed  
the JSSE in c:\jsse, this line ensures that jsse.jar appears on the classpath  
before weblogic.jar)
```


Chapter 8

Uninstalling Application Server Runtimes

This chapter describes how to remove Entrust Application Server Runtimes from your computer.

Topics in this section:

- [“Preparing to uninstall Application Server Runtimes” on page 106](#)
- [“Uninstalling on Windows” on page 109](#)
- [“Uninstalling on Solaris” on page 110](#)

Preparing to uninstall Application Server Runtimes

The Entrust Application Server Runtimes installer application creates an uninstaller as part of the installation process. Before running the uninstaller, read the information in this section.

This section describes specific steps related to WebLogic Server and to WebSphere Application Server to prepare your computer for the removal of Entrust Application Server Runtimes. Complete the procedures in this subsection to ensure that WebLogic Server and WebSphere Application Server and their deployed applications operate correctly after you remove the Application Server Runtime software.

To prepare to uninstall Application Server Runtime for WebLogic

- 1** Remove the Entrust GetAccess Identity Verification servlet filter from all Web applications.
Use a text editor to edit the Web application deployment descriptor directly, or refer to the WebLogic Server documentation (<http://e-docs.bea.com/wls/docs70/index.html>) to use the Administration Console to reconfigure the deployment descriptor for your Web application.
- 2** Remove method permissions specifying Entrust GetAccess roles from all EJBs.
You can do this using a text editor to edit the EJB deployment descriptor directly, or you can use the WebLogic Server Administration Console.
- 3** Remove the Entrust security realm from each of your WebLogic Server domains.

For each domain:

- Start the domain's Administration server
- Start the Administration Console
- Set the default security realm to `myrealm` (or any realm other than the Entrust realm)
- Locate and run the `removeEntrustRealm` script file in your Application Server Runtimes installation directory.

```
<App Server Runtimes root>\bin\weblogic\removeEntrustRealm.cmd
```

```
<App Server Runtimes root>/bin/weblogic/removeEntrustRealm.sh
```

The script file should display output similar to the following:

```
C:\Program
Files\Entrust\AppServerRuntimes\bin\weblogic>removeEntrustRealm
WebLogic home: c:\bea\weblogic700
```

```
Domain administration server listen address: t3://localhost:7001
Enter username for domain admin: wlsusr1
Enter password for domain admin: *****
```

```
This will un-install the Entrust Realm from your WebLogic domain.
This action may take a few minutes. Are you sure you want to
continue (Y/N)? y
```

```
Inspecting the domain...
Removing EntrustRealm...
```

```
Done!
```

Note: Make sure that you remove the Entrust security realm from all of your WebLogic domains before performing the Application Server Runtime uninstallation procedure. If you do not take this step, the WebLogic servers will not start after you remove the Application Server Runtime software.

To prepare to uninstall Application Server Runtime for WebSphere

- 1 Locate and run the `removeEntrustRealm` script file in your Application Server Runtimes installation directory.

```
<App Server Runtimes root>\bin\websphere\removeEntrustRealm.cmd
```

```
<App Server Runtimes root>/bin/websphere/removeEntrustRealm.sh
```

The script file should display output similar to the following:

```
C:\Program
Files\Entrust\AppServerRuntimes\bin\websphere>removeEntrustRealm
```

```
WAS_HOME=C:\Program Files\WebSphere\AppServer
ASR_HOME=C:\Program Files\entrust\AppServerRuntimes
```

```
Found cell: <hostname>
The following files MAY be modified:
Found: C:\Program
Files\WebSphere\AppServer\config\cells\<hostname>\
security.xml
Found: C:\Program
Files\WebSphere\AppServer\config\cells\<hostname>\
admin-authz.xml
```

```
Continue and modify these configuration files (y/n) [y]: y
```

```
Disable global security... done
Reset Custom Registry to WebSphere defaults... done
```

```
Reset Active User Registry to default LocalOSUserRegistry... done
Reset Active Authentication Mechanism to default SWAM... done
Disable Trust Association ... done
Reset default Trust Association Interceptor... done
Remove Entrust Authorization Table ... done
```

```
Updated: C:\Program
Files\WebSphere\AppServer\config\cells\\
security.xml
```

```
** Operations completed successfully. **
Global security is now disabled.
Press any key to continue . . .
```

2 Enable or disable Java 2 Security as required.

Refer to the note in [Step 3](#) of the procedure called “[To configure Global Security settings for WebSphere Application Server](#)” on page 48.

To turn off or to turn on Java 2 Security, in the Administrative Console left pane, click **Security > Global Security**. On the **Global Security** page, under the **Configuration** tab, **General Properties**, clear or select the **Enforce Java 2 Security** checkbox as required.

3 Amend any EJB deployment descriptors that are affected by the removal of the Entrust security realm.

The `removeEntrustRealm` script turns off global security, but does not remove EJB method permissions from the EJB deployment descriptors. If you want to remove these permissions, open the WebSphere Application Assembly tool to make the changes, or use a text editor to edit the files directly.

Uninstalling on Windows

The following procedure describes the steps you should take to remove the Application Server Runtime software from your computer.

To uninstall on Windows

- 1** Refer to the section entitled “[Preparing to uninstall Application Server Runtimes](#)” on page 106 and note the actions you should take to prepare for the removal of the Application Server Runtime software.
- 2** From the **Start** menu click **Settings > Control Panel** and open **Add or Remove Programs**.
- 3** In the left pane, click **Add or Remove Programs** and select **Entrust Application Server Runtimes 7.0**.
- 4** Click **Change/Remove** and follow the instructions in each dialog box.
- 5** Click **Finish** when the last dialog box displays a message saying that the uninstaller application has successfully uninstalled the Application Server Runtime software.
- 6** Restart your Web application server.

Note: You can invoke the uninstaller application from the command line with the following command:

```
<path to uninstaller executable file>\uninstaller.exe -console
```

Uninstalling on Solaris

The following procedure describes the steps you should take to remove the Application Server Runtime software from your computer.

To uninstall on Solaris

- 1** Refer to the section entitled “[Preparing to uninstall Application Server Runtimes](#)” on page 106 and note the actions you should take to prepare for the removal of the Application Server Runtime software.
- 2** Locate the uninstaller executable file.
The uninstaller executable file is in
`<app_server_installation_directory>/_uninst/uninstaller.bin`
- 3** Run the application.
- 4** Follow the instructions in each dialog box.
- 5** Click **Finish** when the last dialog box displays a message saying that the uninstaller application has successfully uninstalled the Application Server Runtime software.
- 6** Restart your Web application server.

Note: You can invoke the uninstaller application from the command line with the following command:

```
<path to uninstaller executable file>/uninstaller.bin -console
```

Chapter 9

Troubleshooting Application Server Runtimes

This chapter provides information and tips about troubleshooting problems you might encounter installing, uninstalling, and using Entrust Application Server Runtimes.

Topics in this chapter:

- [“Overview” on page 112](#)
- [“Troubleshooting installation problems” on page 113](#)
- [“Error logging” on page 115](#)
- [“Error messages” on page 118](#)

Overview

This chapter provides information to help you solve problems you might encounter when you install and use Entrust Application Server Runtimes. The chapter includes a section with installation-related troubleshooting advice, a section that describes the Application Server Runtime log files (where you should look for error messages), and a section with tables listing error messages, their causes, and possible solutions.

The Application Server Runtimes depend upon Entrust Identification Server, which is a set of servlets deployed on an Entrust GetAccess Infrastructure. The “Troubleshooting” chapter of the *Entrust Identification Server 7.0 and Entrust Entitlements Server 7.0 Administration Guide* and the *Entrust GetAccess 7.0 Troubleshooting Guide* are additional sources of problem solving information.

If you are unable to solve your problem after reading this chapter, refer to the section called [“Getting help” on page 4](#) for information about how to obtain technical support.

Troubleshooting installation problems

This section contains information about problems related to the installation of Application Server Runtimes.

Removing remnants of a previous installation

The Entrust Application Server Runtimes installer application will not allow you to install more than one instance of the Application Server Runtime software on the same computer. If the installer displays the message “A previous install of this product has been found on this system. You must uninstall the previous installation before running this installer. Click Next to exit the installer.”, when you try to install the software, refer to the chapter entitled “[Uninstalling Application Server Runtimes](#)” on page 105 for the steps you should take to remove the product.

If the installer continues to display the error message after you have taken these steps, there might be remnants of a previous failed installation or uninstallation on your computer. Follow the steps set out in this section to remove these remnants and to prepare your computer for a fresh installation of the Application Server Runtimes.

To recover from a failed installation on Windows

- 1 In the %WINDIR% directory, usually, C:\WINDOWS or C:\WINNT, locate the file called `vpd.properties`.
- 2 Open the `vpd.properties` file in a text editor, search for, and delete all lines containing the string, `AppServerRuntimes`
- 3 Locate and delete the Application Server Runtimes installation folder, usually C:\Program Files\Entrust\AppServerRuntimes
- 4 Refer to the section entitled “[Installing Application Server Runtimes](#)” on page 30 of the Installation chapter in this guide and follow the instructions to reinstall the Application Server Runtime software.

To recover from a failed installation on Solaris — root user

- 1 Locate the `/var/sadm/pkg` directory.
- 2 Delete the subdirectory called `ENTUASR7`.
- 3 Locate and delete the Application Server Runtimes installation directory, usually `/opt/entrust/AppServerRuntimes`.

- 4 Refer to the section entitled [“Installing Application Server Runtimes” on page 30](#) of the Installation chapter in this guide and follow the instructions to reinstall the Application Server Runtime software.

To recover from a failed installation on Solaris — non root user

- 1 In the user's \$HOME directory, locate the file called `vpd.properties`.
- 2 Open the `vpd.properties` file in a text editor, search for, and delete all lines containing the string, `AppServerRuntimes`
- 3 Locate and delete the Application Server Runtimes installation directory, usually `/opt/entrust/AppServerRuntimes`.
- 4 Refer to the section entitled [“Installing Application Server Runtimes” on page 30](#) of the Installation chapter in this guide and follow the instructions to reinstall the Application Server Runtime software.

Redirecting temporary installation files

The installer application writes files to a temporary directory. If the temporary directory is full, or if the system's `temp` directory is not defined, you might receive an error message telling you to use the `-is:tempdir` command. In this case run the installer application as follows:

- On Windows

```
AppServerRuntimes_setupwin32.exe -is:tempdir <"Path to temporary directory">
```
- On Solaris

```
AppServerRuntimes_setupsolarisSparc.bin -is:tempdir <"Path to temporary directory">
```

Error logging

If you encounter problems with an Application Server Runtime, read the error log file with a text editor. The Application Server Runtime logs errors in file called `asr.log` in the `AppServerRuntimes\logs` directory.

The `asr.log` file is the current error log, created daily, but you can read older log entries in dated log files. Log files written earlier than the current date have the following name format:

```
asr.log.YYYY-MM-DD
```

You can also read WebSphere Application Server error logs in the `SystemOut.log` file

Customizing the log file entries

You can customize the error level that the Application Server Runtime writes to the log file to meet your requirements for the depth of error logging information you need. The following table lists the error logging levels in order of increasing severity.

Table 3: Error logging levels

Logging level	Description
TRACE	Messages useful for tracing events — Updating the <code>server-list</code> cache, for example.
DEBUG	Messages generated by events in the application's code — successful connection information for example. Intended for use by developers.
INFO	Informative messages — Initializing Entrust Trust Association Interceptor, for example.
WARNING	Warnings of recoverable errors.
ERROR	Error messages that are not fatal — Java exceptions, for example.
ALERT	Messages for the administrator — too many login attempts, for example.
FATAL	Fatal error that have occurred — unrecoverable exceptions, for example.

Note: If you set error logging to a particular level, all logging levels with a severity greater than the level you set are written to the log. For example, the default logging level is INFO, but errors with log levels WARNING, ERROR, ALERT, and FATAL are logged as well.

Set the logging level by editing the Application Server Runtimes configuration file, `AppServerRuntimes\config\configuration.global.xml`. Refer to the section called ["Configuration" on page 55](#) for information about editing the configuration file.

Log file format

The log file contains a header section followed by a section that contains the log file entries.

Log file header

The top of the log file contains a header that indicates environment parameters such as the operating system and version. The log file header has the following format:

```
Entrust Application Server Runtime for IBM WebSphere
Version: <product version>
Logger version: <logger version>

Operating system: <name and version>
System architecture: <architecture information>
JRE Vendor: <name of JRE vendor>
JRE Version: <JRE version>
JVM Vendor: <name of JVM vendor>
JVM Version: <JVM version>
Classpath in use: <classpath>
User: <username>
```

Log file entries

Log file entries have the following format:

```
[date/time][level][component][classname] message
```

where:

- `date/time` is the local host time in ISO 8601 format: YYYY-MM-DD HH:MM:SS (+HHMM or -HHMM indicates that the local time being used is HH hours and MM minutes ahead of or behind Coordinated Universal Time (UTC), respectively).
- `level` is the logging level for that entry.

- `component` is the name of the Application Server Runtimes component that originated the log entry:
- `classname` is always blank
- `message` is the log entry description

The following is an example of a typical log file entry:

```
[2003-05-02 09:04:38-0700][INFO ][GaAuthorizationTable][]  
WebSphere role category ID: webspHERE.
```

Error messages

This section describes error messages, their causes, and possible solutions. Application Server Runtimes displays error messages either in message boxes during the installation and uninstallation processes, or writes them to a log file. Refer to the section called [“Error logging” on page 115](#) for information about the error log file.

Installer error messages

The following table lists error messages associated with the installer and uninstaller applications for Entrust Application Server Runtimes.

Table 4: Installer error messages

Error message	Description
A suitable JVM could not be found. Please run the program again using the option <code>-is:javahome <JAVA HOME DIR></code>	<p>Cause: The installer application requires a JRE version between 1.3.0 and 1.4.1. It searches the Windows registry or the <code>/usr/java</code> directory on UNIX for the default JRE. If it cannot find a suitable JRE, it displays this error message.</p> <p>Solution: Run the installer with the command-line option <code>-is:javahome</code> to point to a suitable JRE other than the default.</p>
A previous install of this product has been found on this system. You must uninstall the previous installation before running this installer. Click Next to exit the installer.	<p>Cause: The installer application does not support multiple installations of the Application Server Runtimes.</p> <p>Solution: Uninstall the previously installed version and reinstall Application Server Runtime software.</p> <p>Refer to “Uninstalling Application Server Runtimes” on page 105.</p> <p>Cause: A previous installation has failed.</p> <p>Solution: Refer to “Removing remnants of a previous installation” on page 113.</p>
Please enter a valid value	<p>Cause: The Host name field in the dialog box is empty.</p> <p>Solution: Type a valid host name to continue the installation.</p>
You must specify a valid number	<p>Cause: The Port number field in the dialog box contains characters that are not numbers.</p> <p>Solution: Type a valid numerical value in the Port number field.</p>

Table 4: Installer error messages

Error message	Description
WebLogic (or WebSphere) Win32 Registry Not Found. Click "Next" to go back to the App Server Selection Panel	<p>Cause: The installer application cannot find a correct registry value in the Windows registry for the specified application server.</p> <p>Solution: Return to the previous dialog box and select the application server you are using.</p>
Please type or select a valid WebLogic Server installation directory.	<p>Cause: The installer application cannot find the specified directory for WebLogic Home.</p> <p>Solution: Type a valid directory for the location of WebLogic Home before the installation can continue.</p> <p>The default location on Windows operating systems is C:\bea\weblogic700, and /opt/bea/weblogic700 on Solaris operating systems.</p>
Please type or select a valid WebSphere Application Server installation directory.	<p>Cause: The installer application cannot find the specified directory for WebSphere Home.</p> <p>Solution: Type a valid directory for the location of WebSphere Home before the installation can continue.</p> <p>The default location on Windows operating systems is C:\Program Files\websphere\AppServer, and /opt/websphere/AppServer on Solaris operating systems.</p>
Please type or select a valid WebLogic Server domain.	<p>Cause: The installer application cannot find the specified WebLogic Domain to configure.</p> <p>Solution: Type a valid location for the WebLogic Domain you want to configure.</p> <p>For example, C:\bea\user-projects\mydomain on Windows operating systems, and /opt/bea/user_projects/mydomain on Solaris operating systems.</p>
The WebLogic startup script is not found under the WebLogic domain to configure. You can click Back to select another WebLogic domain, or click Next to continue the install and modify the WebLogic startup script after the install completes.	<p>Cause: The installer application cannot locate startup scripts for the WebLogic domain you selected.</p> <p>Solution: Click Back to select another domain or continue installing and modify startup scripts manually.</p> <p>Refer to "Post installation steps — WebLogic Server" on page 39 for instructions.</p>

Application Server Runtime error messages

This section contains tables listing error messages that might occur when running Application Server Runtime for WebLogic Server and Application Server Runtime for WebSphere Application Server.

Application Server Runtime for WebLogic error messages

When an error occurs, the Application Server Runtime for WebLogic writes one or more of the error messages listed in [Table 5](#) to the log file, `asr.log`.

Table 5: Application Server Runtime for WebLogic error messages

Error message	Description
Ignoring illegal authentication control flag {0}	<p>Cause: This error occurs if the authentication control flag for the Application Server Runtime authentication provider is not set to SUFFICIENT.</p> <p>Solution: Recreate the Entrust security realm. Refer to the procedure called "To create an Entrust realm using the createEntrustRealm script" on page 39.</p>
Authentication failed — userid not supplied	<p>Cause: The authentication failed because a user name was not specified.</p> <p>Solution: Ensure that you specify a valid Entrust GetAccess user name.</p>
Authentication failed — password not supplied for user {0}	<p>Cause: The authentication failed because a password was not specified.</p> <p>Solution: Ensure that you specify a valid password for the user.</p>
Invalid authentication token found in the request	<p>Cause: The authentication token found in the HTTP request was not valid.</p> <p>Solution: Ensure that you log in to Entrust GetAccess before attempting to gain access to the servlet or JSP protected by the Application Server Runtime servlet filter.</p>
Identification Server error	<p>Cause: The Entrust Identification Server has reported an error.</p> <p>Solution: Review the Identification Server error log file for more information. The <code>ies.log</code> file is in the <code>IdEntitlementsServer\logs</code> directory.</p>

Table 5: Application Server Runtime for WebLogic error messages

Error message	Description
Internal error — authentication token not found in the call-back	<p>Cause: This is an internal error.</p> <p>Solution: Please contact Entrust Customer Support. Refer to “Getting help” on page 4.</p>
Internal error — logout not expected to be called!	<p>Cause: This is an internal error.</p> <p>Solution: Please contact Entrust Customer Support. Refer to “Getting help” on page 4.</p>

Application Server Runtime for WebSphere error messages

When an error occurs, the Application Server Runtime for WebSphere writes the error messages listed in [Table 6](#) to the log file, `asr.log`.

Table 6: Application Server Runtime for WebSphere error messages

Error message	Description
CustomRegistry method "mapCertificate" is not supported	<p>Cause: The Application Runtime for WebSphere does not support client authentication by way of SSL/TLS directly to the WebSphere Application Server.</p> <p>Solution: Use a Web server with the Entrust GetAccess Certificate PAAM. Refer to the <i>Entrust Identification Server 7.0 and Entrust Entitlements Server 7.0 Administration Guide</i> for details.</p>
<p>CustomRegistry method "method-name" is not supported</p> <p>Where <code>method-name</code> can be any of the following:</p> <pre> getUsers() getUsersForGroup() isValidUser() getGroups(String pattern) getGroupDisplayName() getGroupSecurityName() isValidGroup() getUsers(String) </pre>	<p>Cause: With the Application Server Runtime for WebSphere, user and role management is accomplished under Entrust GetAccess. User, group, and role management tasks performed with WebSphere tools, such as the Administrative Console and the Application Assembly Tool, are no longer necessary and the Application Server Runtime for WebSphere does not support them. However, you can still use the WebSphere Administrative Console to add a console user.</p> <p>Solution: Perform user and role management tasks in Entrust GetAccess.</p>

Table 6: Application Server Runtime for WebSphere error messages

Error message	Description
Required property "gaConfigurationFile" for Custom Registry is not specified	<p>Cause: The <code>gaConfigurationFile</code> custom property of the Custom User Registry is not set.</p> <p>Solution: Recreate the Entrust security realm. Refer to the procedure called "To configure an Entrust security realm" on page 46.</p>
Required property "gaConfigurationMetadataFile" for Custom Registry is not specified	<p>Cause: The <code>gaConfigurationMetadataFile</code> custom property of the Custom User Registry is not set.</p> <p>Solution: Recreate the Entrust security realm. Refer to the procedure called "To configure an Entrust security realm" on page 46.</p>
Property file <code>init()</code> is not supported	<p>Cause: Some versions of WebSphere Application Server older than version 5.0 support the passing of properties to the <code>TrustAssociationInterceptor</code> by way of property files.</p> <p>Solution: Application Server Runtime for WebSphere supports only WebSphere Application Server version 5.0.</p>
Unknown userid: {0}	<p>Cause: You are trying to add a user as a Console User in the WebSphere Administrative Console, but the <code>userID</code> you specified was unknown to Entrust GetAccess.</p> <p>Solution: Ensure that you specify a valid Entrust GetAccess user when you add a Console User in the Administrative Console.</p> <p>Cause: This error might occur if you are attempting to gain access to an EJB by way of a Web resource (a servlet or a JSP).</p> <p>Solution: Verify that the Entrust Trust Association Interceptor is properly configured and loaded. The log file will display a message similar to the following:</p> <pre>[date/time][INFO][GaTrustAssociationInterceptor] [] Initializing Entrust Trust Association Interceptor</pre> <p>Refer to the procedure called "To configure an Entrust security realm" on page 46.</p>

Table 6: Application Server Runtime for WebSphere error messages

Error message	Description
Error parsing principal: {0} getSidFromPrincipal	<p>Cause: This error might occur if the Trust Association Interceptor is not functioning.</p> <p>Solution: Verify that the Entrust Trust Association Interceptor is properly configured. The log file will display a message similar to the following:</p> <pre>[date/time][INFO][GaTrustAssociationInterceptor] [] Initializing Entrust Trust Association Interceptor</pre> <p>Refer to the procedure called “To configure an Entrust security realm” on page 46.</p>
Error parsing principal: {0} getUidFromPrincipal	<p>Cause: This error might occur if the Application Server Runtime for WebSphere cannot communicate with the AttributeAuthority servlet of the Entrust Identification Server.</p> <p>Solution: Ensure that the Entrust Identification Server is running and available for use by the Application Server Runtime for WebSphere.</p>

createEntrustRealm script error messages

The error message listed in the following table can occur when you run the createEntrustRealm script.

Table 7: createEntrustRealm script error messages

Error message	Description
Error creating security realm: EntrustRealm java.net.ConnectException: t3://localhost:7001: Destination unreachable	<p>Cause: The host name localhost is not recognized.</p> <p>Solution: Edit the createEntrustRealm script, (createEntrustRealm.cmd or createEntrustRealm.sh) and replace localhost with the fully qualified host name.</p>

Web application server error messages

This section lists error messages that relate to the Application Server Runtimes, but that are displayed by your Web application server.

Table 8: Web application server error messages

Error message	Description
Unable to process login. Please check User ID and password and try again.	<p>Cause: WebSphere Administrative Console displays this error message if you have previously accessed an Entrust GetAccess-protected URL in this browser session, and the Entrust GetAccess user does not have WebSphere administrator privileges.</p> <p>Solution: Close and reopen your browser to delete your Entrust GetAccess cookie, then login again to the WebSphere Administrative Console. Avoid performing WebSphere administrative operations in the same browser session you use to access Entrust GetAccess-protected URLs.</p>