Oracle® Access Manager

Integration Guide

10*g* (10.1.4.2)

E10356-01

August 2007

Explains how to set up Oracle Access Manager to run with other Oracle products, for example, OracleAS Single Sign-On, and with third-party products, for example, WebLogic SSPI, Siebel 7, and IBM Websphere.

ORACLE®

Oracle Access Manager Integration Guide 10*g* (10.1.4.2)

E10356-01

Contributing Author: Nina Wishbow, Norman Bock, Gail Tiberi

Contents

4 Integrating with Oracle Application Servers

5 Federated Single Sign-On Using Oracle Identity Federation

6 Integrating Oracle Identity Management

11 Integrating with IBM WebSphere

12 Integrating Plumtree Corporate Portal

13 Integrating mySAP Applications

14 Integrating the RSA SecurID Authentication Plug-In

15 Integrating Smart Cards

18 Integrating With ASP.NET

19 Integrating Authorization Manager Services

20 Enabling Impersonation with the Access System

Preface

This Integration Guide provides information about integrating Oracle Access Manager with third-party applications servers and portals.

> **Note:** Oracle Access Manager was previously known as Oblix *NetPoint*.

This Preface covers the following topics:

Audience

This guide is intended for administrators who are responsible for integrating their product with Oracle Access Manager.

This guide assumes that you are familiar with your LDAP directory and Web servers, Oracle Access Manager, and the product that you are integrating.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible, with good usability, to the disabled community. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Accessibility standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For more information, visit the Oracle Accessibility Program Web site at

`http://www.oracle.com/accessibility/`

Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an

otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

TTY Access to Oracle Support Services

Oracle provides dedicated Text Telephone (TTY) access to Oracle Support Services within the United States of America 24 hours a day, seven days a week. For TTY support, call 800.446.2398.

Related Documents

For more information, see the following documents in the Oracle Access Manager Release 10*g* (10.1.4.0.1) documentation set:

- *Oracle Access Manager Introduction*—Provides an introduction to Oracle Access Manager, a road map to the manuals, and a glossary of terms.
- *Oracle Application Server Release Notes*—Read these for the latest Oracle Access Manager updates. The release notes are available with the platform-specific documentation. The most current version of the release notes is available on Oracle Technology Network at:
  `http://www.oracle.com/technology/documentation`
- *Oracle Access Manager Patchset Notes Release 10.1.4 Patchset 1 (10.1.4.2.0) For All Supported Operating Systems.* It provides the system requirements and instructions needed to install or de-install the Patchset itself, a list of known issues related to the patchset, a list of the platform-specific bugs fixed in this Oracle Access Manager Patchset.
- *Oracle Access Manager List of Bugs Fixed Release 10.1.4 Patchset 1 (10.1.4.2.0)* . It supplements the Patchset notes document for this release. It provides a list of all generic (common to all operating systems) Oracle Access Manager bugs that have been fixed in this Patchset, sorted by component.
- *Oracle Access Manager Installation Guide*—Describes how to install and set up the Oracle Access Manager components.
- *Oracle Access Manager Upgrade Guide*—Explains how to upgrade earlier releases to the latest major Oracle Access Manager release.
- *Oracle Access Manager Identity and Common Administration Guide*—Explains how to configure Identity System applications to display information about users, groups, and organizations; how to assign permissions to users to view and modify the data that is displayed in the Identity System applications; and how to configure workflows that link together Identity application functions, for example, adding basic information about a user, providing additional information about the user, and approving the new user entry, into a chain of automatically performed steps. This book also describes administration functions that are common to the Identity and Access Systems, for example, directory profile configuration, password policy configuration, logging, and auditing.
- *Oracle Access Manager Access System Administration Guide*—Describes how to protect resources by defining policy domains, authentication schemes, and authorization schemes; how to allow users to access multiple resources with a

single login by configuring single- and multi-domain single sign-on; and how to design custom login forms. This book also describes how to set up and administer the Access System.

- *Oracle Access Manager Deployment Guide*—Provides information for people who plan and manage the environment in which Oracle Access Manager runs. This guide covers capacity planning, system tuning, failover, load balancing, caching, and migration planning.
- *Oracle Access Manager Customization Guide*—Explains how to change the appearance of Oracle Access Manager applications and how to control operation by making changes to operating systems, Web servers, directory servers, directory content, or by connecting CGI files or JavaScripts to Oracle Access Manager screens. This guide also describes the Access Manager API and the authorization and authentication plug-in APIs.
- *Oracle Access Manager Developer Guide*—Explains how to access Identity System functionality programmatically using IdentityXML and WSDL, how to create custom WebGates (known as AccessGates), and how to develop plug-ins. This guide also provides information to be aware of when creating CGI files or JavaScripts for Oracle Access Manager.
- *Oracle Access Manager Integration Guide*—Explains how to set up Oracle Access Manager to run with third-party products such as BEA WebLogic, Siebel 7, and IBM Websphere.
- *Oracle Access Manager Schema Description*—Provides details about the schema.
- *Oracle Access Manager Configuration Manager Installation and Administration Guide*—Provides information about pushing configuration data changes from one Oracle Access Manager 10*g* (10.1.4.0.1), or Oracle COREid Release 7.0.4, deployment to another. For example, when pushing changes from a development deployment to a pre-production deployment. Included are considerations, prerequisites, and step-by-step instructions to help ensure your success.

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

What's New in Oracle Access Manager?

This section introduces new features of Oracle Access Manager 10*g* (10.1.4.0.1) and provides pointers to additional information in this book. New features information from previous releases is also retained to help those users migrating to the current release.

The following sections describe the new features and changes in Oracle Access Manager that are described in this book:

- Product and Component Name Changes
- Supported Integrations
- Updates to the OracleAS Single Sign-On Integration
- Updates to the Oracle Identity Federation Integration
- Updates to the Siebel 7 Integration
- Updates to the SAP Integration
- Updates to the RSA Securid Integration
- Updates to the WebLogic Integration
- Updates to the WebSphere Integration
- Updates to the Plumtree Integration
- Configuring Single Sign-On with Oracle Identity Management
- Configuring Impersonation
- Configuring Single Sign-On with Lotus Domino

Note: For a comprehensive list of new features and functions in Oracle Access Manager 10*g* (10.1.4.0.1), and a description of where each is documented, see the chapter on What's New in Oracle Access Manager in the *Oracle Access Manager Introduction*.

Product and Component Name Changes

The original product name, Oblix NetPoint, has changed to Oracle Access Manager. Most component names remain the same. However, there are several important changes that you should know about, as shown in the following table:

Item	Was	Is
Product Name	Oblix NetPoint Oracle COREid	Oracle Access Manager
Product Name	Oblix SHAREid NetPoint SAML Services	Oracle Identity Federation
Product Name	OctetString Virtual Directory Engine (VDE)	Oracle Virtual Directory
Product Release	Oracle COREid 7.0.4	Also available as part of Oracle Application Server 10g Release 2 (10.1.2).
Directory Name	COREid Data Anywhere	Data Anywhere
Component Name	COREid Server	Identity Server
Component Name	Access Manager	Policy Manager
Console Name	COREid System Console	Identity System Console
Identity System Transport Security Protocol	NetPoint Identity Protocol	Oracle Identity Protocol
Access System Transport Protocol	NetPoint Access Protocol	Oracle Access Protocol
Administrator	NetPoint Administrator COREid Administrator	Master Administrator
Directory Tree	Oblix tree	Configuration tree
Data	Oblix data	Configuration data
Software Developer Kit	Access Server SDK ASDK	Access Manager SDK
API	Access Server API Access API	Access Manager API
API	Access Management API Access Manager API	Policy Manager API
Default Policy Domains	NetPoint Identity Domain COREid Identity Domain	Identity Domain
Default Policy Domains	NetPoint Access Manager COREid Access Manager	Access Domain
Default Authentication Schemes	NetPoint None Authentication COREid None Authentication	Anonymous Authentication
Default Authentication Schemes	NetPoint Basic Over LDAP COREid Basic Over LDAP	Oracle Access and Identity Basic Over LDAP
Default Authentication Schemes	NetPoint Basic Over LDAP for AD Forest COREid Basic Over LDAP for AD Forest	Oracle Access and Identity for AD Forest
Access System Service	AM Service State	Policy Manager API Support Mode

All legacy references in the product or documentation should be understood to connote the new names.

Supported Integrations

New integrations are supported for 10*g* (10.1.4.0.1).

- The introduction describes supported integrations for 10*g* (10.1.4.0.1).

  An overview of supported integrations is provided for quick reference. All other chapters in this guide describe implementation details for a specific integration.

  See Also: Chapter 1, "Introduction" on page 1-1.

- This guide now provides information on integrating with Oracle HTTP Server.

  Oracle HTTP Server (OHS) is a Web server extension that identifies Oracle Access Manager Web components that communicate with the OHS.

  See Also: Chapter 2, "Integrating the Apache v1.3 and Oracle HTTP Server" on page 2-1.

- This guide now provides information on integrating with Oracle Virtual Directory.

  The Oracle Virtual Directory combines user data from multiple data sources to create an aggregated virtual directory.

  See Also: Chapter 3, "Integrating the Oracle Virtual Directory" on page 3-1.

- This guide now provides information on integrating with Oracle Identity Management.

  Oracle Identity Management enables provisioning and de-provisioning of user accounts and other IT resources.

  See Also: Chapter 6, "Integrating Oracle Identity Management" on page 6-1.

Updates to the OracleAS Single Sign-On Integration

You can configure single sign-on between the Access System and the OracleAS Single Sign-On server.

- The updated chapter provides information on configuring single sign-on between Oracle Access Manager and Oracle Application Server 10g (OracleAS 10g).

  When you configure single sign-on you also provide identity management functionality across the Web-based applications running on Oracle Application Servers, for example, Oracle e-Business Suite, Oracle Forms, Portals, and other Access System-protected resources.

- Included in this new version is information about the OHS WebGate. Apache WebGate information has been removed.

  See Also: "Integrating with Oracle Application Servers" on page 4-1.

- Information is also provided on configuring an integration with an older version of OracleAS Single Sign-On (10.1.2.0.2).

  See also: "Configuring the Access System for OracleAS Single Sign-On 10.1.2.0.2" on page 4-11

Updates to the Oracle Identity Federation Integration

You can authorize users by querying external authentication systems. This is known as federated authorization.

- When the Access System at a Service Provider site receives a request from a user in a federated environment, it may need to get additional information about the user from the user's Identity Provider. You can configure the Access System to query external Identity Providers for user authorization.

  See Also: "Federated Single Sign-On Using Oracle Identity Federation" on page 5-1.

Updates to the Siebel 7 Integration

Information on configuration of this integration has been updated.

- This chapter has been updated for completeness and accuracy.

  See Also: "Integrating Siebel 7" on page 7-1.

Updates to the SAP Integration

The SAP integration now supports SAP Enterprise Portal 6.0.

- This chapter now explains how SAP Enterprise Portal 6.0 can be protected by the Access System.

  See Also: "Integrating the SAP Enterprise Portal 6.0 and SAP NetWeaver Enterprise Portal SP9" on page 13-8.

Updates to the RSA Securid Integration

Information has been updated for completeness.

See Also: Chapter 14, "Integrating the RSA SecurID Authentication Plug-In".

Updates to the WebLogic Integration

Information has been updated for completeness.

See Also: Chapter 10, "Integrating the Security Provider for WebLogic SSPI".

Updates to the WebSphere Integration

Information in this chapter has been updated.

- The integration with WebSphere Application Server (WAS) 4 is deprecated in this release.

- The information in this chapter for WAS 5 and 6 has been updated for accuracy and completeness.

See Also: Chapter 11, "Integrating with IBM WebSphere".

Updates to the Plumtree Integration

Note that the most recent version of Plumtree Corporate Portal is now known as BEA Aqualogic Interaction.

- The previous integration with Plumtree Corporate Portal is supported and documented in this release.

See Also: Chapter 12, "Integrating Plumtree Corporate Portal".

Configuring Single Sign-On with Oracle Identity Management

This document now contains information on integrating with Oracle Identity Management. Oracle Identity Management is a secure enterprise provisioning system that streamlines the creation and management of user accounts and revocation of user access rights and privileges.

- The integration of Oracle Access Manager with Oracle Identity Management provides a secure Web-based infrastructure for identity management for all customer applications and processes.

See Also: Chapter 6, "Integrating Oracle Identity Management".

Configuring Impersonation

In a Windows environment, all processes and threads execute in a security context. Impersonation is the ability of a thread to execute in a security context that is different from that of the process that owns the thread.

- The information on configuring Windows Impersonation with the Access System has been moved from the *Oracle Access Manager Access System Administration Guide* to this book.

See Also: Chapter 20, "Enabling Impersonation with the Access System".

Configuring Single Sign-On with Lotus Domino

Lotus Domino is a server platform for messaging, collaboration, and applications.

- The information on configuring single sign-on with Lotus Domino has been moved from the *Oracle Access Manager Access System Administration Guide* to this book.

See Also: Chapter 16, "Single Sign-On for Lotus Domino".

1 Introduction

This chapter provides an overview of the Oracle Access Manager integrations for 10*g* (10.1.4.0.1) described in this guide. For an introduction to Oracle Access Manager, see the *Oracle Access Manager Introduction*.

> **Note:** Oracle Access Manager was previously known as Oblix *NetPoint*. However, you may see the name NetPoint in manuals and within the product itself when references are made to specific functions, paths, file names, and so on.

About Oracle Access Manager Integrations

Integrating Oracle Access Manager 10*g* (10.1.4.0.1) with other applications and portals requires some knowledge of both products. This guide provides the details you need to successfully set up Oracle Access Manager for specific applications and portals you may integrate with Oracle Access Manager.

Integrations With Other Oracle Products

The following integrations with other Oracle products are described in this guide:

- **Oracle HTTP Server (OHS):** OHS is a platform in the Oracle security framework that includes the integration of Oracle Access Manager and OracleAS Single Sign-On. See "Integrating the Apache v1.3 and Oracle HTTP Server" on page 2-1 for details.
- **Oracle Virtual Directory (OVD):** This product combines user data from multiple data sources to create an aggregated virtual directory. The virtual directory looks and behaves like any other LDAP directory, and the user does not know that the data has come from heterogeneous sources. See "Integrating the Oracle Virtual Directory" on page 3-1 for details.
- **OracleAS Single Sign-On:** Oracle Application Server Single Sign-On (also referred to as OracleAS Single Sign-On) enables you to use a single user name, password, and optionally a realm ID to log in to all features of the Oracle Application Server as well as to other Web applications. You can enable single sign-on between resources protected by Oracle Access Manager and OracleAS Single Sign-On. See "Integrating with Oracle Application Servers" on page 4-1 for details.
- **Oracle Identity Federation:** Users need to access content on different corporate Web sites. Corporate Web sites need to authenticate and authorize users from different domains that use different security products. The Oracle Identity Federation product addresses these problems. This document describes

integrating with Oracle Identity Federation to enable federated authorization. See "Federated Single Sign-On Using Oracle Identity Federation" on page 5-1 for details.

- **Oracle Identity Management:** Oracle Identity Management is a secure enterprise provisioning system that streamlines the creation and management of user accounts and revocation of user access rights and privileges. Oracle Identity Management automates access rights management, security, and provisioning of IT resources, and connects users to the resources they need to be productive. See "Integrating Oracle Identity Management" on page 6-1 for details.
- **Siebel 7 e-Business Platform:** Siebel 7 is a Web-based suite that combines customer relationship management, partner relationship management and employee relationship management applications. See "Integrating Siebel 7" on page 7-1 for details.
- **Peoplesoft:** PeopleSoft is a Web-based eBusiness application suite that provides human resources, supply chain, CRM, analytics, portal, and other applications. See "Integrating PeopleSoft" on page 8-1 for details.
- **eBusiness Suite:** Oracle E-Business Suite is a comprehensive suite of business applications for the enterprise. See "Integrating Oracle E-Business Suite" on page 9-1 for details.

Integrations with Third-Party Products

The following integrations with third-party products are discussed in this guide:

- **BEA WebLogic Security Service Provider Interface (SSPI):** The Oracle Access Manager Security Provider for WebLogic ensures that only appropriate users and groups can access Oracle Access Manager-protected WebLogic resources to perform specific operations. The Security Provider also enables you to configure single sign-on between Oracle Access Manager and WebLogic resources. See "Integrating the Security Provider for WebLogic SSPI" on page 10-1 for details.
- **IBM WebSphere:** The Oracle Access Manager Connector for WebSphere provides identity management, access control, and single sign-on across J2EE resources and applications developed on the IBM WebSphere platform. See "Integrating with IBM WebSphere" on page 11-1 for details.
- **Plumtree Corporate Portal (now BEA Aqualogic Interaction):** Oracle Access Manager provides identity management, access control, and single sign-on for the Plumtree Corporate Portal. This product is now named BEA Aqualogic Interaction, however, the certified integration is for Plumtree 5.0.4. See Integrating Plumtree Corporate Portal on page 10-1 for details.
- **mySAP:** Integrating Oracle Access Manager with mySAP enables the use of Oracle Access Manager functionality across all mySAP Web-based applications and other Oracle Access Manager-protected enterprise resources and applications. See "Integrating mySAP Applications" on page 13-1 for details.
- **RSA SecurID Authentication:** Oracle Access Manager supports RSA Security features and provides the SecurID authentication plug-in and components needed to integrate a native SecurID authentication scheme into Oracle Access Manager policy domains for Web single sign-on. See "Integrating the RSA SecurID Authentication Plug-In" on page 14-1 for details.
- **Smart Card Authentication:** Oracle Access Manager supports smart card authentication with Active Directory and IIS Web servers using ActivCard Cryptographic Service Provider (CSP) for Windows 2000, ActivCard Gold utilities,

and ActivCard USB Reader v2.0 in homogeneous Windows environments. See "Integrating Smart Cards" on page 15-1 for details.

- **IBM Lotus Domino:** Lotus Domino is a server platform for messaging, collaboration, and applications. You can configure Domino impersonation that is similar to Windows impersonation on IIS. See "Single Sign-On for Lotus Domino" on page 16-1 for details.
- **Microsoft Products:** Oracle Access Manager supports integration with the following Microsoft features and services:
  - **Microsoft SharePoint Portal Server (SPS) 2003:** Oracle Access Manager provides authentication for SPS resources and services, URL level authorization, and single sign-on for seamless navigation between the portal and other protected resources. The SharePoint Portal Server will enforce application-specific authorization polices for capabilities within the SharePoint application and offers the option of controlling access to specific documents. See "Integrating SharePoint Server" on page 17-1 for details.
  - **Microsoft ASP.NET:** Oracle Access Manager supports the ASP.NET component of the Microsoft .NET Framework, which developers can use to build, deploy, and run Web applications and distributed applications. ASP.NET is a set of technologies in the Microsoft .NET Framework that enables the building of Web applications and XML Web services. The Security Connector for ASP.NET supports and enhances native .NET role-based security. See "Integrating With ASP.NET" on page 18-1 for details.
  - **Authorization Manager:** Oracle Access Manager includes a custom authorization plug-in, the Oracle Access Manager AzMan Plug-in to use Authorization Manager services to make authorization decisions for Access Server clients, including WebGates and callers of the Access Manager API. See "Integrating Authorization Manager Services" on page 19-1 for details.
  - **Windows Impersonation:** In a Windows environment, all processes and threads execute in a security context. Impersonation is the ability of a thread to execute in a security context that is different from that of the process that owns the thread. The primary purpose of impersonation is to trigger access checks against a client's identity. See "Enabling Impersonation with the Access System" on page 20-1 for details.

Part I

Integration with Oracle Applications and Middleware

This part provides all the information you need to successfully integrate Oracle Access Manager with Oracle Fusion applications and middleware.

Part I contains the following chapters:

2

Integrating the Apache v1.3 and Oracle HTTP Server

Oracle HTTP Server (OHS) is a Web server extension that identifies Oracle Access Manager Web components that communicate with the OHS. Oracle Access Manager's WebPass, Policy Manager, and WebGate components can be installed on a standalone Oracle HTTP Server on Linux and Windows.

OHS is also an enabling technology for the integration of Oracle Access Manager's single sign-on with OracleAS Single Sign-On.

About the Integration of OHS and Oracle Access Manager

Details of this integration are provided in other documents. For more information about using OHS as the host for the Web components of Oracle Access Manager, see the *Oracle Access Manager Installation Guide*.

For information about using OHS in the integration of Oracle Access Manager and OracleAS Single Sign-On, see "Integrating with Oracle Application Servers" on page 4-1.

3

Integrating the Oracle Virtual Directory

The Oracle Virtual Directory combines user data from multiple data sources to create an aggregated virtual directory.

From the point of view of Oracle Access Manager applications, the virtual directory looks and behaves just like any other LDAP directory, and the Oracle Access Manager user usually does not receive any obvious indications that the data retrieved by Oracle Access Manager has come from heterogeneous sources.

About the Integration of Oracle Virtual Directory and Oracle Access Manager

From the perspective of the target data store owners, the impact of Oracle Virtual Directory is minimal. The data store owners do not relinquish ownership of their data, Oracle Virtual Directory does not reformat the native data structures, and no permanent copies of the original data are maintained by Oracle Virtual Directory.

From the perspective of the administrator, Oracle Virtual Directory enables you to use multiple data sources for Oracle Access Manager. See the *Oracle Access Manager Installation Guide* for details.

4

Integrating with Oracle Application Servers

This chapter describes integrating with OracleAS Single Sign-On for authentication and authorization purposes. When integrating Oracle Access Manager's authorization functionality, either Oracle Access Manager or OracleAS Single Sign-On can act as the authentication engine.

This integration enables you to provide identity management functionality across Web-based applications that run on Oracle Application Servers, for example, Oracle E-Business Suite, Oracle Forms, Portals, and other Access System-protected resources.

This chapter covers the following topics:

> **Note:** This chapter does not describe configuration of the Oracle Virtual Directory Server. See the *Oracle Access Manager Installation Guide* for details.

Integration Overview and Environment Preparation

This section discusses the following topics:

Supported Authentication Schemes for the Oracle Application Servers

Oracle Access Manager provides authentication and single sign-on for OracleAS 10g. This enables you to use a single user name and password (and optionally a realm ID), to log in to all features of the Oracle Application Servers and other Web applications. The integration uses the following authentication schemes:

- Form based
- Basic
- Custom
- Integrated Windows Authentication

OracleAS 10g Infrastructure

OracleAS 10g applications provide a similar infrastructure and a security framework for single sign-on for Oracle and other partner applications. The integration of Oracle Access Manager single sign-on with OracleAS 10g involves the following components.

OracleAS Single Sign-On Server: This enables Oracle applications to accept authentication from other applications. You can enable single sign-on between Access System-protected applications and applications protected within the OracleAS 10g single sign-on framework. You can use a single user name and password and optionally a realm ID to log in to all features of the Oracle Application server and other Web applications.

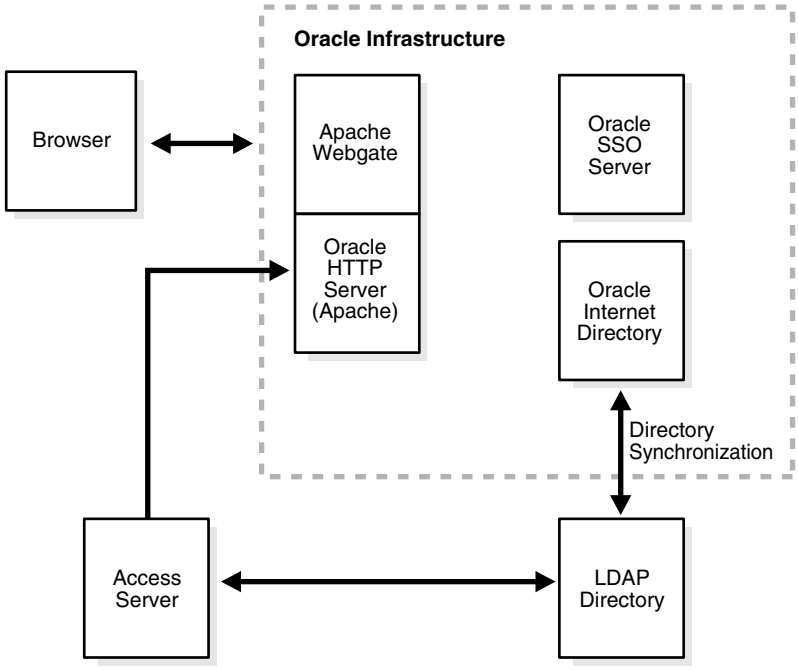
Oracle HTTP Server: This is the Web server interface for OracleAS 10g. Oracle HTTP Server is the integration point between Oracle Access Manager and OracleAS 10g. During the installation, a WebGate is installed as a module on Oracle HTTP Server. You must use the 10*g* (10.1.4.0.1) WebGate for Oracle HTTP Server.

Oracle Internet Directory (OID): The LDAP directory that serves as a user repository for OracleAS 10g applications. The OID can be synchronized with other connected directories.

Integration Architecture

Figure 4–1 illustrates the integration between Oracle Access Manager and Oracle Application Servers.

Figure 4–1 Oracle Access Manager and Oracle Application Server Integration Architecture



Process overview: Integration of Oracle Access Manager with Oracle Application Server

1. When a user attempts to access an Oracle Access Manager-protected application or Web resource, a WebGate intercepts the request.
2. WebGate requests the security policy from the Access Server to determine if the resource is protected.
3. When the resource is protected, WebGate prompts the user to authenticate.
4. The credentials entered by the user are validated against the directory for authentication.
5. When authentication is successful, an encrypted Oracle Access Manager single sign-on cookie is set on the user's browser.
6. After successful authentication, the Access System determines if the user is authorized by applying policies that have been configured for the resource.
7. Upon successful authorization, the Access System executes the actions that have been defined in the security policy and sets an HTTP header variable that maps to the OracleAS 10g user ID.
8. The OracleAS Single Sign-On Server recognizes the Oracle Access Manager HeaderVar, authenticates the user, and sets the Oracle single sign-on Cookie.

Note: The OID must be synchronized with the Oracle Access Manager directory to ensure that user data is up-to-date. OID performs the synchronization.

Supported Versions and Platforms

You can find support and certification information at the following URL:

http://www.oracle.com/technology/documentation/

You must register with OTN to view this information.

Also, you can see the supported versions and platforms for this integration on Metalink, as follows.

To view information on Metalink

1. In your browser, enter the following URL:

   https://metalink.oracle.com

2. Log in to MetaLink.
3. Click the **Certify** tab.
4. Click **View Certifications by Product**.
5. Select the **Application Server** option and click **Submit**.
6. Choose **Oracle Identity Management** and click **Submit**.
7. Click **Oracle Identity Management Certification Information 10g (10.1.4.0.1) (html)** to display the Oracle Identity Management page.
8. Click the link for **Section 6, Oracle Access Manager Certification** to display the certification matrix.

Preparing Your Environment

The following task overview lists the requirements for preparing for configuring single sign-on.

Task overview: Preparing your Environment

1. Install OracleAS 10g.
2. Install the Oracle Infrastructure.

   OracleAS Infrastructure 10g includes:

   - Oracle Application Server Metadata Repository
   - OracleAS Single Sign-On Server
   - Oracle Internet Directory (a lightweight directory access protocol (LDAP))

   Note: The servers where the Oracle infrastructure and Oracle Access Manager are installed must have fully qualified domain names, for example, *hostname.domain*.net.

3. Install and set up Oracle Access Manager components.

   See the *Oracle Access Manager Installation Guide* for details. Install the following:

   - Identity Server
   - WebPass
   - Access Server (includes Policy Manager)

4. On the Oracle HTTP Server, install a WebGate for use with OracleAS 10g.

   See the *Oracle Access Manager Installation Guide* for details.

   Update the Web server configuration file:

   - **Automatic Web Server Updates**: Click Yes to automatically update your Web server configuration file (Oracle HTTP Server httpd.conf) during WebGate installation, as described in the *Oracle Access Manager Installation Guide.*
   - **Manual Web Server Updates**: Use one of the following methods:

     Either: Locate the Oracle HTTP Server httpd.conf file after WebGate installation, add the WebGate entry at the end of the file, then run the following commands on an infrastructure terminal:

     ```
     Opmnctl restartproc process-type=HTTP_Server
     ```

     Or: Use the Oracle Enterprise Manager Console to:

     Launch the Oracle Enterprise Manager.

     Select the Oracle Application Server hosting the Oracle Infrastructure.

     Select the HTTP Server hosting the WebGate.

     Navigate to Advanced Server Properties.

     From the list of configured files, select httpd.conf for update.

     Include the WebGate entry at the end of the file.

5. Restart the Oracle HTTP Server after the Web Server configuration file update.
6. Configure OracleAS Single Sign-On for integration with third-party access management systems.

   See the related chapter in the *Oracle Application Server Single Sign-On Administrator's Guide* for details.
7. Configure the Web browser to allow cookies.
8. Proceed to "Single Sign-On with OracleAS 10g" on page 4-5.

Single Sign-On with OracleAS 10g

When integrating Oracle Access Manager with OracleAS 10g Application Server, each OracleAS application's configuration is provided separately. This integration requires configuring OracleAS 10g to integrate with third-party access management systems and configuring Oracle Access Manager logout.

You complete the following procedures to set up OracleAS 10g for the integration:

- Enabling Single-Sign On
- Integrating the Delegated Administration Service
- Integrating the Portal
- Enabling Single-Sign On for Forms
- Integrating Reports Services
- Synchronizing the Oracle Internet Directory and Oracle Access Manager LDAP Directory
- Implementing Global Logout from OracleAS Single Sign-On and Access Server

Task overview: Integrating Oracle Access Manager with OracleAS 10g

1. Set up your machines, as described in "Preparing Your Environment" on page 4-4.
2. Set up the OracleAS.
3. Set up Oracle Access Manager, as described in "Configuring Oracle Access Manager for Integration with OracleAS 10g" on page 4-10.
4. Test the integration, as described in "Testing the Integration with OracleAS" on page 4-17.

Enabling Single-Sign On

Enabling single-sign on for the integration between Oracle Access Manager and OracleAS 10g includes creating a java class and editing the policy.properties file, as discussed in the following paragraphs.

Creating the Java Class for Integration

The first step in enabling single sign-on for the integration involves coding a Java class, which will look for the Header variable from Oracle Access Manager.

Note: This example assumes you have installed and set up the Identity System and Access System, created a policy domain in the Access System, defined an authorization action that sets a Header Variable with the ID of the user, and configured global logout. See "Protecting the Single-Sign On Login URL" on page 4-13 and "Implementing Global Logout from OracleAS Single Sign-On and Access Server" on page 4-9 for details.

To code a JAVA class to look for a Oracle Access Manager HeaderVar

1. In the Access System, create rules to protect the following URIs:

   `/sso/auth/`

   `/pls/orasso/orasso.wwsso_app_admin.ls_login`

   See "Protecting the Single-Sign On Login URL" on page 4-13 for details.

2. Create a Java file for your package.

   For help, copy the source code from "SSOOblixAuth.java" on page 4-18, or the Sample Files section #SSOOblixAuth.java in the following location:

   ORCLE_HOME`/sso/lib`

   Save the file as SSOOblixAuth.java. Before it is compiled, this package directive must be added to it:

   ```
   package oblix.security.ssoplugin;
   ```

3. Compile the file, including *ORACLE_HOME*`/sso/lib/ipastoolkit.jar` in the class path. The sample file SSOOblixAuth.java is compiled this way:

   ```
   ORACLE_HOME/jdk/bin/javac -classpath
   ORACLE_HOME/sso/lib/ipastoolkit.jar:ORACLE_HOME/lib/servlet.j
   ar -d ORACLE_HOME/sso/plugin SSOOblixAuth.java
   ```

   Note that the colon separator (":") is appropriate for Linux. On Windows, use a semicolon (";") as the separator.

This command creates SSOOblixAuth.class and places it in the directory *ORACLE_HOME*/sso/plugin/oblix/security/ssoplugin.

4. Next you need to register the Java class for integration by editing the policy.properties file in the following location:

   OracleAS_install_dir/sso/conf

   Where *OracleAS_install_dir* is the directory where OracleAS Single Sign-On infrastructure is installed.

5. In the OracleAS Single Sign-On policy.properties file, replace the simple authentication plug-in with the plug-in that you created in the previous steps. In this class, navigate to the line MediumSecurity_AuthPlugin:

   ```
   MediumSecurity_AuthPlugin =
   oracle.security.sso.server.auth.SSOServerAuth
   ```

   Comment out the existing line and add a new line to register your Java class, as follows:

   ```
   MediumSecurity_AuthPlugin =
   oblix.security.ssoplugin.SSOOblixAuth
   ```

   When editing policy.properties, take care not to insert blank space at the end of a line.

6. Save the file.

7. Restart the single sign-on middle tier, and restart the OC4J instance OC4J_SECURITY to have your changes to take effect:

   ```
   ORACLE_HOME/opmn/bin/opmnctl restartproc
   process-type=HTTP_Server
   ```

   ```
   ORACLE_HOME/opmn/bin/opmnctl restartproc
   process-type=OC4J_SECURITY
   ```

8. Test the integrated system

Integrating the Delegated Administration Service

The Delegated Administration Service (DAS) is part of the Oracle Identity Management, an integrated infrastructure that includes the following components:

- Oracle Internet Directory—An LDAP V3-compliant directory service
- Delegated Administration Service (DAS)—The Oracle Internet Directory component that provides trusted proxy-based administration of directory information by users and application administrators.
- Oracle Directory Integration Service—A component of the Oracle Internet Directory that permits synchronization between the Oracle Internet Directory and other directories and user repositories.
- Provisioning Integration Service—The Oracle Internet Directory component that provides automatic provisioning of services, as described in Oracle documentation.

The DAS is installed by default when you install the OracleAS 10g Infrastructure, and should integrate automatically. No additional steps are needed for a user to access DAS when Oracle Access Manager is integrated with single sign-on.

The DAS link is:

`http://infra-machine-name:port/oiddas`

Note: If you experience errors using Create/Edit user and Create/Edit groups portlets, move the DAS to the middle tier from the Infrastructure. For details, see "Integrating the Portal" on page 4-8.

Integrating the Portal

The Oracle Application Server Portal enables you to build, deploy, and maintain self-service, integrated Enterprise Information Portals (EIPs). A customized portal page can present information from different providers and can include both enterprise search and directory lookup fields.

A portal page consists of multiple portlets. Each portlet is a region of the portal page that provides dynamic access to a Web-based resource.

When Oracle Access Manager single sign-on is integrated with OracleAS 10g, users should be able to access the portal as follows:

`http://midtier_home:port/pls/portal`

Note: The Create/Edit user and Create/Edit groups portlets call the DAS from the portal. If you experience errors using Create/Edit user and Create/Edit groups portlets, you need to move the DAS to the middle tier from the Infrastructure.

Enabling Single-Sign On for Forms

The Oracle Application Server Forms Services is a middle-tier application framework that you use to deploy complex transactional forms applications to the internet.

When you integrate Oracle Access Manager with OracleAS 10g, you need to enable single sign-on for forms. Once single sign-on is enabled for forms, Oracle Access Manager handles authentication and you should not be challenged to enter the schema user ID and password either by the single sign-on login page or by the forms.

To enable single sign-on for forms

1. Locate the forms90.conf file located in the following directory:

   `midtier_home/forms90/server`

2. At the end of the forms90.conf file add the following lines.

```
<IfModule mod_osso.c>
   <Location /forms90/f90servlet>
      require valid-user
      AuthType Basic
   </Location>
</IfModule>
```

3. Restart OC4J_BI_FORMS and the forms server to have you changes take affect.

   Next you create a Resource Access Descriptor (RAD) for the OID users. A RAD can be created at a global level so all users can use the same RAD to access the resource. Alternatively, the RAD can be created for each user.

4. Create a Resource Access Descriptor (RAD) for the OID users to map the LDAP user to the Database schema.

The next step can be done at the global level in the formsweb.cfg file (the default configuration), or at the application level to make individual applications single sign-on enabled.

5. Set the ssoMode to true to make the application single sign-on enabled using the Enterprise Manager to update the formsweb.cfg file.

   For example, to make an individual application single sign-on enabled:

   ```
   [myApp]
   form=myFmxs
      ssoMode=true
   ```

   For more information, see chapter 6 in the *Oracle Application Server Forms Services Deployment Guide 10g (9.0.4) for Windows and Unix*, Part No. B10470-02.

6. Test this implementation by navigating to the following URL:

   `http://midtier_home:port/forms90/f90servlet?config=default`

Integrating Reports Services

The Oracle Application Server Reports Services allow you to deploy reports to the OracleAS 10g, as described in your Oracle documentation.

Reports are single sign-on-enabled out of the box and should work without further steps when you integrate Oracle Access Manager with OracleAS 10g.

To access the protected reports page

1. Point your browser to the following URL:

   `http://machine:port/reports/rwservlet/showenv`

2. Log in when challenged by WebGate.

3. Confirm that once authenticated you can view the Environment settings for Oracle Reports (an single sign-on-protected page).

For more information, see chapter 10 of the *Oracle Application Server Reports Services Publishing Reports to the Web 10g (9.0.4),* Part No B13673-01.

Synchronizing the Oracle Internet Directory and Oracle Access Manager LDAP Directory

The next step in the configuration of OracleAS 10g for integration with Oracle Access Manager is to use the Oracle synchronization tool to synchronize user information between the Oracle Internet Directory and the LDAP directory server used by Oracle Access Manager.

For details about this synchronization tool and process, see the Oracle Internet Directory documentation.

Note: To test the integration without synchronizing the directories, you need to create an Oracle administrator (oracladmin) within Oracle Access Manager for login purposes.

Implementing Global Logout from OracleAS Single Sign-On and Access Server

By default, the WebGate logs a user out when it receives a URL containing "logout." See the section on logout from a single domain single sign-on session in the *Oracle*

Access Manager Access System Administration Guide for details. As a result, the default single sign-on logout page does not work with OracleAS Single Sign-On. The discussion "Logout.jsp" on page 4-19 provides a sample file you that need to configure logout.

To implement global logout from OracleAS Single Sign-On

1. Edit the following parameters in *ORACLE_HOME*/sso/conf/policy.properties. Substitute the paths to your logout page for the value shown in the following example:

   ```
   #Deployment login page link
   loginPageUrl = /sso/pages/login.jsp
   logoutPageUrl = /sso/pages/logout.jsp
   ```

2. Restart the single sign-on server:

   ```
   ORACLE_HOME/opmn/bin/opmnctl restartproc process-type=HTTP_Server
   ORACLE_HOME/opmn/bin/opmnctl restartproc process-type=OC4J_SECURITY
   ```

3. In the Access System, go to the page where you configure the single sign-on logout URL.

   From the Access System Console, click System Configuration, then click Server Settings, then click Configure SSO Logout URL.

4. On this page, configure the single sign-on logout URL to invoke the OracleAS Single Sign-On logout URL.

   Add a logout URL similar to the following:

   http://*host*:*port*/sso/logout

   Where *host* is the computer where the OracleAS Single Sign-On server is installed and *port* is the listen port for the server. When the user clicks the Logout link in Oracle Access Manager, the logout URL removes session cookies and redirects users to a logout page. See the appendix on configuring logout in the Oracle Access Manager Access System Administration Guide for details.

5. Go to the page where you configure the WebGate logout URL from the Access System Console by clicking Access System Configuration, then click AccessGate Configuration, then select a WebGate.

6. On the page that shows the WebGate details, click Modify, then provide a new logout URL similar to the following:

   /access/oblix/lang/en-us/style2/oblixlogo.gif

   The URL can be any gif file or Web page. This page is embedded in logout.jsp. See "Logout.jsp" on page 4-19 for details.

7. Repeat the previous two steps for every WebGate-protected cookie domain.

8. Add a page that you want to display after the user is logged out.

9. Confirm that you can perform a global logout both from Oracle AS Single Sign-On Server and from the Access Server.

Configuring Oracle Access Manager for Integration with OracleAS 10g

After installing Oracle Access Manager and installing a WebGate on the OracleAS HTTP Server, you need to create Oracle Access Manager access control policies to protect OracleAS resources.

Task overview: Setting up Oracle Access Manager for integration with OracleAS 10g includes

1. Install and set up the Identity System and Access System, as outlined in "Preparing Your Environment" on page 4-4.
2. Navigate to the Identity System Console and create an Oracle Administrator (orcladmin) user to match the orcladmin user who already exists in the Oracle OID, as described in the *Oracle Access Manager Identity and Common Administration Guide*.
3. Complete "Protecting the Single-Sign On Login URL" on page 4-13.

Configuring the Access System for OracleAS Single Sign-On 10.1.2.0.2

In addition to following the other information in this chapter, you must also complete the following procedure to integrate the Access System with OracleAS Single Sign-On 10.1.2.0.2.

To configure the integration with OracleAS Single Sign-On 10.1.2.0.2

1. Follow the steps in this chapter on configuring the integration.
2. In the Access System Console, click **System Configuration**, then click **Server Settings**, and configure the following logout URL:

   `http://[host.domain]:[port]/pls/orasso/ORASSO.wwsso_app_admin.ls_logout?p_done_url=http%3A%2F%2F[host.domain]%3A[port]`

   URL-encode the `p_done_url` value.

   See the *Oracle Application Server Single Sign-On Administrator's Guide* for release 10.1.2.0.2 for details on configuring the logout link for single sign-on. A sample JSP that can be used for this purpose is included at the end of this release note.
3. If you use the following sample JSP, go to the Access System Console, click **Access System Configuration**, then click **AccessGate Configuration**, and include the following in the **LogOutURLs** parameter for every WebGate in your environment:

   `/access/oblix/lang/en-us/style2/oblixlogo.gif`

The following is a sample `logout.jsp` file:

```
<!-- Copyright (c) 1999, 2003, Oracle. All rights reserved. -->
<%@page autoFlush="true" session="false"%>
<%
// Declare English Message Strings
String msg1 = "Single Sign-Off";
String msg2 = "Application Name";
String msg3 = "Logout Status";
String msg4 = "ERROR: The return URL value not found.";
String msg5 = "ERROR: Logout URL for partner applications not found.";
// Get the user language preference
String userLocaleParam = null;
java.util.Locale myLocale = null;
// Get the user locale preference sent by the SSO server
try
{
userLocaleParam = request.getParameterValues("locale")[0];
}
catch(Exception e)
{
userLocaleParam = null;
```

```
}
if( (userLocaleParam == null) || userLocaleParam.equals("") )
{
myLocale = request.getLocale();
}
else
{
if(userLocaleParam.indexOf("-") > 0 )
{
// SSO server sent the language and territory value (e.g. en-us)
myLocale = new java.util.Locale(userLocaleParam.substring(0, 2),
userLocaleParam.substring(3, 5));
}
else
{
// SSO server sent only the language value (e.g. en)
myLocale = new java.util.Locale(userLocaleParam, "");
}
}
// The following two lines will be used only for the Multilingual support
with
// proper resource bundle class supplied
// java.util.ResourceBundle myMsgBundle
// = java.util.ResourceBundle.getBundle("MyMsgBundleClassName", myLocale);
// Get the message string in the appropriate language using the message key.
// Use this string to display the message in this page.
// String mesg = myMsgBundle.getString("mesg_key");
%>
<html>
<body bgcolor="#FFFFFF">
<h1><%=msg1%></h1>
<%
String done_url = null;
int i = 0;
// Get the return URL value
try
{
done_url = request.getParameterValues("p_done_url")[0];
}
catch(Exception e)
{
done_url = "";
}
// Get the application name and logout URL for each partner application
try
{
%>
<b> <%=msg2%>   <%=msg3%> </b>
<br>
// Substitute an actual host, domain, and port for
myhost.us.mydomain.com:7777
// that points to the WebGate.
<img
src="http://myhost.us.mydomain.com:7777/access/oblix/lang/en-us/style2/oblixlo
go.gif">
<%
for(;;)
{
i++;
String app_name = request.getParameterValues("p_app_name"+i)[0];
```

```
String url_name = request.getParameterValues("p_app_logout_url"+i)[0];
%>
<%=app_name%>

<img src="<%=url_name%>">
<br>
<%
}
}
catch(Exception e)
{
if(done_url == null)
{
%>
<%=msg4%> <br>
<%
}
if(i>1)
{
%>
<br> <a href="<%=done_url%>">Return</a>
<%
}
else
{
%>
<%=msg5%><br>
<%
}
}
%>
</body>
</html>
```

Protecting the Single-Sign On Login URL

You need to protect the following single sign-on login URL so that the WebGate challenges the user whenever the OracleAS Single Sign-On 10g is accessed:

```
/sso/auth/
```

The following activities are required to protect the single sign-on login URLs, or any other resources, using the Access System.

Each step in the following task list is a full procedure. For complete details, see the related chapters in this guide.

Task overview: Protecting resources with Oracle Access Manager

1. Define an authentication scheme using the Access System Console.

   For example:

   ```
   Access System Console, Access System Configuration, Authentication Management,
   Add
   ```

2. Create a policy domain using the Policy Manager.

   For example:

   ```
   Policy Manager, Create Policy Domain
   ```

3. Add a resource to your policy domain using the Policy Manager.

   For example:

   ```
   Policy Manager, Create Policy Domain, Resources
   ```

4. Define rules for your policy domain using the Policy Manager.

   For example:

   ```
   Policy Manager, Create Policy Domain, Default Rules
   ```

5. Define an Authorization action that sets a Header Variable with the ID of the user.

   For example:

   ```
   Policy Manager, Create Policy Domain, Default Rules, Authorization Expressions,
   Actions
   ```

   Authorization Success

   Return

   Type: HeaderVar

   Name: *XXX*_REMOTE_USER

   Return Attribute: *loginAttribute*

   where *XXX* is any prefix (used because "REMOTE_USER" is often an internal header for HTTP servers) and where *loginAttribute* is the attribute configured as the Login semantic type in the Identity System. This name must map to the login name of the user stored in the OracleAS single sign-on repository. Some people have used the "EMPLID" attribute, which will pass the Employee ID of logged in user.

   Upon successful authorization, the value of *loginAttribute* is passed on to the OracleAS 10g server.

   > **Note:** To use a HeaderVar that is different from *XXX*_REMOTE_USER, you need to replace *XXX*_REMOTE_USER with the desired variable in two locations: Access System Console, Authorization Rule, Actions, and in the OracleAS Java class. See "Creating the Java Class for Integration" on page 4-6 for details.

6. In the Authorization rule, allow access to Anyone.

   For example:

   ```
   Policy Manager, Create Policy Domain, Authorization Rules, Name, Allow Access,
   Any one
   ```

7. Enable the Authorization rule.

   For example:

   ```
   Policy Manager, Create Policy Domain, Authorization Rules, Name,
   ```

8. Enable the Policy Domain.

   For example:

   ```
   Policy Manager, My Policy Domains, Name, Modify, Enabled
   ```

The single sign-on configuration is now complete.

9. Test your policy domain, as described in the section on using Access Tester in the *Oracle Access Manager Access System Administration Guide*.

Authorization Support for Applications Protected by OracleAS Single Sign-On

By default, the WebGate component of Oracle Access Manager intercepts all URLs, and the Access System authenticates the users who invoked the URLs. However, if you want to use OracleAS Single Sign-On to provide the authentication functionality for application login, you can configure the OHS Web server to pass authentication requests to mod_osso. This enables OracleAS Single Sign-On to continue to authenticate the user. Additionally, you can configure OracleAS Single Sign-On to pass the user's information to Oracle Access Manager for authorization.

This section describes how to implement Access System-based authorization for OracleAS Single Sign-On-protected HTTP resources.

The rest of this section discusses the following topics:

- About Authorization of OracleAS Single Sign-On-Protected Applications
- Configuring Authorization Support for OracleAS Single Sign-On-Protected Resources

About Authorization of OracleAS Single Sign-On-Protected Applications

In this type of integration, it is assumed that you have configured user authentication for various applications using OracleAS Single Sign-On. See the *Oracle Application Server Single Sign-On Administrator's Guide* for details.

After OracleAS Single Sign-On authenticates a user, Oracle Access Manager applies an external authentication scheme that looks for a REMOTE_USER header variable and maps it to an Oracle Access Manager user. If Oracle Access Manager can authenticate the user, the Access System performs user authorization. During authorization, the WebGate checks for the REMOTE_USER header variable. If it is set, the WebGate performs authorization according to polices that are defined in the Access System.

Configuring Authorization Support for OracleAS Single Sign-On-Protected Resources

This section assumes that you have installed OracleAS Single Sign-On, configured the middle tier applications to use OracleAS Single Sign-On authentication, and installed the WebGate on the middle-tier OHS. See the information on configuring the middle tier in the *Oracle Application Server Single Sign-On Administrator's Guide* and the section on "Preparing Your Environment" on page 4-4 in this chapter for details.

The following procedure describes configuring OracleAS Single Sign-On authentication with Oracle Access Manager authorization.

To configure authentication using OracleAS Single Sign-On and authorization using Oracle Access Manager

1. On the computer that hosts the OHS Web server, comment following lines in the WebGate section in the file *ORACLE_HOME*/Apache/Apache/conf/httpd.conf:

```
<LocationMatch "/*">
AuthType Oblix
require valid-user
</LocationMatch>
```

2. On Linux, locate the WebGate-specific section in the httpd.conf file.

   This section is enclosed by the following lines:

   ```
   #*** BEGIN WebGate Specific ****
   #*** END WebGate Specific ****
   ```

   Move this section before the line that contains the include statement for mod_osso.conf.

3. Restart the Web server for this WebGate.

4. Protect your resources on the middle-tier OHS with OracleAS Single Sign-On using static pattern rules.

   See the *Oracle Identity Management Application Developer's Guide* for details. This is required to use OracleAS Single Sign-On authentication features, for example, Windows Native Authentication.

To define an external authentication scheme in Oracle Access Manager

1. From the Oracle Access Manager landing page, click Access System Console, click Authentication Management, and click Add.

2. Define an authentication scheme similar to the following on the General tab for the authentication scheme:

   Name: External auth scheme

   Challenge Method: Ext

   Challenge Parameter: creds:REMOTE_USER

3. On the Plug-ins tab for the authentication scheme, add a credential mapping plug-in that uses the REMOTE_USER header variable, for example:

   ```
   obMappingBase="dc=us,dc=mycompany,dc=com",obMappingFilter="(&(&
   (objectclass=inetorgperson)(uid=%REMOTE_USER%))(|(!(obuseraccountcontrol=*))
   (obuseraccountcontrol=ACTIVATED)))"
   ```

   When implementing this plug-in, substitute values for obMappingBase and the person object class that are appropriate for your environment.

To define the policies to protect the middle-tier application URLs

1. From the landing page for Oracle Access Manager, click Policy Manager.

2. Click Create Policy Domain.

3. Define policies to protect any middle-tier application URL.

   Configure the polices using the external authentication scheme that you configured in the previous procedure. See the *Oracle Access Manager Access System Administration Guide* for details.

4. If a WebPass and Policy Manager are installed on the same Web server as the WebGate, configure OracleAS Single Sign-On to authenticate users who try to access the Identity and Access Systems.

   Add two static URL patterns to the OracleAS Single Sign-On http.conf file:

   ```
   <LocationMatch "/identity/oblix">
     AuthType Basic
     require valid-user
   </LocationMatch>
   ```

```
<LocationMatch "/access/oblix">
  AuthType Basic
  require valid-user
</LocationMatch>
```

These rules enable OracleAS Single Sign-On to perform authentication for the Identity System and Policy Manager.

5. Also, if a WebPass and Policy Manager are installed on the same Web server as the WebGate, ensure that the external authentication scheme that you configured in the previous procedure is protecting the Identity and Access domains.

   See the *Oracle Access Manager Access System Administration Guide* for details.

To configure logout for the integration

1. See"Implementing Global Logout from OracleAS Single Sign-On and Access Server" on page 4-9 for details.

Testing the Integration with OracleAS

After you set up OracleAS and Oracle Access Manager for integration, test to ensure that the integration is successful.

To test Oracle Access Manager single sign-on for OracleAS

1. For OracleAS 10.1.4.0.1 and higher, enter the following URL in the browser:

   `http://machinename:port/sso/`

   Where *machinename* is the machine where the OracleAS Server is installed and *port* is the port number of the machine.

   For OracleAS 10.1.2, enter the following URL in the browser:

   `http://machinename:port/pls/orasso/`

   You should be presented with a login page. After you have successfully authenticated, the OracleAS Web resource page appears.

2. You can try to access various applications as the same user.

   If Oracle Access Manager single sign-on is successful, you will be allowed access to the page without being challenged for authentication.

3. You an also try to test different authorization rules in the Access System.

   For example, if there are time conditions set for login, you may try logging in at different times.

4. When you are ready to log out, click the Logout link.

   If Oracle Access Manager single sign-on is successful, you will be logged out of all Oracle Access Manager-protected resources.

OracleAS 10g Files

The following two sample files can be customized to meet your requirements:

- SSOOblixAuth.java
- Logout.jsp

SSOOblixAuth.java

```
package oblix.security.ssoplugin;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import oracle.security.sso.ias904.toolkit.IPASAuthInterface;
import oracle.security.sso.ias904.toolkit.IPASAuthException;
import oracle.security.sso.ias904.toolkit.IPASUserInfo;
import oracle.security.sso.ias904.toolkit.IPASInsufficientCredException;
import java.net.URL;
import java.util.*;

public class SSOOblixAuth implements IPASAuthInterface
{
    private static String OBLIX_USER_HEADER = "XXX_REMOTE_USER";
    private static String CLASS_NAME = "SSOOblixAuth";

    public SSOOblixAuth()
    {
      System.out.println("Inside SSOOblixAuth constructor.....");
    }
    public IPASUserInfo authenticate(HttpServletRequest request)
      throws IPASAuthException, IPASInsufficientCredException {


      String OblixUserName = null;

      try
       {
         System.out.println(".............Getting Header Variable.............");
         OblixUserName = request.getHeader(OBLIX_USER_HEADER);

         System.out.println("The Header name............."+OblixUserName);
       }
       catch (Exception e)
       {
         throw new IPASInsufficientCredException("No Oblix Header");
       }

       if (OblixUserName == null)
           throw new IPASInsufficientCredException("No Oblix Header");

       IPASUserInfo authUser = new IPASUserInfo(OblixUserName);
       System.out.println("The IPASUserInfo Class............."+authUser);
       return authUser;

    }


    public URL getUserCredentialPage(HttpServletRequest request,String msg) {

System.out.println("Inside Get User Credietial Page ..........Should not come
here>......");

        URL errorURL=null;
        try
        {
           errorURL=new URL(new String(request.getRequestURL()));
        }
        catch(Exception ee){};
```

```
        return  errorURL;
    }


}
```

Logout.jsp

You can use the following sample file as discussed in "Implementing Global Logout from OracleAS Single Sign-On and Access Server" on page 4-9.

```
<!-- Copyright (c) 1999, 2003, Oracle. All rights reserved. -->
<%@page autoFlush="true" session="false"%>
<%
// Declare English Message Strings
String msg1 = "Single Sign-Off";
String msg2 = "Application Name";
String msg3 = "Logout Status";
String msg4 = "ERROR: The return URL value not found.";
String msg5 = "ERROR: Logout URL for partner applications not found.";
// Get the user language preference
String userLocaleParam = null;
java.util.Locale myLocale = null;
// Get the user locale preference sent by the SSO server
try
{
userLocaleParam = request.getParameterValues("locale")[0];
}
catch(Exception e)
{
userLocaleParam = null;
}
if( (userLocaleParam == null) || userLocaleParam.equals("") )
{
myLocale = request.getLocale();
}
else
{
if(userLocaleParam.indexOf("-") > 0 )
{
// SSO server sent the language and territory value (e.g. en-us)
myLocale = new java.util.Locale(userLocaleParam.substring(0, 2),
userLocaleParam.substring(3, 5));
}
else
{
// SSO server sent only the language value (e.g. en)
myLocale = new java.util.Locale(userLocaleParam, "");
}
}
// The following two lines will be used only for the Multilingual support with
// proper resource bundle class supplied
// java.util.ResourceBundle myMsgBundle
// = java.util.ResourceBundle.getBundle("MyMsgBundleClassName", myLocale);
// Get the message string in the appropriate language using the message key.
// Use this string to display the message in this page.
// String mesg = myMsgBundle.getString("mesg_key");
%>
<html>
<body bgcolor="#FFFFFF">
```

```
<h1><%=msg1%></h1>
<%
String done_url = null;
int i = 0;
// Get the return URL value
try
{
done_url = request.getParameterValues("p_done_url")[0];
}
catch(Exception e)
{
done_url = "";
}
// Get the application name and logout URL for each partner application
try
{
%>
<b> <%=msg2%>   <%=msg3%> </b>
<br>
// Substitute an actual host, domain, and port for myhost.us.mydomain.com:7777
// that points to the WebGate.
<img
src="http://myhost.us.mydomain.com:7777/access/oblix/lang/en-us/style2/oblixlogo.g
if">
<%
for(;;)
{
i++;
String app_name = request.getParameterValues("p_app_name"+i)[0];
String url_name = request.getParameterValues("p_app_logout_url"+i)[0];
%>
<%=app_name%>
 
<img src="<%=url_name%>">
<br>
<%
}
}
catch(Exception e)
{
if(done_url == null)
{
%>
<%=msg4%> <br>
<%
}
if(i>1)
{
%>
<br> <a href="<%=done_url%>">Return</a>
<%
}
else
{
%>
<%=msg5%><br>
<%
}
}
%>
```

```
</body>
</html>
```

Troubleshooting the OracleAS 10g Integration

The following are troubleshooting tips for the Oracle 10g integration.

Problem: With a form-based authentication scheme, while accessing OIDDAS/Form application/ externally deployed J2EE applications, the OracleAS single sign-on login page is displayed after the Oracle Access Manager Form login page.

Solution: This happens if mod_osso uses a POST based redirection method instead of GET to call the single sign-on server. The redirection method used is based on value of OssoRedirectByForm directive. To use GET method, this directive needs to be set to false. In Oracle 10g Application Server, this value is set to false by default.

To verify that this directive is set to false

1. Verify the value of OssoRedirectByForm directive.
2. Launch the Oracle Enterprise Manager.
3. Select the Oracle Application Server instance where the Oracle Infrastructure is installed.
4. Select the HTTP Server where WebGate is installed and navigate to Advanced Server Properties.
5. From the list of configured files, select the mod_osso.conf file.
6. Check if OssoRedirectByForm is set to true.

   By default the values is false.
7. If the default directive value is not used, set it to false as shown in the following example:

   ```
   <IfModule mod_osso.c>
   OssoIpCheck off
   OssoIdleTimeout off
   OssoConfigFile
   /private1/iasinst/install_set1/904infra/Apache/Apache/conf/osso/osso.conf
   OssoRedirectByForm off
   </IfModule>
   ```
8. Click Apply.
9. Restart the OracleAS HTTP Server.

Problem: How do I find ORASSO and Portal schema passwords?

Solution: Complete the following procedure.

To find these database schema passwords

1. Login to Oracle Directory Manager as the super user orcladmin.
2. Expand the tree on the left hand side, as follows:

   Cn= OracleContext

   Cn=Products

   Cn=IAS

   Cn=IAS Infrastructure Databases

OrclReferenceName=<global database name>

OrclResourceName=ORASSO

3. Click the ORASSO entry and look for the value for attribute orclpasswordattribute (the Password for ORASSO schema).

> **Note:** Similarly you can click the OrclResourceName=PORTAL for the portal schema password.

Problem: How do I check the single sign-on logs?

Solution: You can view the single sign-on logs from Enterprise Manager (EM).

1. Log in to EM.
2. Click the Logs link at the bottom of the page.

   A search screen appears.
3. From the Available Components list select Single Sign-on:orasso and move it to the Selected Components.
4. Perform the search to view the single sign-on logs.

Problem: How do I create a default RAD?

Solution: Complete the following steps to create a default RAD:

To create a default RAD

1. Access OIDDAS Console, Configuration, Preference, as usual.
2. Scroll to the bottom of the page to display Resource Access Information.
3. Click Create to create a new resource file.
4. Enter a Resource Name:

   For example, for a default configuration you can use:

   ```
   default
   ```

> **Note:** Resource name created over here should be the same as the configuration present in formsweb.cfg file.

5. Click Next, fill in the user ID and password and the connect string for the database, and click Submit.

   The user ID is a valid DB user. Database refers to the DB used. For example, if a schema named "Scott" is used and a Database "asdb"', the test entries are:

   Username: *scott*

   Password: *tiger*

   Database: *asdb*

Problem: How do I create a user-specific RAD?

Solution: Complete the following steps to create a user-specific RAD:

To create a user-specific RAD

1. Access the OIDDAS console, as usual.
2. Select the Directory tab found at the top right hand corner of the page.
3. Click Create to create a new user.
4. Select a user name, for example, ssotest with a password of ssotest1.

   You can choose to add all other details.
5. Scroll to the bottom of the page to Resource Access Information.
6. Click Create to create a new resource file.
7. Enter a Resource Name, for example, ssotest_db.
8. Click Next, fill in the user ID, password, and connect string for the database, then click Submit.

The user ID here is a valid DB user. For testing purposes, the default Scott schema can be used. Database is the DB used, with a default value of asdb. For example, the test entries could be:

Username: scott

Password: tiger

Database: asdb

5

Federated Single Sign-On Using Oracle Identity Federation

Users need to access content on different corporate Web sites. Corporate Web sites need to authenticate and authorize users who are entering from different domains that use different security products. The Oracle Identity Federation product addresses these problems.

When a user tries to access a protected resource on a remote Web site, the Oracle Identity Federation product at the user's site transfers information about the user to the remote site for use in authorizing the user's request. For example:

- Users from an airline can access technical documentation in an airplane vendor's documentation database.
- Customers of a wireless company can access a bill-paying application that is outsourced from the vendor to a third-party supplier.
- Employees of an organization can access a 401(k) application through an internal HR portal that connects to the benefits provider.

Users might click a link on their own company's Web site to request access to content on a partner (Service Provider) Web site. The first time users request access, they are authenticated on their home (or Identity Provider) domain using information stored in their home site's user data repository. The user's home domain forwards the access request to the Service Provider site along with the credentials that the Service Provider site needs to authorize the request.

Oracle Access Manager can serve as the authentication engine for federated single sign-on using Oracle Identity Federation.

In addition to federated single sign-on, you can extract user attributes from the Oracle Access Manager user directory for federated authorization. This is useful when a Service Provider needs additional information to authorize users to access resources. For example, if you are a Service Provider and your authorization policies require specific user attributes that are not available on a user's request, you may need to send a request to the identity Provider for additional user attributes that are needed for authorization. In Oracle Access Manager, you configure authentication and authorization plug-ins to query the user's home site (the Identity Provider) for these attributes.

This chapter discusses the following topics:

- Configuring the Authentication Scheme for Attribute Sharing
- Configuring the Authorization Schemes and Policies for Attribute Sharing

About Federated Single Sign-On

Information about single sign-on between Oracle Access Manager and Oracle Identity Federation is provided in the *Oracle Identity Federation Administrator's Guide*. This guide is available from the Oracle Documentation page on the Oracle Technology Network. The URL is as follows:

`http://www.oracle.com/technology/documentation`

About Federated Authorization

When the Access System at a Service Provider site receives a request from a user in a federated environment, it may need to get additional information about the user from the user's Identity Provider. For example, the user may authenticate to the Service Provider using SSL with an x.509 certificate, then attempt to access a resource that is protected by the Access System using an x.509-based authentication scheme. If the user is not defined in the Service Provider's user repository, the Access System may need additional user attributes from the Identity Provider to be able to make an authorization decision.

If you have Oracle Identity Federation installed at your site, and you receive requests from other sites that have installed Oracle Identity Federation (or another product that implements the SAML Attribute Sharing Profile for x.509 Authentication-Based Systems), you can configure the Access System to perform federated authorization.

This section describes the components involved in authorization decisions based on federated requests for additional user attributes.

The following process overview describes a scenario where federated attribute sharing is used for authorization.

Process overview: Attribute sharing used for federated authorization

1. A user authenticates to your Service Provider domain using x.509 client certificate authentication.

   Client certificate authentication is required for this process.

2. The user requests a resource in your domain.

   Your domain is the Service Provider. A WebGate protects the requested resource.

3. The WebGate intercepts the request and directs it to the Access Server.

4. An attribute-sharing authentication plug-in determines that the user is not defined in the local Oracle Access Manager System.

5. An authentication step sets up the conditions for using an attribute-sharing authorization plug-in.

6. The attribute sharing authorization plug-in extracts user identity from the user's x.509 certificate and forwards it to the local instance of Oracle Identity Federation.

7. The local Oracle Identity Federation engine constructs a SAML attribute request and forwards it to the SAML service (for example, **Oracle Identity Federation**) at the user's home domain (the Identity Provider).

8. The SAML service at the Identity Provider extracts the user attributes from the request, verifies the user identity, and returns a response to the Service Provider instance of Oracle Identity Federation.
9. The response is forwarded to the Access Server to complete the authorization process.

Components of attribute sharing-based authorization are as follows:

- **A config.xml file**—A configuration file is read during Access Server start-up.

  This configuration file specifies user DN patterns and associated URLs that point to instances of Oracle Identity Federation at the Service Provider domain. The authentication and authorization plug-ins for attribute sharing use the DN patterns to determine if a user is local or remote. For remote users, the associated URLs indicate the instance of Oracle Identity Federation that should receive requests for additional information about the user.

  This file also provides authentication information that Access System sends requests to identify itself to the Oracle Identity Federation instance.

  You also configure logging and system-related settings in this file.

- **An authentication plug-in for attribute sharing**—This plug-in determines if a user is local, that is, if the user has an identity in the local Oracle Access Manager System.

  If the plug-in determines that the user is local, the authentication scheme associated with the plug-in creates a local Oracle Access Manager session for the user. If the plug-in determines that the user is remote, the scheme creates a placeholder session for the OblixAnonymous user.

- **An authorization plug-in for attribute sharing**—The attribute sharing authorization plug-in constructs a request for attributes that are required for authorizing the user. The plug-in sends the request to an instance of Oracle Identity Federation at the local domain. The configuration file that the Access Server reads at start-up time determines the URL where the request is sent.

  The Oracle Identity Federation instance at the local domain forwards a SAML request to the SAML service at the Identity Provider.

  The SAML service at the Identity Provider responds to the request, and the response is sent along the same route back to the Access System. If the Access System obtains the requested attributes, it provides access to resources and other authorization actions based on the rules configured in the authorization scheme.

Note: This chapter only describes configuration of attribute sharing authorization in the Access System. However, reciprocal configuration must be performed on the local Oracle Identity Federation server. For more information on Oracle Identity Federation, see the *Oracle Identity Federation Administrator's Guide*, available from the Oracle Documentation page on the Oracle Technology Network. The URL is as follows:

http://www.oracle.com/technology/documentation

Setting up the Federated Attribute Sharing Environment

Setting up the environment includes configuring basic parameters to identify the attributes to be used in authorization, URLs for the Oracle Identity Federation, setting up secure communication channels, configuring logging, and other tasks.

The following sections provide information on these topics:

- Setting Parameters in the config.xml File
- Configuring Basic Authentication
- Configuring SSL and Client Certificate Authentication
- Configure the Session Token Cache for Federated Attribute Sharing

Setting Parameters in the config.xml File

The config.xml file contains settings for associating local Oracle Identity Federation instances with users's Identity Providers based on information in the user's DN entry. It also contains settings that control what information related to attribute sharing is written to a log, and other settings to control interaction with the Access System.

The config.xml file is located in the following directory:

Access_Server_install_dir/access/oblix/config/attributePlugin/config.xml

The following is a sample config.xml file:

```
<Config LogLevel="audit" WaitTime="30" SizeLimit="0" MaxConnections="5"
     InitialConnections="2" Authn="cert" Username="MyAccessServer"
     Password="MyPassword" KeyPassword="MyKeyPassword" CacheTimeout="30"
     MaxCachedUsers="1000" HeaderKeyLength="128" HeaderKeyRegen="86400"
     RequestFormat="values">
   <Mapping Local="true">
       <DN>OU=Engg,O=MyCompany,L=MyCity,ST=California,C=US</DN>
   </Mapping>
   <Mapping URL="https://host1.us.company.com:8888/osfs/ar/soap">
       <DN>OU=Test,O=MyCompany,L=MyCity,ST=California,C=US</DN>
   </Mapping>
   <Mapping URL="https://host2.us.company.com:8777/osfs/ar/soap">
       <DN>C=US</DN>
       <DN>C=IN</DN>
   </Mapping>
   <Mapping URL=http://SP-HOST:SP-PORT/fed/ar/soap">
        <DN>O=Partner1,C=US</DN>
   </Mapping>
   <Mapping URL="http://SP-HOST:SP-PORT/fed/ar/soap" RequestFormat="all">
        <DN>O=Partner2,C=US</DN>
    </Mapping>
</Config>
```

This file contains the following attributes:

Table 5–1 Attributes in the config.xml File

Attribute	Description
LogLevel	Controls the amount of information that is logged to *AS_install_dir*/oblix/logs/authz_attribute_plugin_log.txt. The following are possible values for this parameter: ■ **none**—Nothing is logged except errors (the default). ■ **audit**—One line is logged for each authorization request, showing the access decision, the user's certificate subject DN or local directory DN, and the HTTP operation and the local part of the requested URL. ■ **debug**—Extensive information used in debugging problems.
HTTP connection parameters	The authorization plug-in uses these parameters. These parameters contain information that is sent from the Access System to the Oracle Identity Federation instance. You can configure the following HTTP connection parameters: ■ **WaitTime**—Time in seconds to wait for a response. Default: 30 seconds. ■ **SizeLimit**—Maximum size in bytes of HTTP messages sent and received. 0 means unlimited (the default). ■ **MaxConnections**—The maximum number of concurrent HTTP connections. Default: 5. ■ **InitialConnections**—Number of concurrent HTTP connections opened initially. Default: 2.
Authentication parameters	Authenticates the authorization plug-in to the Oracle Identity Federation instance. You can configure the following authentication parameters: ■ **Authn**—The authentication method. The following are possible values for this parameter: **none**—No authentication is used. **basic**—Use HTTP basic authentication with user name and password (the default). **cert**—Use SSL client certificate authentication using key.pem, cert.pem, and KeyPassword. ■ **Username**—The username for basic authentication. ■ **Password**—The password for basic authentication. ■ **KeyPassword**—The password for key.pem for SSL client certificate authentication.
Attribute value cache parameters	This cache is located in the authorization plug-in memory in the Access Server. You can set values for caching parameters: ■ **CacheTimeout**—The time in seconds that cached attributes values are held before requiring updated values. A value of 0 disables caching. Default: 3600 seconds (1 hour). ■ **MaxCachedUsers**—The maximum number of users whose attribute values can be cached. If the cache is full, entries with the oldest unexpired entries are reclaimed. Default: 1,000.

Table 5–1 (Cont.) Attributes in the config.xml File

Attribute	Description
Encryption parameters	**HeaderKeyLength**—The length (in bits) of a key that is generated for AES encryption of the SubjectDN header. This header is passed from the authentication plug-in to the authorization plug-in. Possible values for this parameter: 128, 192, or 256. Higher values provide stronger encryption but slower performance. A value of 0 disables encryption of the header. Oracle does not recommend setting a value of 0 due to potential impersonation attacks. **HeaderKeyRegen**—The interval (in seconds) for regenerating the key that encrypts the SubjectDN header. The default is 86400 (one day).
Attribute query properties	The RequestFormat parameter determines what attributes and values are returned on an attribute response. This parameter overrides authorization rules. For example, if an authorization rule specifies attributes and values, the RequestFormat parameter can omit the values from a request. You can specify the RequestFormat parameter in the global CONFIG element or in a local MAPPING element in the config.xml file. For example, in the sample config.xml file shown above this table, Partner2 uses a local RequestFormat setting. The RequestFormat parameter can be configured as follows: ■ **RequestFormat="*values*"**—This setting enables a query for information about a user to contain attribute names and values. The names and values are taken from the authorization rule expression that you configured for federated attribute sharing. With this setting, the response from the Identity Provider only returns user attributes and values that match those in the query. This is the default setting. This setting minimizes the memory used for cached attribute values because the request contains only the values neededed for authorization. This setting results in more frequent attribute requests. ■ **RequestFormat="*names*"**—This setting permits a query to contain attribute names, but not the values that you configured in the authorization rule expression that you configured for federated attribute sharing. The response from the Identity Provider returns the user's values for the named attributes, as long as the Identity Provider's Responder policies permit access to the values. This setting uses more cache memory than the "values" setting, but less than the "all" setting. This setting does not disclose to the Identity Provider what attribute values are required for authorization. For security reasons, this setting may be preferable over the "values" setting. ■ **RequestFormat="all"**—The setting omits attribute names and values from a query. The Identity Provider returns all user attributes and values, as long as the Identity Provider's Responder policies permit access to the attributes and values. This setting minimizes the number of attribute requests to one per user, but it uses the most cache memory. Use this setting if the Attribute Responder policies are configured to only return attributes that the Service Provider may want. This setting does not disclose to the Identity Provider what attributes are required for authoirzation. For security reasons, this may be preferable to the "values" and "names" settings.

Table 5–1 (Cont.) Attributes in the config.xml File

Attribute	Description
Mappings of Subject DNs to Attribute Requester Service URLs.	The following parameters enable the Access System to determine if users are local—that is, if they have an identity in Oracle Access Manager—or remote and must be identified by an Attribute Requester Service. For remote users, these parameters map the subject DNs to a URL for your domain's (the Service Provider) instance of Oracle Identity Federation. For local users, authorization can be determined by the local Oracle Access Manager. You can configure following parameters: ■ **DN**—One or more elements of a DN pattern to match against the user Subject DN in the request that the Access System receives. The pattern consists of the rightmost components of the DN, for example: `O=MyCompany,L=MyCity,ST=California,C=US` ■ **Local**—If this parameter is true, the matching users are assumed to be local and the URL parameters are ignored. ■ **URL**—The URL for the Oracle Identity Federation instance. The form is HTTP:// or HTTPS://*OIF_host*:*OIF_port*/fed/ar/soap Where *OIF_host* is the host name and *OIF_port* is the port of a local Oracle Identity Federation server. This is the server that receives requests from the Access System for verification of a user's attributes and forwards a SAML-formatted request to an Identity Provider's SAML services.

The following table provides examples of how subject DNs are evaluated according to the config.xml file. When Oracle Access Manager evaluates the Subject DN of a user, users who are deemed to be local are given local Oracle Access Manager credentials. All other users are referred to the attribute sharing authorization plug-in. Depending on the DNs of the remote users, a corresponding URL is used for sending a query to a local Oracle Identity Federation instance:

Table 5–2 Mappings of Subject DNs

User Subject DN	Mapping
E=user1k1@company.com,CN=John Smith,OU=Engg,O=MyCompany, L=MyCity,ST=California,*C=US*	local
E=user1k2@company.com, CN=Margaret Abel, OU=Test,O=MyCompany, L=MyCity, ST=California,*C=US*	https://host1.us.company.com:8888/fed/ar/soap
E=user1k3@othercompany.com, CN=Fred Jones, OU=Sales,O=OtherCompany, L=OtherCity,ST=Iowa,*C=US*	https://host2.us.company.com:8777/fed/ar/soap
E=user1k4@othercompany.com, CN=Mahitha Chandra,OU=Sales,O=OtherCompany, L=OtherCity,ST=TamilNadu,*C=IN*	https://host2.us.company.com:8777/fed/ar/soap

To configure the config.xml file:

1. Log in to the Access Server host as the user who installed the Access Server.

2. Create a file named config.xml in this directory.
3. Edit the attributes and elements of the config.xml file in the following directory:

   AS_install_dir/access/oblix/config/attributePlugin
4. Restart the Access Server.

Configuring Basic Authentication

If you are using basic authentication between the plug-in and Oracle Identity Federation, you need to add the following to the httpd.conf file of the Oracle HTTP Server for your Oracle Identity Federation instance. The following shows Basic authentication for a user named "Alice":

```
<LocationMath "/fed/ar/soap">
     Allow Override None
     AuthType Basic
     AuthName "Restricted Files"
     AuthUserFile /private/oifpassword
     Require user alice
     Order allow,deny
     Allow from all
</LocationMatch>
```

A user passwords file must also be created using the htpasswd utility. In the previous example, the AuthUserFile containing the users and their passwords points to the/private/oifpassword file, in which the user alice is defined. This example creates such a file by adding the user alice:

```
Apache/Apache/bin/htpasswd -c /private/oifpassword "alice"
```

Set the authn parameter to basic in "Setting Parameters in the config.xml File" on page 5-4, and set the other required authentication parameters.

Finally, restart the Access Server.

Configuring SSL and Client Certificate Authentication

To send and receive information over a secure channel, you can configure SSL and client certificate authentication. You would select the type of secure channel as follows:

- **HTTPS**—Configure HTTPS if you specified HTTPs in the config.xml file and have configured HTTPS between the authorization plug-in and t at least one instance of Oracle Identity Federation. Configuring HTTPS involves pasting the CA certificate for each trusted Oracle Identity Federation instance into the cacerts.pem file for the plug-in.
- **Client certificate authentication**—Configure client certificate authentication if you configured SSL client certificate authentication between the authorization plug-in and at least one instance of Oracle Identity Federation. Configuring client certificate authentication involves generating a private key, requesting a new certificate, and pasting the certificate into cert.pem.

To set up HTTPS

1. Go to the wallet of the Oracle Identity Federation instance and export the CA certificate.
2. Open the exported file in a text editor and copy the contents, including the `----BEGIN CERTIFICATE-----` and `-----END CERTIFICATE-----` statements.

3. In another text editor, open the following file:

   AS_install_dir/oblix/config/attributePlugin/cacerts.pem

4. Paste the information that you copied to the end of the cacerts.pem file.
5. Restart the Access Server.

To set up client certificate authentication

1. Generate the private key and certificate request using the openssl command provided with Oracle Access Manager:

   cd *AS_install_dir*/oblix/tools/openssl

   ```
   openssl req -config openssl.cnf -newkey rsa:1024 -keyout
   ../../config/attributePlugin/key.pem - out
   ../../config/attributePlugin/req.pem
   ```

2. Send the req.pem file produced in the previous step to your CA to get a certificate.
3. Copy the generated certificate to the following location:

   AS_install_dir/oblix/config/attributePlugin/cert.pem

4. Change the mode to "cert" in the following file:

   AS_install_dir/oblix/config/attributePlugin/config.xml

5. Set the authn parameter to cert in "Setting Parameters in the config.xml File" on page 5-4, and set the other required authentication parameters.
6. Restart the Access Server.

Configure the Session Token Cache for Federated Attribute Sharing

The federated attribute sharing feature requires that an authentication be performed on each request. The authentication creates a new session token that is not actually used by the attribute sharing feature. However, the session token can be cached in the Access Server.

If all of your users authenticate using the attribute sharing authentication scheme, you can disable the session token cache to minimize the amount of memory used by the Access Server. If some users authenticate using other authentication schemes, you should enable the session token cache. See the section on Access Server configuration parameters in the *Oracle Access Manager Access System Administration Guide* for details.

Set the maximum number of elements in the session token cache to balance Access Server memory usage and performance requirements. A large value for this cache increases memory usage. A small value for the cache reduces memory usage but can cause the cache to fill up quickly with unused tokens from the attribute sharing scheme, resulting in performance degradation because the cache is always full.

Configuring the Authentication Scheme for Attribute Sharing

The attribute sharing authentication scheme consists of the following:

- Basic characteristics, including a name and the X509Cert authentication method.
- An authentication step to handle a local user.

  If the user is local, Oracle Access Manager can provide the attributes required for authorization.

- An authentication step to handle a remote user.

This step sets an OblixAnonymous login for the user.

This section discusses the following topics:

- Configuring the Basic Components of the Authentication Scheme
- Configuring Plug-ins and Steps for the Authentication Scheme

Configuring the Basic Components of the Authentication Scheme

The basic components of the authentication scheme include defining the name of the scheme, its authentication level, and challenge method.

To configure the basic characteristics of the attribute sharing authentication scheme

1. Log in to the Access System Console as a Master Access Administrator.
2. Select the Access System Configuration tab.
3. Click Authentication Management in the left navigation pane.

   The Authentication Management: List All Authentication Schemes page appears.
4. Click Add.
5. Define a new authentication scheme with the following parameters:

   Name—OIF Attribute Sharing

   Description—Performs an SSL client certificate authentication for OIF Attribute Sharing authorization.

   Level—Set this value based on the requirements of the protected resources. The level should be higher than any password schemes.

   Challenge Method—Select the X509Cert option.

   Challenge Parameter—Enter the following in this field:

   ```
   ssoCookie:Expires=Tue, 1 Nov 2005 00:00:00 GMT
   ```

   This ensures that the authentication scheme is run on every access to protected resources. The challenge parameter forces the browser to discard the obSSOCookie, which forces a re-authentication.

   SSL Required—Select Yes.

   Enabled—Select No (set this to Yes after the plug-ins are configured).
6. Click Save.

Configuring Plug-ins and Steps for the Authentication Scheme

After defining the authentication scheme, you need to add plug-ins to the scheme. The plug-ins evaluate user credentials and create an appropriate Oracle Access Manager session for the user. You also need to configure the flow of plug-in evaluation.

To configure the plug-ins for the attribute sharing authentication scheme:

1. After completing the steps in "To configure the basic characteristics of the attribute sharing authentication scheme" on page 5-10, select the Plugins tab.
2. Click Modify.
3. Add the following plug-ins and parameters.

To enter built-in plug-ins, select the plug-in name from the list. To enter custom plug-ins, select Custom Plugin from the Plugin Name list and enter the plug-in name in the text box.

Table 5–3 Plug-in Descriptions

Plug-in Name	Type	Parameters
authz_attribute	custom	Leave this field blank. This plug-in extracts the Subject DN field from the certificate and determines if the user is local or remote. Although this is a custom plug-in, the plug-in code is provided by Oracle. When an authentication scheme includes this plug-in in a step, the step calls a function in the plug-in library *AS_install_dir*/access/oblix/lib/authz_attribute.so or .dll. Note that this is the same library used by the authorization plug-in described later. Built-in plug-ins such as credential_mapping and decode_cert consist of code within the Access Server, and have no externally visible libraries.
cert_decode	built-in	Leave this field blank. This plug-in addresses client certificate (cert) authentication by validating the certificate. It does not use a data source.

credential_mapping	built-in	Add plug-in parameters using the following syntax. This plug-in assigns an OblixAnonymous login to a remote user. The *user_ID* attribute is the login attribute defined for Oracle Access Manager. You can use the credential_mapping parameters from the Anonymous authentication scheme if it is available.
		
		`obMappingBase="mapping_base", obMappingFilter="(user_id=OblixAnonymous)"`
		
		Example:
		
		`obMappingBase="o=Company,c=US", obMappingFilter="(uid=OblixAnonymous)"`

credential_mapping	built-in	Add plug-in parameters using the following syntax. This plug-in determines if the user is active, and if so, creates a session for the user.
		
		`obMappingBase="mapping_base", obMappingFilter="(&(& (objectclass=person_object_class) (user_attribute=%certSubject.field%)) (|(!(obuseraccountcontrol=*)) (obuseraccountcontrol=ACTIVATED)))"`
		
		Example:
		
		`obMappingBase="o=Company,c=US", obMappingFilter=" (&(&(objectclass=inetorgperson) (mail=%certSubject.E%)) (|(!(obuseraccountcontrol=*)) (obuseraccountcontrol=ACTIVATED)))"`

4. When all the plug-ins have been added, click Save.

To configure the steps for the attribute sharing authentication scheme

1. After completing the tasks in "To configure the plug-ins for the attribute sharing authentication scheme:" on page 5-10, select the steps panel, click Add, and add the following steps:

Table 5–4 Step Configuration

Step Name	Add Plug-In or Plug-Ins	Purpose of this Step
SubjectDN	authz_attribute	Extract the SubjectDN from the certificate and determine if the user is remote or local.
RemoteUser	first credential_mapping plug-in	Create an anonymous session for a remote user.
LocalUser	cert_decode plug-in second credential_mapping plug-in	Create a real session for a local user.

2. Select the Authentication Flow tab, click Modify, and set the following flow.

   The authz_attribute plug-in returns Success if the user is remote and Failure if the user is local. Remote users are assigned the OblixAnonymous login and the attribute sharing authorization scheme is invoked. For local users, a session that applies to that user is assigned.

Table 5–5 Authentication Flow Configuration

Step Name	Initiating Step	On Success, the Next Step Is. . .	On Failure, the Next Step Is. . .
Default Step	No	Stop	Stop
SubjectDN	Yes	RemoteUser	LocalUser
RemoteUser	No	Stop	Stop
LocalUser	No	Stop	Stop

3. Return to the Steps tab and remove the Default step.
4. Return to the General tab and enable the authentication scheme.

Configuring the Authorization Schemes and Policies for Attribute Sharing

The attribute sharing authentication scheme determines whether a user is local or remote. If the user is local, the user is logged in to Oracle Access Manager, and an authorization scheme determines the user's access permissions. If the user is found to be remote and more information is needed, an authorization plug-in sends a request to one or more local instances of Oracle Identity Federation.

The local instances of Oracle Identity Federation request verification of the remote user's identity from the instance of Oracle Identity Federation at the Identity Provider site for the user. The local instances of Oracle Identity Federation also return the values to Oracle Access Manager and an access decision is made based on the user attributes that are returned. The returned attributes are specified in the ruleExpression in the authorization rule.

The authorization plug-in maintains a cache of previously retrieved user attributes. If the user attributes that are required for authorization are already stored in the cache, these are used instead of sending a request for the values to the Oracle Identity Federation instance. This reduces the number of attribute requests for subsequent authorizations that require the same attributes.

An attribute sharing policy determines what resources are to be protected, the order of authorization rule evaluation, and the authentication and authorization schemes that apply to the protected resources.

This section discusses the following topics:

- Configuring Basic Characteristics of the Authorization Scheme
- Configuring Rules and Policies for the Attribute Sharing Authorization Scheme

> **Note:** In the following sections, the authorization rules evaluate individuals. This has been done to simplify the examples. It is likely that in practice these rules would select a set of users, for example jobTitle = "Manager" & clearance = "Secret", or OU = "Server Development Team".

Configuring Basic Characteristics of the Authorization Scheme

Basic characteristics of the authorization scheme include a name and a required parameter named ruleExpression, which is defined in "Configuring Rules and Policies for the Attribute Sharing Authorization Scheme" on page 5-14. In addition to creating a basic definition for the authorization rule, you need to indicate what resources this rule protects.

To configure basic characteristics of the attribute sharing authorization scheme:

1. Log in to the Access System Console as a Master Access Administrator.
2. Click the Access System Configuration tab.
3. Click Authorization Management in the left navigation pane.
4. Click Add.

   The Define a New Authorization Scheme page appears.
5. Enter the following:

   Name—OIF Attribute Sharing (or any other appropriate name).

   Description—Uses OIF to obtain attributes for remote users to evaluate the rule expression.

   Shared Library—Enter the following: `oblix/lib/authz_attribute`

   Plugin is Managed Code—Select No.

   Managed Code Name Space—Leave this field blank.

   User Parameter—Enter the following: `RA_SubjectDN`

   This value invokes an external query function to obtain the SubjectDN header that is set by the authz_attribute authentication plug-in. See the section on retrieving external data for an authorization request in the *Oracle Access Manager Access System Administration Guide* for details.

   Required Parameter—Enter `ruleExpression` in the Name field. Leave the Value field blank. Each access policy authorization rule will supply the rule expression.
6. Click Save.

To configure the protected resources in an attribute sharing access policy

1. Log in to the Policy Manager as a Master or Delegated Access Administrator.
2. Select Create Policy Domain.
3. Complete the General page, as follows:

   Name—Provide a unique name, for example, OSFS Attribute Sharing Test.

   Description—Provide an appropriate description.
4. Click Save.
5. Click the Resource tab and add one or more URL prefixes to protect.

   Example: `/attribute-test`

Configuring Rules and Policies for the Attribute Sharing Authorization Scheme

After defining the authentication scheme and protected resources for the scheme, you need to do the following:

- Define rules that determine who is authorized to access the protected resources.

  These rules contain expressions that parse user attributes and sets of attributes. The rules provide yes-or-no decisions based on the presence or absence of these attributes or sets of attributes in the user's credentials.

  You can create different rules for local and remote users.
- Define the evaluation order of these rules in the authorization policy, and create a default rule to apply if none of the other rules can be used when performing an authorization.
- Associate the policy with the attribute sharing authentication scheme as well as the attribute sharing authorization scheme.

To configure an authorization rule for remote users in an attribute sharing access policy

1. After completing the steps in "To configure an authorization rule for remote users in an attribute sharing access policy" on page 5-14, click the Authorization Rules tab, then click Add.

   Each rule is represented as an expression.
2. Select Custom Authorization Scheme, click Add, and fill out the form as follows:

   Name—Provide an appropriate name, for example, Remote User 1.

   Description—Provide a description that will help others understand this rule.

   Authorization Scheme—OIF Attribute Sharing.
3. Click the Plugin Parameters tab, click Modify, and define the rule expression.

   In the Name field, enter the following name: `ruleExpression`.

   In the Value field, provide a value that the Identity Provider is to search for. The value can be an expression. Use the minus ("-") and plus ("+") buttons to add expressions.

   White space is allowed around the equals sign ("="), bang equals ("!="), ampersand ("&"), and or operator ("|"). Table 5–6 lists the available parameters.

   Examples of rule expressions:

firstname="John" & lastname="Smith"

firstname="Conserving" & lastname="Levitin" | firstname="Mahitha" & lastname = "Valiveti"

firstname = "Charles" & (lastname = "Smith" | lastname="Chandra")

firstname=any & lastname=any.

Table 5–6 Expression Parameters for a Rule

Element	Syntax	Description
name	alphanumeric string including dash ("-"), underscore ("_"), and period ("..").	Name of an attribute to request from the user's Identity Provider.
value	one of "string", any, or null	Required attribute value. The string is restricted to Latin-1 characters. The any value retrieves and matches all values for the attribute. The null value matches a SAML <Attribute> with the xsi:nil="true" attribute.
comparison	name = value, name! = value, or (expression).	True if the user has or does not have the attribute value.
and-clause	comparison & comparison	True if both comparisons are true.
or-clause	comparison | comparison	True if either comparison is true. the ampersand ("&") has a higher precedence than the vertical bar ("|").

4. Save the authorization rule form.
5. Set any timing conditions or actions for the authorization rule.
6. Click the General tab and click Modify.
7. Select Yes in the Enabled list to enable the rule.

To configure an authorization rule for local users in an attribute sharing access policy

1. After completing the steps in "To configure an authorization rule for local users in an attribute sharing access policy" on page 5-15, click the Authorization Rules tab.
2. To add an authorization rule for each set of local user attributes, click Add.
3. In the Authorization Scheme list, select Oracle Authorization Scheme, then click Add.
4. Complete the authorization rule form as follows:

   Name—Provide an appropriate name, for example, User 1 Local.

   Description—Provide a description that will help others understand the rule.

   Enabled—Select Yes.

   Allow Takes Precedence—Select No.
5. Save the form.
6. Select the Allow Access tab and click Add.

   Add an LDAP filter for the local attributes.

You can use the Query Builder in the Identity System's User Manager application to build the filter. (From the Identity System Console, click Configuration, then click Delegate Administration, then click Build Filter.) The following is an example of a filter:

```
ldap:///o=Company,c=US??sub?(&(givenname=Rohit)(sn=Valiveti))
```

7. Set any timing conditions or actions as required for the authorization rule.
8. Click the General tab and click Modify.
9. Select Yes in the Enabled list to enable the rule.

To configure a rule expression in an attribute sharing access policy

1. After completing the steps in "To configure an authorization rule for local users in an attribute sharing access policy" on page 5-15, select the Default Rules tab.
2. Add the default authentication rule as follows:

   Name—An appropriate name, for instance, Default Rule

   Description—A meaningful description.

   Authentication Scheme—Select the OSFS Attribute Sharing authentication scheme.
3. Save the rule.
4. Select the Authorization Expression tab.
5. Select the remote authorization rule and click Add.
6. Select the local authorization rule, select OR, and add it. For example:

   ```
   Remote User 1 | Local User 2
   ```
7. Save the authorization rule.

6

Integrating Oracle Identity Management

Oracle Identity Management is a secure enterprise provisioning system that streamlines the creation and management of user accounts and revocation of user access rights and privileges. Oracle Identity Management automates access rights management, security, and provisioning of IT resources, and connects users to the resources they need to be productive.

This chapter describes using Oracle Access Manager to manage user authentication and authorization when a user logs in to Oracle Identity Management 9.0.1.1.

This chapter covers the following topics:

> **Note:** While this chapter focuses on using JBoss as the application server in the integration, the same configuration steps apply to instances where Oracle Identity Management is deployed on WebSphere, WebLogic or any other J2EE application server that is supported by Oracle Identity Management.

About the Integration with Oracle Identity Management

The integration of Oracle Access Manager with Oracle Identity Management provides a secure Web-based infrastructure for identity management for all customer applications and processes. Oracle Access Manager integrates identity and access management across Oracle Identity Management, enterprise resources, and other domains deployed on eBusiness networks. Oracle Access Manager provides the foundation for managing the identities of customers, partners, and employees across Internet applications. These user identities are combined with security policies for protected Web interaction.

This integration adds the following features to Oracle Identity Management implementations:

- **Oracle Access Manager authentication, authorization, and auditing** services for Oracle Identity Management.
- **Oracle Access Manager single sign-on** for Oracle Identity Management and other Oracle Access Manager-protected resources within a single domain or across multiple domains.
- **Oracle Access Manager authentication schemes,** the following schemes provide single sign-on for Oracle Identity Management:
  - **Basic**: Users must enter a user name and password in a window supplied by the Web server.

    This method can be redirected to SSL.'
  - **Form:** This method is similar to the basic challenge method, but users enter information in the custom HTML form.

    You can choose the information users must provide in the form that you create.
  - **X509 Certificates**: X.509 digital certificates over SSL.

    A user's browser must supply a certificate.
  - **Integrated Windows Authentication (IWA)**: Users will not notice a difference between an Oracle Access Manager authentication and IWA when they log on to the desktop, open an Internet Explorer (IE) browser, request a Oracle Access Manager-protected Web resource, and complete single sign-on.
  - **Custom**: Additional forms of authentication can be incorporated through use of the Oracle Access Manager Authentication Plug-in API.
- **Session timeout**: Oracle Access Manager enables you to set the length of time that a user session is valid.
- **Ability to use the Oracle Access Manager Identity System**: This system provides identity management features such as user self-service for registration and updating user profiles, portal inserts, delegated administration, and workflows. You can send Identity System data to back-end applications using a custom data template and a workflow.

Oracle Identity Management Components

The integration with Oracle Access Manager single sign-on involves the following Identity Manager components:

Identity Manager Server: This is a J2EE server application that implements business logic in Java Data Objects. The Java Data Objects are managed by a supported J2EE application server, including JBoss application server, BEA WebLogic, and IBM WebSphere.

Identity Manager Database Server: The database server manages the storage of data in Oracle Identity Management, including information about users, resources, business rules, provisioning processes and auditing and attestation in the database.

Identity Manager Web Client: The Oracle Identity Management user interface resides in this tier. Users log in using the Oracle Identity Management client. The Oracle Identity Management client provides the user's login credentials to the Oracle Identity Management server. The Oracle Identity Management server validates the credentials. Through the Oracle Identity Management client, you can submit requests to search for information in the database and save, edit, or delete that information.

Supported Version and Platforms

Any references to specific versions and platforms in this chapter are made for demonstration purposes.

You can find support and certification information at the following URL:

http://www.oracle.com/technology/documentation/

You must register with OTN to view this information.

Also, you can see the supported versions and platforms for this integration on Metalink, as follows.

To view information on Metalink

1. In your browser, enter the following URL:

   https://metalink.oracle.com

2. Log in to MetaLink.
3. Click the **Certify** tab.
4. Click **View Certifications by Product**.
5. Select the **Application Server** option and click **Submit**.
6. Choose **Oracle Identity Management** and click **Submit**.
7. Click **Oracle Identity Management Certification Information 10g (10.1.4.0.1) (html)** to display the Oracle Identity Management page.
8. Click the link for **Section 6, Oracle Access Manager Certification** to display the certification matrix.

Integration Architecture

Oracle Identity Management has two authentication mechanisms:

- Default mode, where Oracle Identity Management manages the credential validation and session maintenance.
- Single sign-on mode, where Oracle Identity Management looks for an HTTP header variable that is passed to it.

  The header variable should contain the user ID of the Oracle Identity Management user.

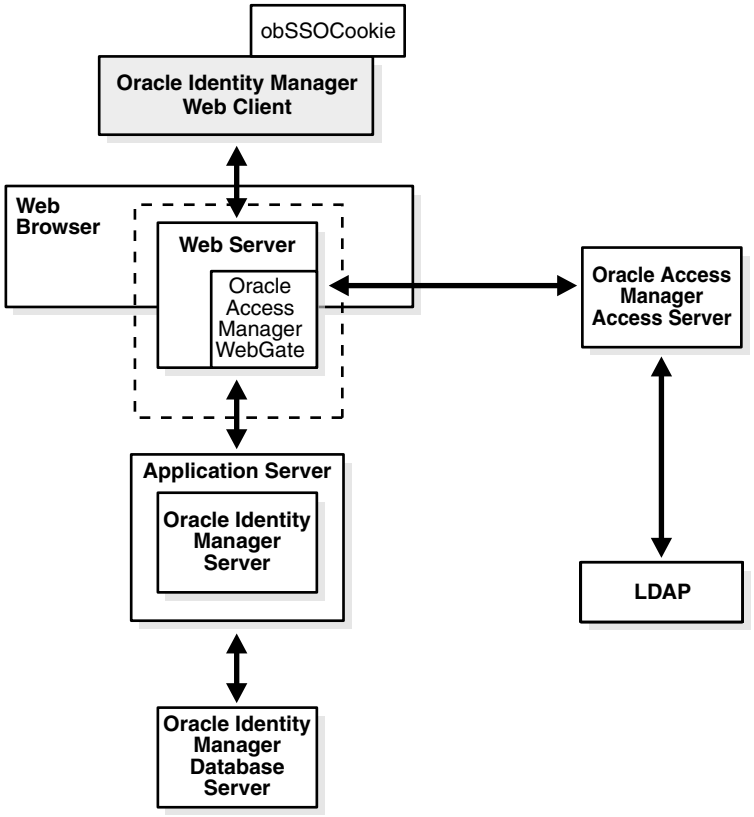
Oracle Access Manager single sign-on with Oracle Identity Management is achieved as follows:

- Deploy an HTTP Server in front of the J2EE Application server.
- Deploy the HTTP Server as a reverse proxy.
- Deploy a WebGate on the HTTP Server.
- Populate a header variable with an attribute value that is stored in the LDAP directory used by Oracle Access Manager.
- Configure IOracle Identity Management to use the single sign-on mode of authentication.

Figure 6–1 shows the architecture for single sign-on between Oracle Identity Management and Oracle Access Manager.

The user accesses the Oracle Identity Management Web client via a Web browser. The WebGate intercepts the user's HTTP request and checks for the presence of an obSSOCookie. If the cookie does not exist or it has expired, the user is challenged for credentials. Oracle Access Manager verfies the credentials, and if the user is authenticated, the WebGate redirects the user to the requested resource and passes the required header variable to Oracle Identity Management. Oracle Identity Management, which has been configured to read a HTTP Header variable instead of its authentication, reads the HTTP Header and uses the value stored in the variable as the logged in user.

Figure 6–1 Integration with Oracle Identity Management



Process overview: Single sign-on with Oracle Identity Management

1. A user attemps to access the Oracle Identity Management Web client.
2. A WebGate that is deployed on the HTTP server intercepts the request.
3. The WebGate checks the Access Server to determine if the resource (the Oracle Identity Management URL) is protected.

   The security policy in the Access System contains an authentication scheme, authorization rules, and allowed operations based on authentication and authorization success or failure.
4. If a valid session does not exist, and the resource is protected, WebGate prompts the user for credentials.
5. If the credentials are validated, Oracle Access Manager performs the actions that are defined in the security policy for the resource and sets an HTTP header variable that maps to the Oracle Identity Management user ID.

6. If a vali session cookie exists, and if the user is authorized to access the resource, WebGate redirects the user to the requested Oracle Identity Management resource.
7. The Oracle Identity Management Web client reads the HTTP header variable and sets the value as the logged-in user.
8. The Web client generates the applications pages, pending any further authorization checks performed in Oracle Identity Management.

Preparing Your Environment

Complete the following to prepare your environment for the integration.

Task overview: Preparing your environment for the integration

1. Install a supported directory server according to vendor instructions.
2. Install and configure Oracle Access Manager using the directory server as the LDAP repository.
3. Ensure that the Oracle Identity Management J2EE application server is proxied by an HTTP server.
4. Configure the Web browser to allow cookies, according to vendor instructions.
5. Set up Oracle Access Manager for Oracle Identity Management.

   See "Setting Up Oracle Access Manager Single Sign-On for Oracle Identity Management" on page 6-5 for details.

Setting Up Oracle Access Manager Single Sign-On for Oracle Identity Management

The following procedures describes setting up WebGate on an HTTP server and configuring Oracle Access Manager for single sign-on with Oracle Identity Management.

Note that you can configure form-based authentication for logins that use either ASCII or non-ASCII characters. Due to browser limitations, Basic authentication schemes only accept ASCII login credentials.

> **See also:** For more information about configuring authentication and authorization in Oracle Access Manager, see the *Oracle Access Manager Access System Administration Guide*.

To set up a WebGate on an HTTP server

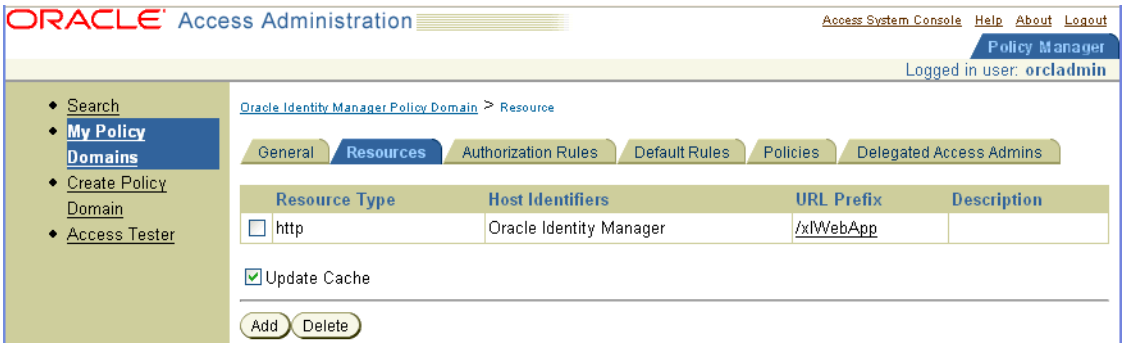
1. Install and configure Oracle Access Manager on a supported platform, using a supported LDAP server.

   See the *Oracle Access Manager Installation Guide* for details.
2. Install a WebGate on the Oracle Identity Management HTTP server.

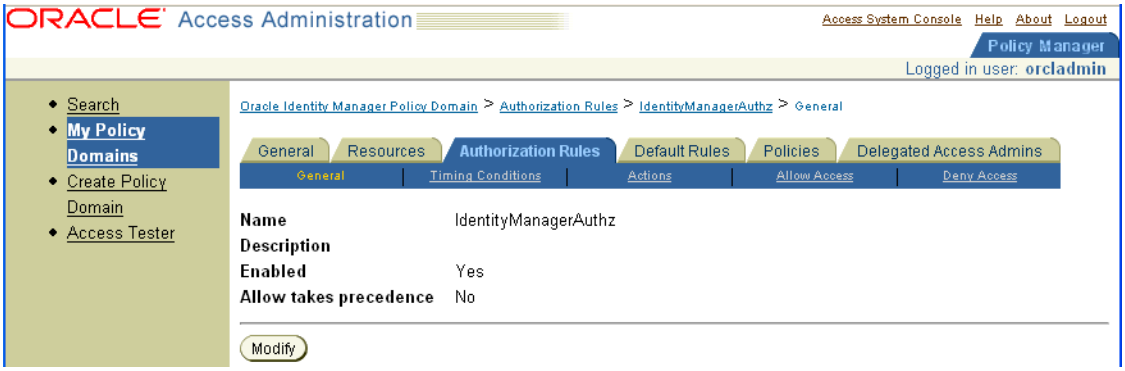
   Do not install the WebGate against an application server that supports HTTP services, for example, BEA Weblogic. If your application server is JBoss, IBM WebSphere, or BEA Weblogic, install an HTTP server such as Apache, iPlanet, or Oracle HTTP Server.
3. Configure the HTTP server to forward user requests to the J2EE application server and forward responses from the Oracle Identity Management back to the user.

To configure single sign-on in Oracle Access Manager

1. In the landing page for the Access System, click the link for the Policy Manager, and click Create Policy Domain.
2. Create a policy domain and policies to restrict access to the Oracle Identity Management URLs.
3. In the Access System Console, define host identifiers for Oracle Identity Management.
4. Click the link for the Policy Manager, click the link for the Oracle Identity Management policy domain, click the Resources tab, and define resources for Oracle Access Manager to protect.



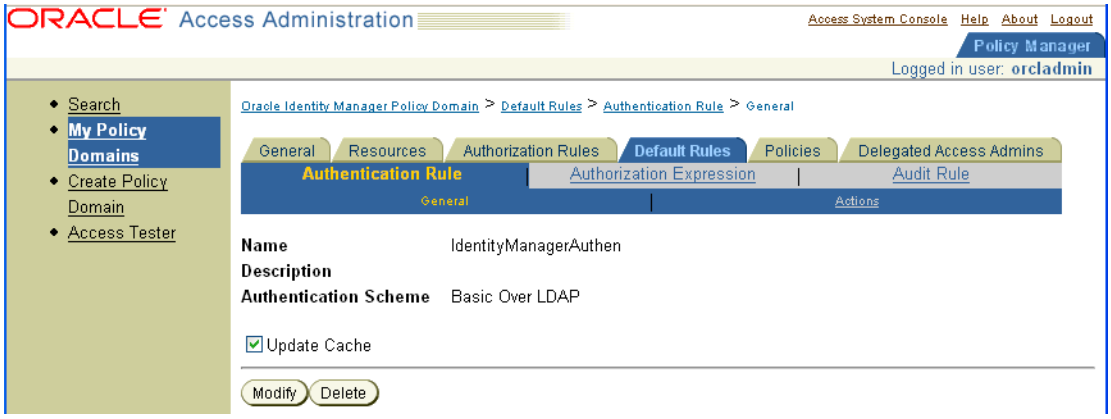
5. Click the Authorization Rules tab and define an authorization rule to determine which authenticated users can access the Oracle Identity Management URLs.



6. Click the Default Rules tab.

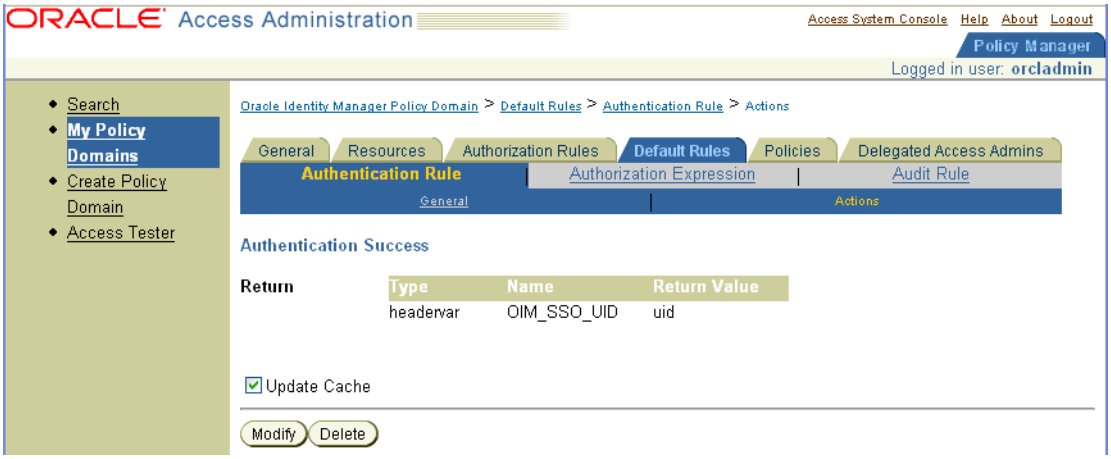
   The Authentication Rule sub-tab is selected.

7. Define an authentication rule, for example, Basic Over LDAP.

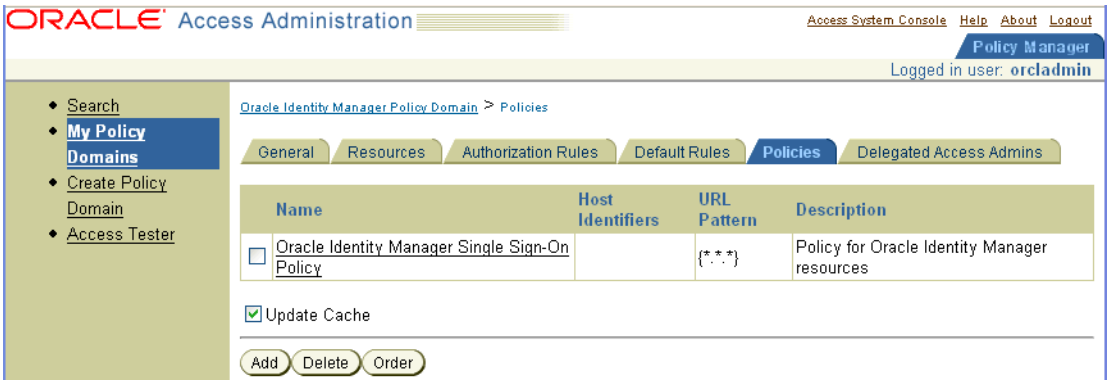


8. Click the Actions sub-tab and define an authorization action that sets a custom HTTP header variable upon successful authorization.

   The header variable should contain a value that maps to the Oracle Identity Management user ID.



9. Click the Policies tab, click Add, and define an access policy in the Oracle Identity Management policy domain and add the Oracle Identity Management URL resources to this policy.



Setting Up Oracle Identity Management for Single Sign-On with Oracle Access Manager

The following procedure describes how to set up Oracle Identity Management for integration with Oracle Access Manager.

To configure single sign-on for Oracle Identity Management

1. Stop the application server gracefully.
2. Launch a plain-text editor and open the following file:

   <*XL_HOME*>\xellerate\config\xlconfig.xml
3. Locate the following Single Sign-On configuration (the following are the default settings without Single Sign-On):

   ```
   <web-client>
   <Authentication>Default</Authentication>
   <AuthHeader>REMOTE_USER</AuthHeader>
   </web-client>
   ```

4. Edit the single sign-on configuration as follows.

   Replace <*SSO_HEADER_NAME*> with the appropriate header configured in your single sign-on system:

   ```
   <web-client>
   <Authentication>SSO</Authentication>
   <AuthHeader><SSO_HEADER_NAME></AuthHeader>
   </web-client>
   ```

   To enable single sign-on with non-ASCII character logins you must include a decoding class name to decode the non-ASCII header value. Add the decoding class name and edit the single sign-on configuration as follows:

   ```
   <web-client>
   <Authentication>SSO</Authentication>
   <AuthHeader><SSO_HEADER_NAME></AuthHeader>
   <AuthHeaderDecoder>com.thortech.xl.security.auth.CoreIDSSOAuthHeaderDecoder</Au
   thHeaderDecoder>
   </web-client>
   ```

   Replace <*SSO_HEADER_NAME*> with the appropriate header configured in your single sign-on system
5. Change your application server and web server configuration to enable single sign-on.

   Refer to your application and web server vendor documentation for details.
6. Restart the application server.

Configuring Apache as a Proxy for JBoss

The Oracle Identity Management Web client runs in a J2EE application server, for example, JBoss, BEA Weblogic, and IBM WebSphere. You cannot install an AccessGate directly against these application servers. You can deploy a Web servre, for example, Apache, Oracle HTTP Server, and iPlanet in front of these application servers. You can deploy the AccessGate on the Web server, and configure the Web server to route requests to the Oracle Identity Management application and forward responsees back to the user.

For application servers such as JBoss, you must deploy an additional plug-in, referred to as the mod_jk plug-in or the JBoss plug-in, on the Web server. You can obtain the mod_jk plug-in from the Apache Tomcat Web site, under the Tomcat connectors section. As of the time of publication, the URL as follows:

http://tomcat.apache.org/download-connectors.cgi

Notes: As mentioned in the section on "Supported Version and Platforms" on page 6-3, version numbers sited in this document are for illustration only. Refer to the URL provided in that section for current supported platforms.

The following procedure is based on JBoss 4.0.2, Apache 2.0 for Windows, and mod_jk 1.2.15.

To configure the Apache HTTP server as a proxy for JBoss

1. Download and install a version of the Apache HTTP Server that is supported by Oracle Access Manager.
2. Download the latest stable version of the Jakarta (also known as Tomcat) mod_jk plug-in from the following URL:

   http://tomcat.apache.org/download-connectors.cgi
3. Extract the file and rename it to mod_jk.so.
4. Copy this file to the following directory:

   Apache_install_dir\modules
5. Create the following text files in the directory *Apache_install_dir*\conf:
   - mod-jk.conf
   - workers.properties
   - uriworkermap.properties

   Oracle recommends that you do not rename uriworkermap.properties and workers.properties. If you do, your configuration may stop working. The locations of these files are defined under two registry keys: worker_file and worker_mount_file. These files are in HKEY_LOCAL_MACHINE\SOFTWARE\Apache Software Foundation\Jakarta Isapi Redirector*version_number*.
6. Copy the following configuration into the mod-jk.conf file:

```
# Load mod_jk module
# Specify the file name of the mod_jk lib
LoadModule jk_module modules/mod_jk.so

# Where to find workers.properties
JkWorkersFile conf/workers.properties

# Where to put jk logs
JkLogFile logs/mod_jk.log

# Set the jk log level [debug/error/info]
JkLogLevel info

# Select the log format
JkLogStampFormat "[%a %b %d %H:%M:%S %Y]"
```

```
# JkOptions indicates to send SSK KEY SIZE
JkOptions +ForwardKeySize +ForwardURICompat -ForwardDirectories

# JkRequestLogFormat
JkRequestLogFormat "%w %V %T"

# Mount  your  applications
JkMount /application/* loadbalancer

# You can use external file for mount points.
# It will be checkded for updates each 60 seconds.
# The format of the file is: /url=worker
# /examples/*=loadbalancer
JkMountFile conf/uriworkermap.properties

# Add shared memory.
# This directive is present with 1.2.10 and
# later versions of mod_jk, and is needed for
# for load balancing to work properly
JkShmFile logs/jk.shm

# Add jkstatus for managing runtime data
<Location /jkstatus/>
JkMount status
Order deny,allow
Deny from all
Allow from 127.0.0.1
</Location>
```

7. Copy the following into the workers.properties file:

```
# Define the list of workers that will be used
# for mapping requests
worker.list=loadbalancer

# Define node1
worker.node1.port=8009
worker.node1.host=<Put your Identity Manager App Server FQDN name here>
worker.node1.type=ajp13
worker.node1.lbfactor=1
worker.node1.local_worker=1 (1)
worker.node1.cachesize=10
#Load-balancing behaviour
worker.loadbalancer.type=lb
worker.loadbalancer.balance_workers=node1
worker.loadbalancer.sticky_session=1
worker.loadbalancer.local_worker_only=1
```

8. Copy the following into the uriworkermap.properties file.

   Configure the mapping according to the `worker.list` entry defined in the workers.properties file. This is not always `loadbalancer`, although this is shown in the following example:

```
# Simple worker configuration file
# Mount the servlet context to the ajp13 worker
/jmx-console=loadbalancer
/jmx-console/*=loadbalancer
/web-console=loadbalancer
/web-console/*=loadbalancer
/xlWebApp=loadbalancer
```

```
/xlWebApp/*=loadbalancer
/Nexaweb=loadbalancer
/Nexaweb/*=loadbalancer
```

7

Integrating Siebel 7

This chapter describes the integration of Oracle Access Manager 10*g* (10.1.4.0.1) with the Siebel 7 e-business platform. Siebel 7 is a Web-based suite that combines customer relationship management, partner relationship management, and employee relationship management applications.

This chapter covers the following topics:

About the Integration with Siebel 7

The integration of Oracle Access Manager with Siebel 7 provides a secure Web-based infrastructure for identity management for all customer applications and processes. Oracle Access Manager integrates identity and access management across Siebel 7, enterprise resources, and other domains deployed on eBusiness networks. Oracle Access Manager provides the foundation for managing the identities of customers, partners, and employees across Internet applications. These user identities are combined with security policies for protected Web interaction.

This integration adds the following features to Siebel 7 implementations:

- **Oracle Access Manager authentication, authorization, and auditing** services for Siebel 7 applications.
- **Oracle Access Manager single sign-on** (SSO) for Siebel 7 applications and other Oracle Access Manager-protected resources within a single domain or across multiple domains.
- **Oracle Access Manager authentication schemes,** the following schemes provide single sign-on for Siebel 7 applications:
  - **Basic**: Users must enter a user name and password in a window supplied by the Web server.

    This method can be redirected to SSL.'
  - **Form:** This method is similar to the basic challenge method, but users enter information in the custom HTML form.

You can choose the information users must provide in the form that you create.

  - **X509 Certificates**: X.509 digital certificates over SSL.

    A user's browser must supply a certificate.

  - **Integrated Windows Authentication (IWA)**: Users will not notice a difference between an Oracle Access Manager authentication and IWA when they log on to the desktop, open an Internet Explorer (IE) browser, request a Oracle Access Manager-protected Web resource, and complete single sign-on.

  - **Custom**: Additional forms of authentication can be incorporated through use of the Oracle Access Manager Authentication Plug-in API.

- **Session timeout**: Oracle Access Manager enables you to set the length of time that a user session is valid.

- **Ability to use the Identity System for identity management**: The Identity System provides identity management features such as portal inserts, delegated administration, workflows, and self-registration to applications such as Siebel 7.

  The self-registration feature for new users and customers provides flexibility in terms of how much access to provide to people upon self-registration. Identity System workflows enable a self-registration request to be routed to appropriate personnel before access is granted.

  Oracle Access Manager also provides self-service, allowing users to update their own identity profiles.

Siebel 7 Components

This integration involves the following Siebel 7 components.

Siebel Gateway Name Server: The name server provides persistent backing of Siebel server configuration information, including definitions and assignments of component groups and component operational parameters as well as Siebel server connectivity.

Siebel Database Server: The Siebel database server contains the data used by Siebel clients.

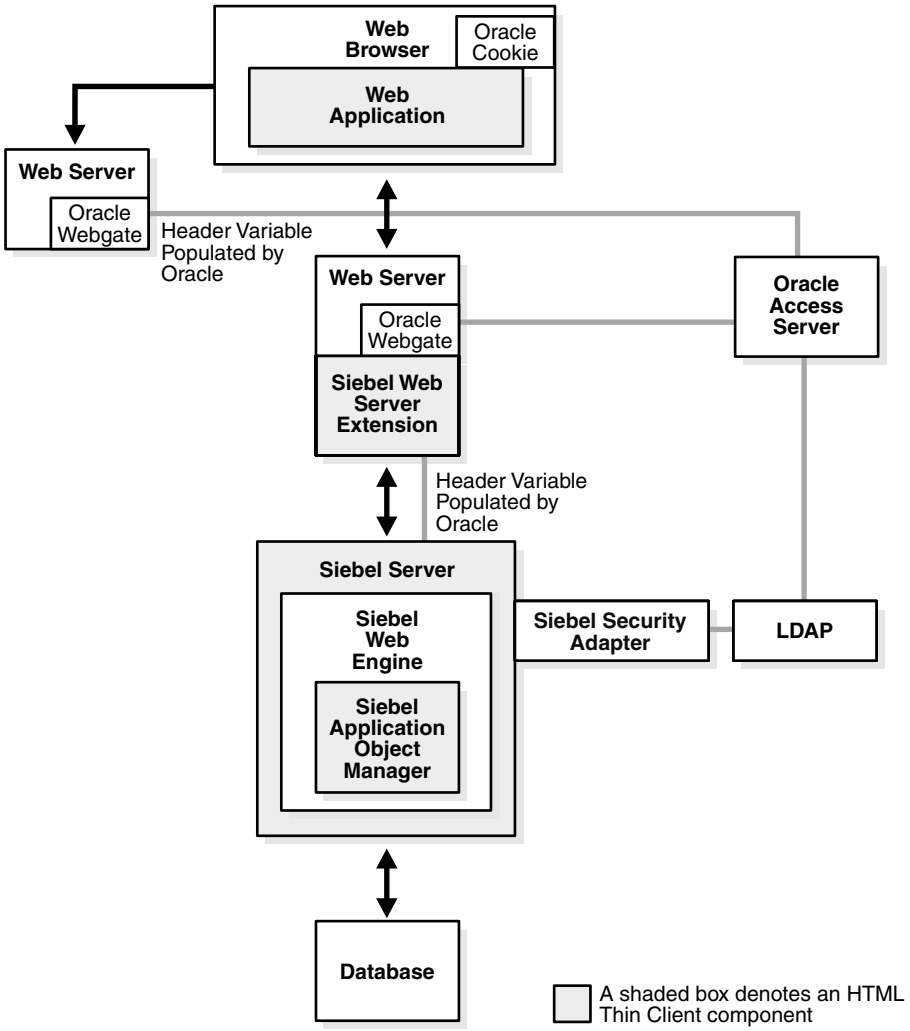
Siebel Server and Siebel Web Server Extension: The Siebel Server along with the Siebel Web Server Extensions supports Siebel Enterprise Web Applications.

Integration Architecture

The preferred method of Web single sign-on with Siebel 7 is achieved by passing a header variable populated with an attribute value that is stored in the LDAP directory. Oracle Access Manager authentication rules permit passing the appropriate HTTP header variable to Siebel 7. The WebGate intercepts the user's HTTP request and checks for a session cookie. If the cookie does not exist or it has expired, the user is challenged for credentials. Oracle Access Manager verifies the credentials, and if the user is authenticated, the WebGate redirects the user to the requested resource and passes the required header variable to the Siebel application. The Siebel application initiates a session which is kept at the Siebel Web Engine.

Figure 7–1 illustrates a scenario where the user authenticates to an Oracle Access Manager-protected resource and is granted access to a Siebel 7 application.

Figure 7–1 Oracle Access Manager Integration with Siebel 7



Process overview: Authentication with the integration

1. A user attempts to access content or an application on a server.
2. WebGate intercepts the request.
3. To determine if the resource is protected, WebGate checks the Access Server for a security policy.

   The security policy consists of an authentication scheme, authorization rules, and allowed operations based on authentication and authorization success or failure.
4. If the resource is protected, WebGate checks for the user's session cookie.

   If a valid session exists, WebGate passes the header variable to the Siebel server. If a valid session does not exist, WebGate prompts the user for credentials.
5. If the credentials are successfully validated, Oracle Access Manager executes the actions that are defined in the security policy and sets an HTTP header variable that maps to the Siebel user ID.
6. WebGate redirects the user to the requested Siebel resource.

7. The Siebel application recognizes the Oracle Access Manager header variable, authenticates the user, and initiates a session.

   The header variable is stored in the Siebel Web Engine. The user can now access any resource that is protected by Oracle Access Manager, for example, a Siebel 7 Web application, without being prompted for credentials.

   If the user is not authorized, the user is denied access and redirected to another URL as determined by the organization's administrator.

Supported Version and Platforms

Any references to specific versions and platforms in this chapter are made for demonstration purposes.

You can find support and certification information at the following URL:

`http://www.oracle.com/technology/documentation/`

You must register with OTN to view this information.

Also, you can see the supported versions and platforms for this integration on Metalink, as follows.

To view information on Metalink

1. In your browser, enter the following URL:

   `https://metalink.oracle.com`

2. Log in to MetaLink.
3. Click the **Certify** tab.
4. Click **View Certifications by Product**.
5. Select the **Application Server** option and click **Submit**.
6. Choose **Oracle Identity Management** and click **Submit**.
7. Click **Oracle Identity Management Certification Information 10g (10.1.4.0.1) (html)** to display the Oracle Identity Management page.
8. Click the link for **Section 6, Oracle Access Manager Certification** to display the certification matrix.

Preparing Your Environment

Complete the following steps to prepare your environment for the integration.

Task overview: Prepare your environment for integration

1. Install a supported directory server according to vendor instructions.
2. Install a supported Web server according to vendor instructions.
3. Configure the Web browser to allow cookies according to vendor instructions.
4. Proceed to "Setting Up Oracle Access Manager Single Sign-on for Siebel Application Server" on page 7-5.

Setting Up Oracle Access Manager Single Sign-on for Siebel Application Server

Setting up Oracle Access Manager single sign-on for Siebel 7 requires the installation and configuration of several Siebel and Oracle Access Manager components.

1. Install and configure Siebel 7, as described in "Setting Up Siebel 7 for integration with Oracle Access Manager".
2. Install Oracle Access Manager and a WebGate, and configure access control policies to protect Siebel resources, as described in "Setting up Oracle Access Manager for Integration with Siebel 7" on page 7-8.
3. Test the integration, as described in "Testing Integration Between Oracle Access Manager and Siebel" on page 7-9.

Setting Up Siebel 7 for integration with Oracle Access Manager

The following procedures describe how to set up Siebel 7 for this integration.

To set up Siebel 7 for integration with Oracle Access Manager

1. Install the following Siebel components, as described in the Siebel documentation:
   a. Siebel Gateway Server
   b. Siebel Server
   c. Siebel Database Server
   d. Siebel Web Server Extension
2. Verify that Siebel eBusiness Applications and Web Server Extension are working properly.
3. Ensure that the Siebel client and the Siebel server are able to communicate with each other through TCP/IP, as described in the Siebel documentation.
4. Add at least three users to LDAP:
   - Test
   - The Siebel Anonymous User
   - The Siebel Application User

   In addition to your regular users, Siebel uses two user accounts from the directory: Anonymous User and Application User. You also need to create an attribute in regular user accounts for storing the Siebel database user information. See the information on creating users in the directory in the *Security Guide for Siebel eBusiness Applications* for details.
5. Add user records in the Siebel database that correspond to the registered users.

   You need a record in the Siebel database that corresponds to the test user that you created in the LDAP directory. You also must confirm that the seed data record exists for the Anonymous User for your Siebel customer or partner application. This database record must match the Anonymous User that you created in the LDAP directory. See the information on adding user records in the Siebel Database in the *Security Guide for Siebel eBusiness Applications* for details.

Table 7–1 on page 7-6 describes the parameters to set for the eapps.cfg file. This file contains configuration details for the Siebel Web Server Extension component. It is located in the \BIN directory where the Siebel Web Server Extension is installed (for

example, C:\sea704\SWEApp). You can add these parameters to the [Default] section or to the Siebel-specific application, for example, [/esales_enu].

Oracle recommends that you add these parameters to the specific Siebel eBusiness application section.

Table 7–1 eapps.cfg Parameters

Parameter and value	Value	Notes
AnonUserName	GuestCST	The anonymous user is a Siebel user with very limited access. It enables a user to access a login page or a page that contains a login form. This user is defined in the Siebel database and must exist in the LDAP directory.
AnonPassword	Ldap	The LDAP password for the anonymous user.
SingleSignOn	TRUE	When this parameter is set to true, the Siebel Web Server Extension Engine (SWSE) operates in WebSSO mode.
TrustToken	HELLO	In a Web single sign-on environment, this token string is a shared secret between the SWSE and the security adapter. It is a measure to protect against spoofing attacks. This setting must be the same on both the SWSE and the security chapter.
UserSpecSource	Header	In a Web single sign-on implementation, this parameter specifies the source from which the SWSE derives the user credentials, as follows: ■ Server—Use if the value is from the Web server name field ■ Header—Use if the variable is in the HTTP request header
UserSpec	SSO_Siebel_User	In a Web single sign-on implementation, this variable name specifies where the SWSE looks for a user's user name in the source provided by UserSpecSource.

The following is an example of a configured eapps.cfg file:

```
[/esales_enu]
SingleSignOn     = TRUE
TrustToken       = HELLO
UserSpec         = SSO_SIEBEL_USER
UserSpecSource   = Header
ConnectString    = siebel.TCPIP.None.None://sdchs24n336:3320/siebel/eSalesObjMgr_
enu
StartCommand     = SWECmd=GotoView&SWEView=Home+Page+View+(eSales)
WebPublicRootDir = c:\19213\eappweb\public\enu
WebUpdatePassword = tieeKaYLjfUBgdi+g==
```

Table 7–2 describes the parameters that you specify in the Siebel Application Parameter File (for example, siebel.cfg).

Table 7–2 Siebel Application Parameter File for the Web Server Extension Component

Parameter	Value	Description
ApplicationUser	Cn=sadmin,cn=users,dc=us,dc=oracle,dc=com	DN of Siebel Application User
ApplicationPassword	Ldap	LDAP password

Table 7–2 (Cont.) Siebel Application Parameter File for the Web Server Extension

Parameter	Value	Description
BaseDN	Cn=users,dc=us,dc=oracle, dc=com	LDAP directory base DN
CRC		CRC code
CredentialsAttribute	Mail	LDAP attribute used to store the user's database credentials
SecAdptDllName	Sscfldap	Security Adapter DLL
HashAlgorithm	RSASHA1	Hash algorithm
HashDBPPwd	FALSE	Should the shared database password be hashed
HashUserPwd	FALSE	Should the user's password be hashed by Siebel
Port	389	LDAP server port
PropagateChange	TRUE	Propagate user changes to an external repository
PasswordExpireWarning	30	Number of days before password expiry, when the user should be warned.
PasswordAttributeType	UserPassword	LDAP attribute used to store the user's password
RolesAttributeType		LDAP attribute used to store the user's responsibilities
ServerName	Ldap.us.oracle.com	LDAP Server Name
SharedCredentialsDN	Cn=sadmin,cn=users, dc=us,dc=oracle,dc=com	DN of LDAP user storing the DB credentials
SiebelUsernameAttributeType	Uid	LDAP attribute used to store the user's user ID
SSLDatabase	C:\oblix-data\oid-key	Path of the SSL database certificate file (required if LDAPS is used)
SingleSignon	TRUE	Is single sign-on enabled
TrustToken	HELLO	Web single sign-on trust token

To set the Siebel Server Configuration Parameters

1. Log in to a Siebel employee application, such as Siebel Call Center, and make one of the following choices from the application-level menu:

   - To set enterprise level parameters, choose View, select Site Map, then select Server Administration and then select Enterprise Configuration.
   - To set server level parameters, choose View, select Site Map, then select Server Administration and then select Servers.
   - To set component level parameters, choose View, select Site Map, then select Server Administration and then select Components.

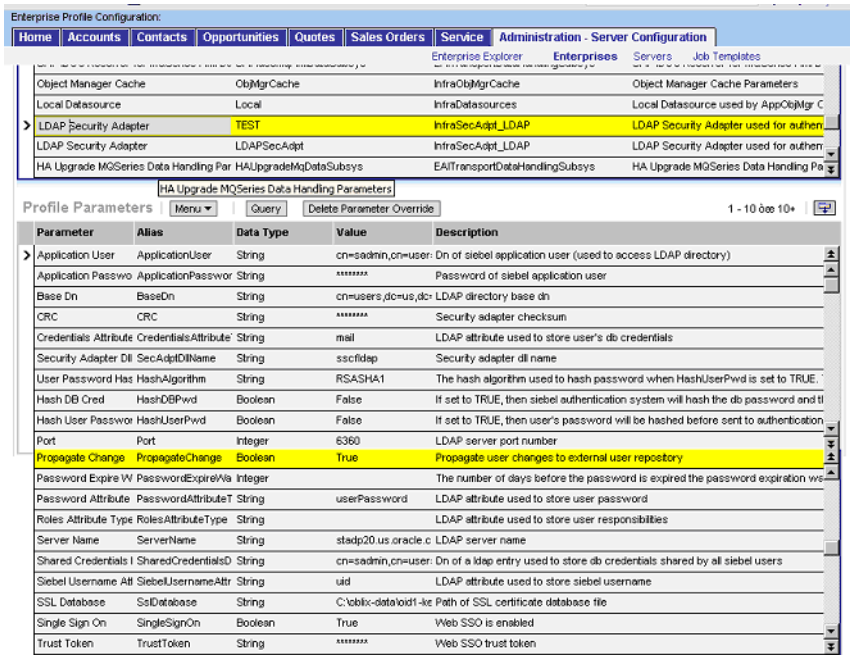
   If you are setting parameters at the server or component level:

   - To set enterprise-level parameters, click the Enterprise Parameters view tab.
   - To set server-level parameters, click the Server Parameters view tab.

- To set component-level parameters, click the Component Parameters view tab.

  Because application-level parameters override enterprise level settings, Oracle recommends that you set the Siebel parameters for SSO integration at the application level.

  The following screen shot illustrates setting Siebel Server configuration parameters:



2. Select a parameter record, edit the Current Value field, and then click Save.
3. Restart the Siebel Server to allow the changes to take effect.

Setting up Oracle Access Manager for Integration with Siebel 7

Setting up Oracle Access Manager for integration with Siebel 7 involves the following steps.

To set up Oracle Access Manager for the integration

1. Install Oracle Access Manager and ensure that you have installed a WebGate on the Web server instance supporting the Siebel Web server extension, as described in *Oracle Access Manager Installation Guide*
2. Synchronize the time on all servers where Siebel and Oracle Access Manager components are installed.

   Each Siebel application has its own document directory. You can either protect each application individually or protect the higher-level directory under which the applications reside.
3. In the Policy Manager, create a policy domain to protect Siebel resources on Web servers where Siebel and the WebGate are installed, as described in the *Oracle Access Manager Access System Administration Guide*

   Oracle Access Manager sets header variables that are passed on to the Siebel eBusiness Application to allow access only to specified users.

4. In the Authorization Rule, choose Actions page of the policy domain protecting the Siebel resource, configure the action to map a Oracle Access Manager Header variable uid to the Siebel uid.

   Note: The Header variable set in the Oracle Access Manager policy should be equal to the value of the UserSpec parameter in the eapps.cfg file.

   In the following example, the uid is mapped to the SSO_SIEBEL_USER HTTP header variable as follows:

   Type: HeaderVar

   Name: SSO_SIEBEL_USER

   Attribute: uid

5. In the Authorization Rules, choose Allow Access page of the policy domain, select the Oracle Access Manager/Siebel users to whom you want to grant access to the resources that are protected by the policy domain.

Testing Integration Between Oracle Access Manager and Siebel

After configuring the integration of Oracle Access Manager with Siebel, you should test for successful Oracle Access Manager authentication and single sign-on with Siebel 7.

The following is a test for single sign-on between a non-Siebel, Oracle Access Manager-protected Web page and Siebel Web Server Extension.

To test Oracle Access Manager single sign-on

1. Create an Oracle Access Manager policy domain to protect a Siebel eBusiness application (for example, eMarketing) and require basic LDAP authentication for it.
2. Open a Web browser and enter the URL for the IIS Web server's main page (http://*hostname*).

   The main page is displayed. User authentication should not be required.
3. Access the Siebel eBusiness application URL for the IIS Web server from the same browser used in step 2.

   Basic authentication should be required.
4. Access the Siebel eBusiness application URL for the IIS Web server from the same browser used in step 2.

   Access to the Siebel eBusiness application should be allowed. The user should not be challenged for credentials.
5. Close the browser and open a new browser session. Access the Siebel eBusiness application URL for the IIS Web server.

   Basic authentication should be required. After the user enters credentials, the Siebel eBusiness application should be displayed.
6. Access the demo document directory URL for the IIS Web server from the same browser user in step 5.
7. Repeat the previous steps for the Sun ONE Web server.

The following is a test of the Oracle Access Manager session timeout.

To test Oracle Access Manager session timeout

1. Configure the Oracle Access Manager session timeout to be five (5) minutes and restart the Web servers.
2. Open a Web browser and the IIS Web server's main page (http://*hostname*).

   The main page is displayed. User authentication should not be required.
3. Access the Siebel eBusiness Application URL for the IIS Web server from the same browser used in step 2.

   Basic authentication should be required. After the user enters credentials, the Siebel eBusiness application should be displayed.
4. Leave the browser window open and idle for more than five minutes.
5. Refresh the browser window using the Refresh button.

   Basic authentication should be required. After the user enters credentials, the Siebel eBusiness Application should be displayed.
6. Repeat step 2 to step 4 for the Sun ONE Web server.

Notes on Integrating in a Multi-Domain Active Directory Environment

There are considerations when configuring this integration in a multi-domain Active Directory environment. When the Siebel application is protected, it obtains the SAMAccountname from the HTTP header variable SSO_SIEBEL_USER. However, the Siebel security adapter performs a lookup in Active Directory to verify the account. In a forest, it is best to perform the query against a single domain controller with a query against port 3268. This is the port that is used for the global catalog.

See the section on configuring LDAP and ADSI security adapters in the *Siebel Security Guide* on the Siebel Bookshelf for details.

Task overview: Configuration in a multi-domain Active Directory environment

1. Enable Siebel to use Active Directory for authentication, configuring the authentication to start at the root of the forest.
2. Configure the Siebel part of the Active Directory search with the global catalog port number as part of the ldap query.

   Add the port number to the hostname in the configuration information, as follows:

   hostname.domainname.com:3268

Configuring Session Logout

You can configure an expiration period for a session by setting a session timeout value in both Siebel and Oracle Access Manager. The timeout values should be the same for both applications. If you configure a timeout value for Siebel that is shorter than the one you configure for Oracle Access Manager, users can re-establish their Siebel session after it times out without providing login credentials.

The rest of this section discusses the following topics:

- Configuring the Siebel Timeout
- Configuring the Oracle Access Manager Session Timeout

Configuring the Siebel Timeout

The following procedure describes configuring the timeout. For users to be asked to re-authenticate after the timeout limit is reached, you must also configure the same timeout value in Oracle Access Manager.

To configure the Siebel timeout

1. Open the eapps.cfg file.

   It is located in the \BIN directory where the Siebel Web Server Extension is installed (for example, C:\sea704\SWEApp).

2. Modify the value for the Set SessionTimeout parameter.
3. Restart the Web server.

Configuring the Oracle Access Manager Session Timeout

The following procedure describes configuring the timeout.

To configure the Oracle Access Manager session timeout

1. Log in to the Access System.
2. From the Access System Console, click Access System Configuration, then click AccessGate Configuration.
3. Search for the AccessGate that you want to configure.
4. Click the link for the AccessGate.
5. Click Modify.
6. Change the value of the Maximum user session time (seconds) field.

   This value should be the same as the one that you set for the Siebel application.

7. Change the value of the Idle session time (seconds) field.

   This value should be the same as the one that you set for the Siebel application.

Configuring the Siebel Logout Behavior

In a Web single sign-on deployment, the user authentication and user management features are the responsibility of Oracle Access Manager. The following features in Siebel 7 are not available in a Web single sign-on environment:

- User self-registration
- Delegated administration
- Login and logout
- Change password

You configure logout functionality for Siebel 7 users by modifying the Siebel Logout link and redirecting the users to the Oracle Access Manager logout page. By doing this, the user is logged out of Oracle Access Manager and by extension from Siebel.

The following procedures describe configuring Siebel to point to the default Oracle Access Manager logout.html page. To ensure that logging out of Siebel is also recognized by Oracle Access Manager, the page that logs users out of Siebel must

contain Oracle Access Manager logout functionality. See the appendix on configuring logout in the *Oracle Access Manager Access System Administration Guide* for details.

The following procedures describe configuring the logout behavior.

To prepare for configuration

1. Create a text file that contains the HTML required to redirect the user to the Oracle Access Manager logout page.

   The following is a URL example:

   ```
   <a href=http://coreidserver.us.orcacle.com/access/oblix/lang/en-us/logout.html>
   ```

   The following is a Javascript example:

   ```
   <html>
     <head>
       <script language="Javascript">
       <!--//

   window.location.href=http://coreidserver.us.oracle.com/access/oblix/lang/en-us/
   logout.html;
       //-->
          </script>
     </head>
   </html>
   ```

2. Copy the file as follows:

   $siebelroot/siebsrvr\WEBTEMPL*name*.swt

   Where name is the name of the file that you created in the previous step, for example, coreidlogout.swt.

3. Stop the Siebel server process.
4. Start Siebel Tools.

To create a new project

1. In the Object Explorer window, click Project.
2. Select Edit.
3. Select New Record.
4. Enter the name of the file that contains the redirection information as the name for the new record.

   Do not include the ".swt" extension. In the previous procedure, this name was coreidlogout.

5. Select Locked.

To create a Web template

1. In the Object Explorer window, click Web Template.
2. Add a new record.

   Use the name of the file with the redirection information. Do not include the ".swt" extension.

   In a previous procedure, an example name of coreidlogout was provided.

3. Enter the Project parameter.

As the name of this parameter, use the name of the file with the redirection information. Do not include the ".swt" extension.

In a previous procedure, an example name of coreidlogout was provided.

4. Specify Web Page Template for the Type parameter.

To create a Web template file

1. Expand the Web Template tree.
2. Click Web Template File.
3. Add a record that is named using the name of the file with the redirection information.

   Do not include the ".swt" extension. In a previous procedure, an example name of coreidlogout was provided.
4. Enter the name of the file with the redirection information, *including* the ".swt" extension, as the Filename parameter.

To create a Web page for logout

1. In the Object Explorer window, click Web Page.
2. Add a record that is named using the name of the file with the redirection information.

   Do not include the ".swt" extension. In a previous procedure, an example name of coreidlogout was provided.
3. Enter the name of the file with the redirection information as the Project parameter.

   Do not include the ".swt" extension. In a previous procedure, an example name of coreidlogout was provided.
4. Select the name of the file with the redirection information as the Web Template parameter.

   In a previous procedure, an example name of coreidlogout was provided.

To complete logout configuration

1. To lock the application project for each project where you want to modify the logout behavior, in the Object Explorer window, click Project.
2. Locate the appropriate project.
3. Select Locked.
4. In the Application window, select the Siebel module to be configured.

   Each module must be configured separately.
5. Scroll to the right and locate the Logoff Acknowledgement Web Page parameter.

   Make a note of this value before changing it.
6. Select the name of the file with the redirection information.

   In a previous procedure, an example name of coreidlogout was provided.
7. Compile the changes.
8. Restart the Siebel Server and the Web server.

8

Integrating PeopleSoft

PeopleSoft is a Web-based eBusiness application suite that provides human resources, supply chain, CRM, analytics, portal, and other applications. This chapter describes the integration of Oracle Access Manager's single sign-on capabilities with PeopleSoft PeopleTools and applications.

This chapter covers the following topics:

About the Integration with PeopleSoft

This integration provides a secure Internet infrastructure for identity management for PeopleSoft's customer applications and processes. Oracle Access Manager provides identity and access management across PeopleSoft applications, enterprise resources, and other domains that are deployed on eBusiness networks. Oracle Access Manager provides the foundation for managing the identities of customers, partners, and employees across Internet applications. These user identities are protected by security policies for Web interaction.

This integration adds the following to PeopleSoft implementations:

- **Oracle Access Manager authentication, authorization, and auditing** services for Siebel 7 applications.
- **Oracle Access Manager single sign-on** (SSO) for PeopleSoft applications and other Oracle Access Manager-protected resources in a single domain or across domains.
- **Oracle Access Manager authentication schemes** that provide single sign-on for PeopleSoft applications:
  - **Basic**: Users enter a user name and password in a window supplied by the Web server.

This method can be redirected to SSL.

  - **Form:** Similar to the basic challenge method, users enter information in a custom HTML form.

    You choose the information that users must provide in the form.

  - **X509 Certificates**: X.509 digital certificates over SSL.

    A user's browser must supply a certificate.

  - **Integrated Windows Authentication (IWA)**: Users will not notice a difference between an Oracle Access Manager authentication and IWA when they log on to the desktop, open an Internet Explorer (IE) browser, request a Oracle Access Manager-protected Web resource, and complete single sign-on.

  - **Custom**: You can use other forms of authentication through the Oracle Access Manager Authentication Plug-in API.

- **Session timeout**: Oracle Access Manager enables you to set the length of time that a user session is valid.
- **Ability to use the Identity System for identity management**: The Identity System provides identity management features such as portal inserts, delegated administration, workflows, and self-registration to applications such as PeopleSoft.

  You can determine how much access to provide to people upon self-registration. Identity System workflows enable a self-registration request to be routed to appropriate personnel before access is granted.

  Oracle Access Manager also provides self-service, allowing users to update their own identity profiles.

PeopleSoft Components

This integration involves the following PeopleSoft components.

PeopleSoft Application Server: The application server is the core of PeopleSoft Pure Internet Architecture (PIA). An application server maintains the SQL connection to the database for browser requests and the PeopleTools development environment in Microsoft Windows. It runs business logic and issues SQL to the database server.

The application server consists of numerous PeopleSoft services and server processes. Just as different elements make up the physical environment in which an application server operates, for example, database servers and Web servers, a variety of elements operate on the application server, enabling it to respond effectively to multiple transaction requests and handle transaction processing, system scaling, browser requests, and so on.

PeopleSoft Database Server: The database server houses a database engine and the PeopleSoft application database. The database includes all the application's object definitions, system tables, application tables, and data. The database server must run one of the PeopleSoft-supported RDBMS and operating system combinations.

Multiple application servers can connect to the database server. The database server simultaneously handles the application server connections, development environment connections, and batch programs running against it.

PeopleSoft Internet Architecture: PeopleSoft Pure Internet Architecture enables Internet application deployment through a browser, and enables you to take

advantage of PeopleSoft intranet solutions, Internet solutions, and integration technologies.

PeopleSoft Pure Internet Architecture runs seamlessly in portals created and managed by PeopleSoft portal technology.

PeopleTools portal technology is built on top of PeopleSoft Pure Internet Architecture and enables you to easily access and administer multiple content providers, including PeopleSoft databases such as PeopleSoft CRM and HRMS, as well as non-PeopleSoft content. It enables you to combine content from these multiple sources and deliver the result to users in a unified, simple-to-use interface.

PeopleSoft Integration Architecture

PeopleSoft has a configurable authentication mechanism that allows it to authenticate a user against the following:

- Native tables
- LDAP
- Custom plug-ins, including the ability to read HTTP Headers

Single sign-on with PeopleSoft involves the following:

- Protecting PIA with a WebGate.
- Populating a header variable with an attribute value that is stored in the LDAP directory used by Oracle Access Manager.
- Writing PeopleCode to read the header variable and generate the PS_TOKEN.

  A cookie is generated by PIA every time a user successfully logs in. It is used to enable single sign-on with other PeopleSoft applications.
- Configuring PeopleSoft to invoke the PeopleCode as part of the authentication process, overriding the default authentication mechanism.

Single Sign-On Process

There are two ways to render PeopleSoft application pages for the user:

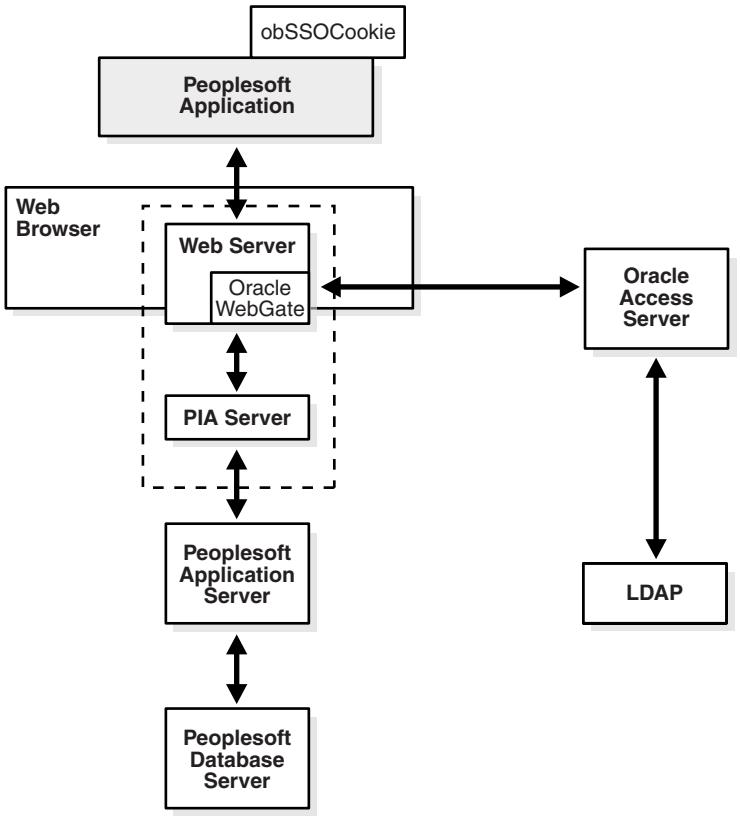
- Using a PIA application server that has an HTTP Server and a J2EE container (required for the PIA servlets and Java code to run), for example, Oracle Application Server 10g.
- Using a Java-enabled application server and setting up an HTTP server as a reverse proxy.

  For example, you can use WebLogic as the PIA application server and iPlanet as the HTTP Server.

In both cases, the WebGate must be installed on the HTTP Server and it must be configured to protect the PeopleSoft URLs. See your PeopleTools version-specific documentation for the URL format.

The user accesses a PeopleSoft application using the Web browser. The WebGate intercepts the user's HTTP request and checks for an obSSOCookie. If the cookie does not exist or it has expired, the user is challenged for credentials. Oracle Access Manager verifies the credentials, and if the user is authenticated, the WebGate redirects the user to the requested resource and passes the required header variable to PeopleSoft. The header variable is read by PeopleSoft and used to generate the PS_TOKEN. Table 8–1 illustrates the integration environment and process flow:

Figure 8–1 PeopleSoft Single Sign-On Using Oracle Access Manager



Process Overview: Single Sign-On with PeopleSoft

1. A user attempts to access a PeopleSoft application.
2. A Webgate that is deployed on the PeopleSoft HTTP Server intercepts the request.
3. The Webgate checks the Access Server to determine if the resource (PeopleSoft URL) is protected.

   The security policy consists of an authentication scheme, authorization rules, and allowed operations based on authentication and authorization success or failure.
4. If a valid session does not exist and the resource is protected, WebGate prompts the user for credentials.
5. If the credentials are validated, Oracle Access Manager executes the actions defined in the security policy for the PeopleSoft resource and sets a HTTP Header variable that maps to the PeopleSoft user ID.
6. If a valid session cookie exists and if the user is authorized to access the resource, WebGate redirects the user to the requested PeopleSoft resource.
7. PeopleSoft receives the request for the PeopleSoft resource and executes the PeopleCode defined in its authentication configuration.
8. The PeopleCode reads the HTTP header variable and sets that value as the logged-in PeopleSoft user.

   It then generates the PS_TOKEN, which contains the same information.
9. PeopleSoft generates the application pages, subject to further authorization verification within PeopleSoft.

Supported Version and Platforms

This chapter describes the integration of Oracle Access Manager 10*g* (10.1.4.0.1) with PeopleTools 8.47 and PeopleSoft Applications (HCM 8.9). However, any references to specific versions and platforms in this chapter are for demonstration purposes.

You can find support and certification information at the following URL:

`http://www.oracle.com/technology/documentation/`

You must register with OTN to view this information.

Also, you can see the supported versions and platforms for this integration on Metalink, as follows.

To view information on Metalink

1. In your browser, enter the following URL:

   `https://metalink.oracle.com`

2. Log in to MetaLink.
3. Click the **Certify** tab.
4. Click **View Certifications by Product**.
5. Select the **Application Server** option and click **Submit**.
6. Choose **Oracle Identity Management** and click **Submit**.
7. Click **Oracle Identity Management Certification Information 10g (10.1.4.0.1) (html)** to display the Oracle Identity Management page.
8. Click the link for **Section 6, Oracle Access Manager Certification** to display the certification matrix.

Preparing Your Environment

Before you can integrate Oracle Access Manager with PeopleSoft, complete the following steps to prepare your environment.

Task overview: Preparing for the PeopleSoft integration

1. Install a supported directory server, according to vendor instructions.
2. Install and configure Oracle Access Manager using the directory server from the previous step as the LDAP repository.
3. Ensure that the PeopleSoft application pages are delivered using an HTTP Server.
4. Configure the Web browser to allow cookies, according to vendor instructions.
5. Follow the instructions in "Setting Up Oracle Access Manager Single Sign-On for PeopleSoft" on page 8-5.
6. Follow the instructions in "Setting up PeopleSoft for Single Sign-On with Oracle Access Manager" on page 8-8.

Setting Up Oracle Access Manager Single Sign-On for PeopleSoft

The following procedure describes setting up Oracle Access Manager single sign-on for PeopleSoft.

See also: For more information on creating policy domains, policies, and associated authentication and authorization schemes, see the *Oracle Access Manager Access System Administration Guide*.

To set up Oracle Access Manager for the PeopleSoft integration

1. Install and configure Oracle Access Manager on a supported platform, using a supported LDAP server.

   See the *Oracle Access Manager Installation Guide* for details.

2. Install a WebGate on the PeopleSoft HTTP Server.

   If your PIA application server is WebSphere or WebLogic, install an HTTP server, for example, Apache, iPlanet, or Oracle HTTP Server, and then configure PIA so that PeopleSoft application pages are accessed and rendered through the HTTP server. You then need to protect the HTTP server with the appropriate WebGate designed for that HTTP server.

3. Create a host identifier for the PeopleSoft HTTP Server.

   From the Access System landing page, select the Access System Console, click Access System Configuration, click Host Identifiers, and add information about the server.

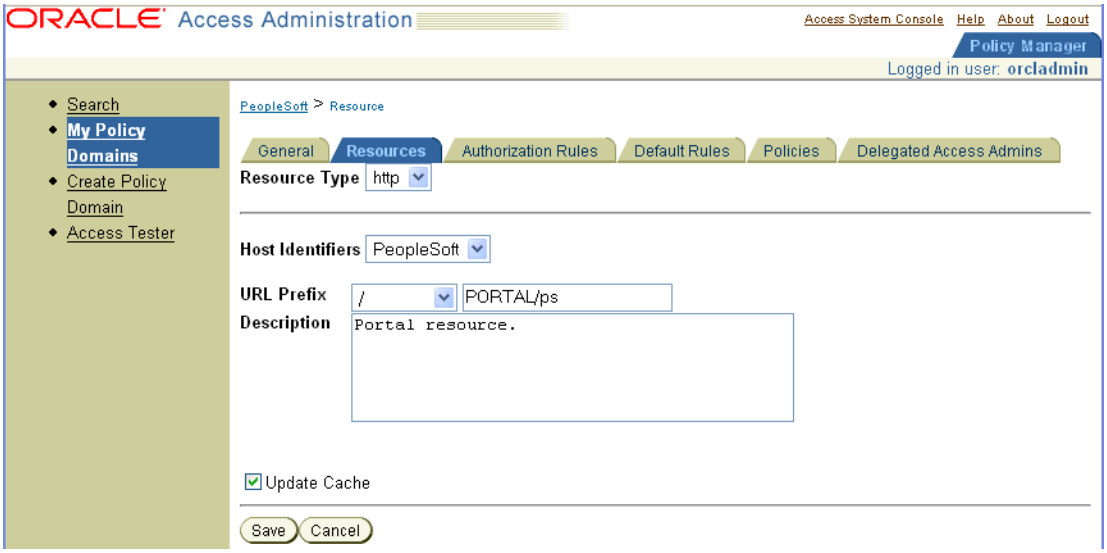
4. Create a policy domain and policies to restrict access to PeopleSoft URLs.

   From the Access System landing page, select the Policy Manager, then click create Policy Domain, and define a policy domain and policies.

   The policy domain should protect all PeopleSoft URLs that users access. For example, if you use PeopleSoft Portal to consolidate access to various PeopleSoft applications, the policy must protect the portal and application URLs.

   URL prefix formats are specific to your PeopleSoft implementation. For example, the version 8.47 URLs have the format /PORTAL/ps, HRMS/ps, and so on.

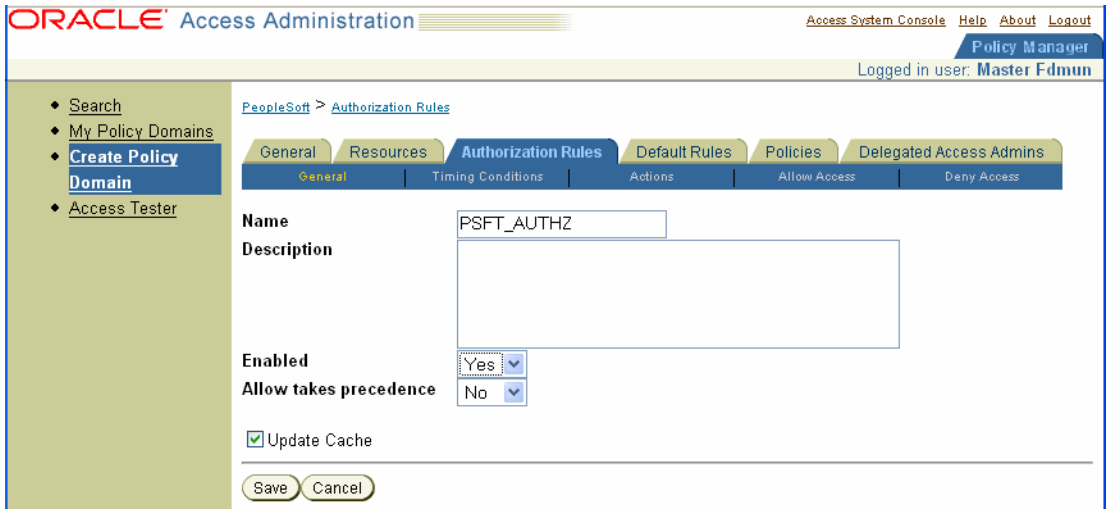
   The following screen shot illustrates a PeopleSoft policy domain.



5. After saving the new policy domain, define an authorization rule that controls who has access to the PeopleSoft resources.

If you are already viewing the new policy domain, click Authorization rules. Otherwise, click My Policy Domains, click the link for the policy domain, and click Authorization rules.

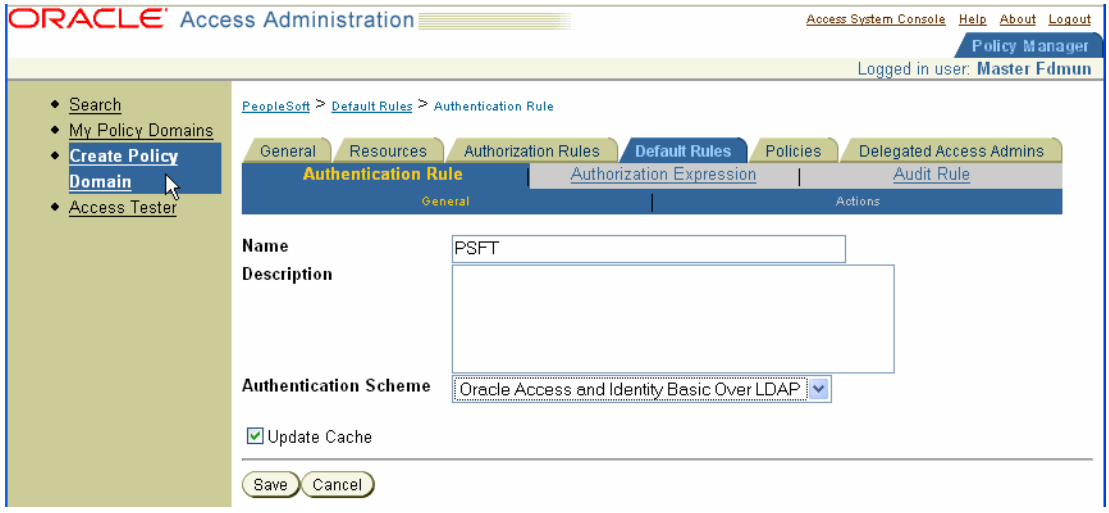
The following is a screen shot of an Authorization Rules configuration page.



6. Define an authentication rule, for example, Oracle Access and Identity Basic Over LDAP, form authentication, and so on.

   If you are already viewing the new policy domain, click Default Rules, then click Authentication Rule. Otherwise, click My Policy Domains, click the link for the policy domain, and click Default Rules, then click Authentication Rule.

   The following is a screen shot of an Authentication Rule configuration page.



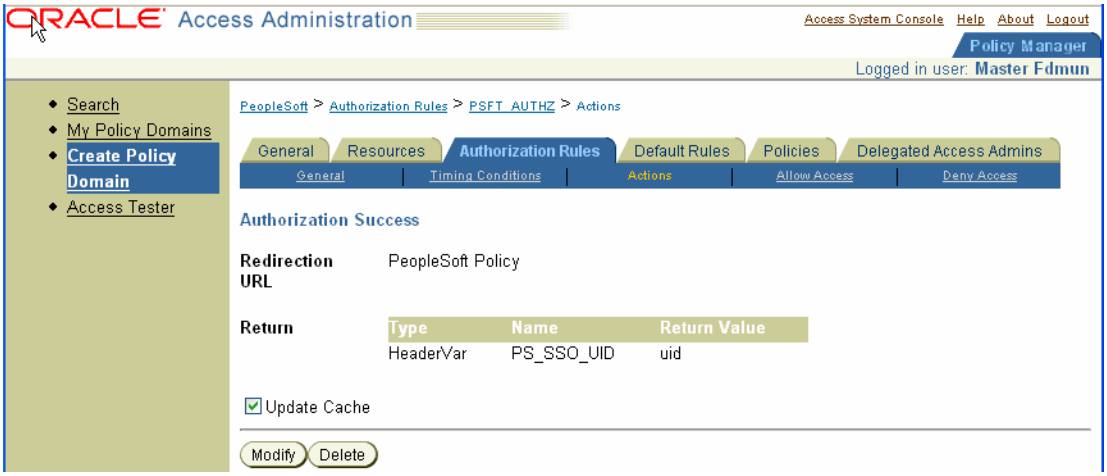
7. Define an authorization action that sets a custom HTTP header variable upon successful authorization.

   If you are already viewing the new policy domain, click Authorization Rules, then click Actions. Otherwise, click My Policy Domains, click the link for the policy domain, click Authorization Rules, then click Actions.

The action should contain a redirection URL for authorization success.

The header variable should contain a value that maps to the PeopleSoft user ID.

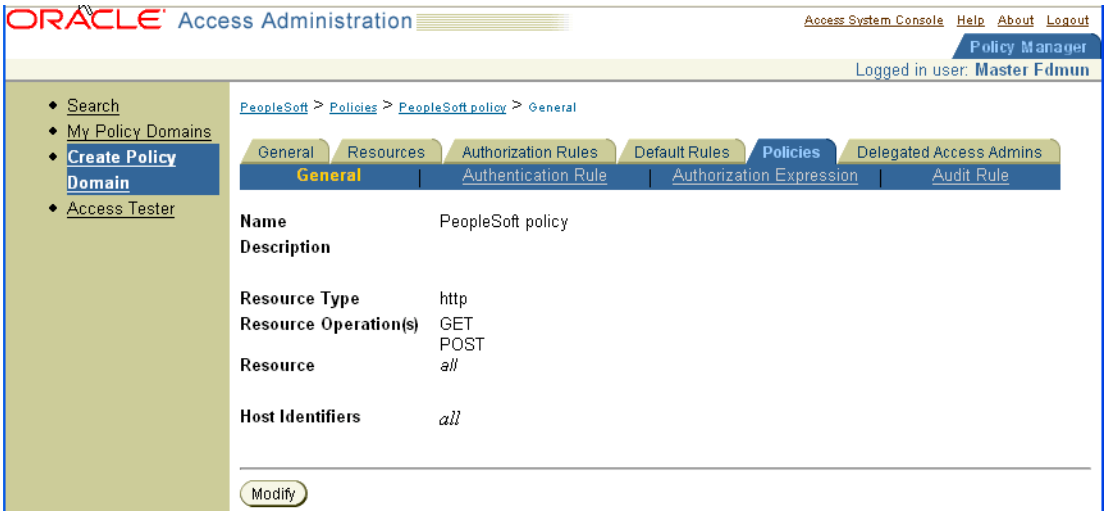
The following is a screen shot of a saved authorization action.



8. Define an access policy and add the PeopleSoft resources to it.

   If you are already viewing the new policy domain, click Policies, then click Add. Otherwise, click My Policy Domains, click the link for the policy domain, click Policies, then click Add.

   The following is a screen shot of a saved policy.



Setting up PeopleSoft for Single Sign-On with Oracle Access Manager

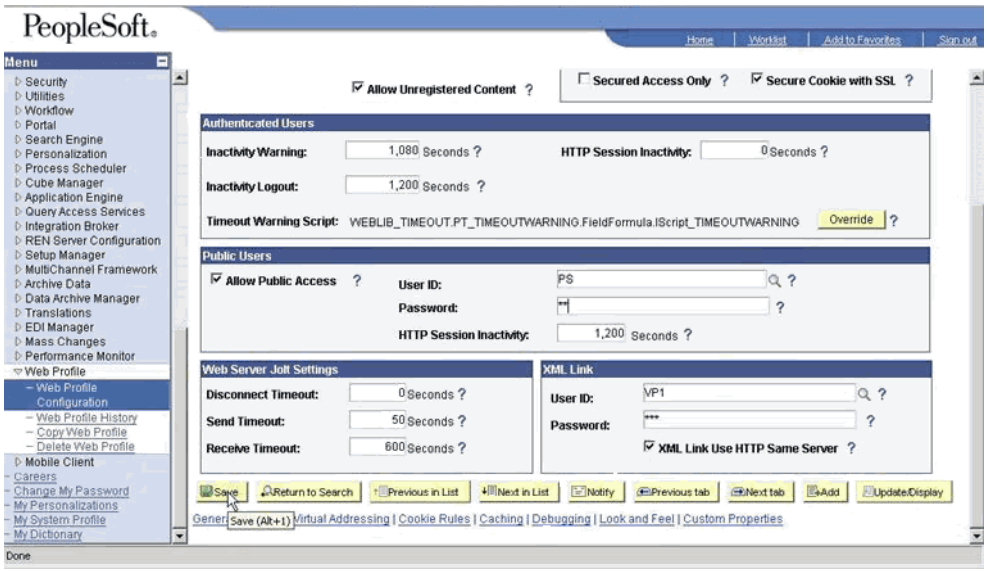
The following procedure describes setting up PeopleSoft for integration with Oracle Access Manager.

To set up PeopleSoft for integration with Oracle Access Manager

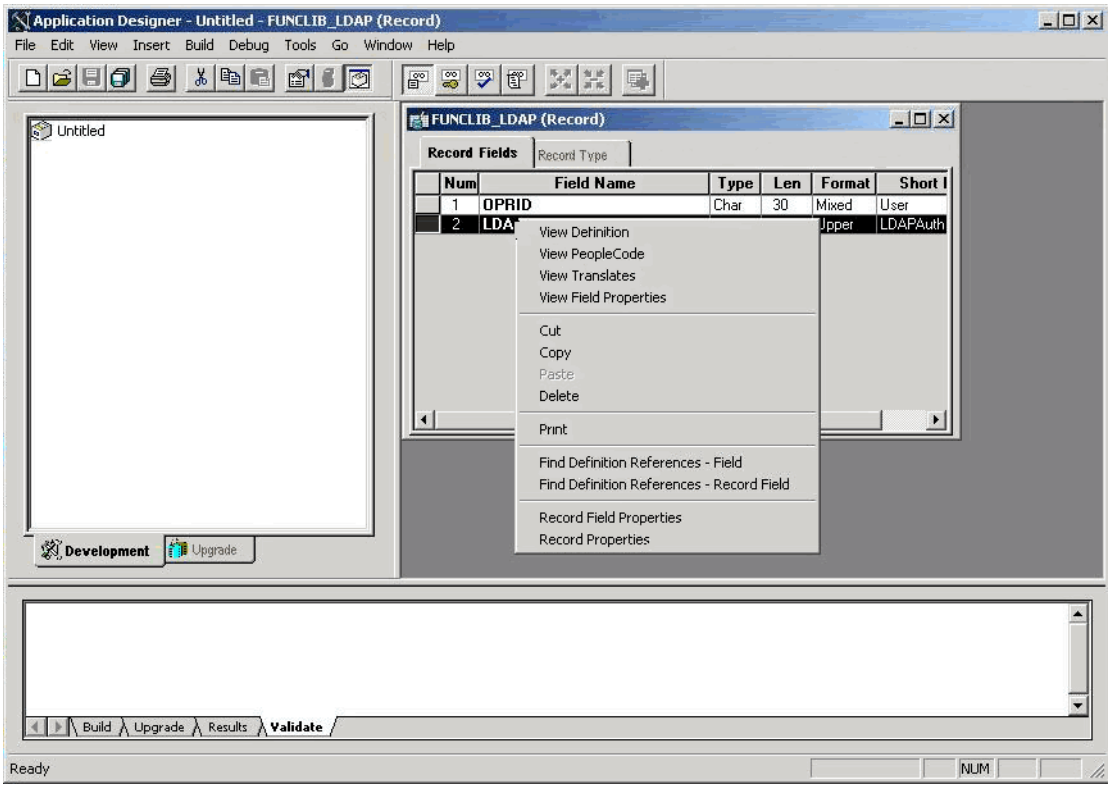
1. Configure PeopleSoft to allow public access.

To do this, you modify the Web profile for the PeopleSoft environment that you are securing. In PeopleTools 8.47, the Web profile definition contains all the information that PIA needs to render the PeopleSoft pages.

To allow public (unprotected) access to PeopleSoft, PIA requires a PeopleSoft application user. PIA logs using this application user and renders the PeopleSoft pages. In the Web profile, provide a user ID and password of a PeopleSoft user. Ensure that this user has minimal applications privileges.



2. From the PeopleTools Application Designer, open the FUNCLIB_LDAP record.



3. Modify the PeopleCode for the LDAPAUTH field event, as follows.

   In the getWWWAuthConfig() function, replace the value that is assigned to the &defaultUserId with the user ID that you defined in the Web profile.

```
Function getWWWAuthConfig()
   /* Begin - New code to support Oracle Access Manager */
   /* authentication */
   /* NOTE: replace v01475 with the default userid */
   &defaultUserId = "v01475";
   /*   &defaultUserId = "";*/
   /* End - New code to support Oblix authentication */
End-Function;
```

4. Add a new function that will read the HTTP header variable that is set by Oracle Access Manager, as indicated in the following code sample.

```
Function Oracle_Access_Manager_Authentication()
   /* Note: Following file will written to the default path.*/
   /* Usually appserv/<instance name>/files */
     &logfile = Getfile("oamaccess.log","A");

 If &logfile.Isopen then
    &logfile.Writeline("INFO: Netpoint_Authentication_Profile started");
        &logfile.Writeline(String(%Datetime));
        &logfile.Writeline("authMethod: " &authMethod));
 End-If;

If %PSAuthResult = True And
&authMethod <> "LDAP" And
&authMethod <> "COREID" And
&authMethod <> "SSO" Then
       If &logfile.Isopen then
          &logfile.Writeline("INFO: AuthResult and AuthMethod ok.");
       End-If;

       getWWWAuthConfig();

       If &logfile.Isopen then
         &logfile.Writeline("INFO: After get wwwauthconfig.");
         &logfile.Writeline(&defaultUserId);
       End-If;

       If %SignonUserId = &defaultUserId Then
         If &logfile.Isopen then
            &logfile.Writeline("INFO: %SignonUserId = &defaultUserId ");
         End-If;
         &userID = %Request.GetHeader("PS_SSO_UID");

         If &logfile.Isopen then
            &logfile.Writeline("INFO: After getting HTTPheader");
            &logfile.Writeline(&userId);
         End-If;

         If &userID <> "" Then
           If &bConfigRead=False then
             getLDAPConfig();
             If &logfile.Isopen then
                &logfile.Writeline("INFO: After getLDAPConfig()");
                &logfile.Writeline(&userId);
             End-If;
```

```
                &GlobalUserID=&userID;
                If &logfile.Isopen then
                    &logfile.Writeline("INFO: After DNTold");
                    &logfile.Writeline(&GlobalUserId);
                End-If;
                If &GlobalUserID <> "" Then
                  SetAuthenticationResult(True,Upper(&GlobalUserID),"",False);
                  &authMethod = "WWW";
                  If &logfile.Isopen then
                     &logfile.Writeline("INFO: Userid is valid");
                     &logfile.Writeline(&GlobalUserID);
                  End-If;
                End-If;
            End-If;
        Else
           If &logfile.Isopen then
             &logfile.Writeline("Warning: %SignonUserId not = &defaultUserId");
             &logfile.Writeline(%SignonUserId);
             &logfile.Writeline(&defaultUserId);
           End-If;
        End-If;
     Else
        If &logfile.Isopen then
         /*Note: Next line sometimes wraps due to length - fix if necessary */
         &logfile.Writeline("ERROR: %PSAuthResult = True And");
         &logfile.Writeline("&authMethod <> ""LDAP"" And ");
         &logfile.Writeline("&authMethod <> ""SSO""");
         &logfile.Writeline(%PSAuthResult);
         &logfile.Writeline(&defaultUSerId);
         &logfile.Writeline(%Request.GetHeader("PS_SSO_UID"));
        End-If;
  End-If;
  If &logfile.Isopen then
  &logfile.Writeline("INFO: Oracle_Access_Manager_Authentication_Profile exit");
       &logfile.close();
    End-If;
End-If;
End-Function;
```

5. Configure PeopleSoft to execute the function in the previous step when a user tries to access PeopleSoft pages.

   To do this, modify the Signon PeopleCode function, enable the function in the previous step, and disable all other functions.

6. Restart the PeopleSoft Application Server and the PeopleSoft HTTP Server.

Configuring Single Signoff for PeopleSoft

The following procedure describes how to configure single signoff for PeopleSoft.

To configure single signoff for PeopleSoft

1. On the Web server where PIA is installed, locate and open signin.html.
2. Copy signin.html to a file named signout.html.
3. Open signout.html in an editor and add the following information to it:

```
<HEAD>
<meta HTTP-EQUIV='Refresh' CONTENT='1; URL=http://<Peoplesoft Web
Server>/<Virtual PATH not protected by Access Manager>/logout.html'>
```

```
</HEAD>
</HTML>
```

4. Place the file logout.html that is provided with your Oracle Access Manager installation in a virtual path that is not protected by a WebGate.

   The following is the default path:

   ```
   Policy_Manager_install_dir/access/oblix/lang/en-us/logout.html
   ```

   Where *Policy_Manager_install_dir* is the directory where the Policy Manager is installed.

   The file contains Javascript that deletes the obTEMC cookie. See the appendix on configuring logout in the *Oracle Access Manager Access System Administration Guide* for details.

5. In your browser, in PIA, select PeopleTools, then select Web Profile, Web Profile Configuration, Look and Feel.

6. In the Signon/Logout Pages group box, change the value of the Logout Page field to signout.html.

Troubleshooting the PeopleSoft Integration

An adminstrator may need to log in directly to PeopleSoft in the event that Oracle Access Manager is unavailable. This requires a workaround that opens a port for PeopleSoft.

The following example is specific to the Apache Oracle HTTP Server for PeopleSoft running on port 7777. In this example, the WebGate is configured to protect PeopleSoft only on port 7777, leaving other ports unprotected.

The following example assumes that you already have a complete integration of Oracle Access Manager and PeopleSoft.

To configure direct login to PeopleSoft on an Apache Oracle HTTP Server

1. Verify that listen port for the application that you want to protect with a WebGate exists in the httpd.conf.

   For example if port 7778 is the port for a WebCache that points to the actual application on port 7777, the following may be configured:

   ```
   Port 7777
   Listen 7778
   ```

2. Comment out the default `LocationMatch` found in the Oblix or WebGate block in httpd.conf.

3. Add a new `VirtualHost` and `LocationMatch` directive at the end of httpd.conf.

   The following is an example of modified httpd.conf file.

   ```
   <!--[if !supportLists]-->1. <!--[endif]-->Comment out the following from the
   webgate section:

   <LocationMatch "/*">
        AuthType Oblix
        require valid-user
   </LocationMatch>

   <!--[if !supportLists]-->2. <!--[endif]-->Add the following to the bottom of
   ```

the file:

```
NameVirtualHost *:7777
<VirtualHost *:7777>
    ServerName psw01.foo.com
    <LocationMatch "/*">
      AuthType Oblix
      require valid-user
    </LocationMatch>
</VirtualHost>
```

9

Integrating Oracle E-Business Suite

Oracle E-Business Suite Release 11i and 12 are comprehensive suites of business applications for the enterprise. They include applications such as the following:

- Customer relationship management, interaction center
- Financials, contracts, sales, service, order management, marketing, product lifecycle management
- Logistics, transportation management, maintenance, manufacturing
- Intelligence, learning management, customer data management, corporate performance
- Supply chain execution, supply chain management, supply chain planning, procurement
- Human resources

This chapter describes using Oracle Access Manager to manage user authentication and authorization when a user logs in to Oracle eBusiness Suite.

This chapter covers the following topics:

About the Integration with Oracle E-Buisiness Suite

This integration enables single sign-on between Oracle Access Manager and Oracle e-Business Suite applications. It requires the following components:

- Configuring single sign-on between OracleAS Single Sign-On and Oracle Access Manager
- Configuring single sign-on between OracleAS Single Sign-On and Oracle e-Business Suite applications

Note that OracleAS Single Sign-On requires Oracle Internet Directory (OID) as its identity store. You can also use OID with Oracle Access Manager. See the *Oracle Access Manager Installation Guide*. However, you can configure OID for OracleAS Single Sign-On and use a different identity store for Oracle Access Manager. If different directories are used, you will need to ensure that the identities for Oracle e-Business Suite applications are kept in sync in the directories. There must be a unique identifier that can link a single user entry in both directories.

For information on integrating Oracle Access Manager and OracleAS Single Sign-On, see "Integrating with Oracle Application Servers" on page 4-1.

For information on integrating OracleAS Single Sign-On and the Oracle e-Business Suite, see Metalink article 261914.1 on Metalink, as follows.

To find an article on Metalink

1. Go to the following URL:

   https://metalink.oracle.com

2. Click the Knowledge tab.
3. In the Quick Find menu, select Knowledge Base and enter the article number in the associated entry field.

Part II

Integration with Third-Party Applications

This part provides all the information you need to successfully integrate Oracle Access Manager with several third-party applications.

Part II contains the following chapters:

10

Integrating the Security Provider for WebLogic SSPI

This chapter describes how to use Oracle Access Manager with BEA WebLogic running in a Security Service Provider Interface (SSPI) implementation. WebLogic provides an environment for creating, integrating, securing, and managing distributed Java applications. The Security Provider for WebLogic SSPI (Security Provider) ensures that only appropriate users and groups can access Oracle Access Manager-protected WebLogic resources to perform specific operations. The Security Provider also enables you to configure single sign-on between Oracle Access Manager and WebLogic resources.

This chapter covers the following topics:

Note: If you are installing the Oracle Access Manager 10.1.4.2 patch release, you must configure the Weblogic Authorization Provider and Adjudication Provider before applying the patch.

About the Security Provider

The Security Provider for WebLogic SSPI provides authentication, authorization, and single sign-on across J2EE applications that are deployed in the BEA WebLogic platform. The Security Provider enables WebLogic administrators to use Oracle Access Manager to control user access to business applications.

The Security Provider provides authentication to BEA WebLogic Portal resources and supports single sign-on between Oracle Access Manager and BEA WebLogic Portal Web applications. Apart from this, the Security Provider also offers user and group management functions.

Note: The integration with WebLogic supports only one policy domain for each WebLogic server instance. All WebLogic policies must reside in this single policy domain.

WebLogic and Oracle Access Manager Integration Points

The WebLogic security framework provides Security Service Provider Interfaces (SSPIs) to protect J2EE applications. The Security Provider takes advantage of these SSPIs, enabling you to use Oracle Access Manager to protect WebLogic resources via:

- User authentication
- User authorization
- Role mapping

The Security Provider consists of several individual providers, each of which enables a specific Oracle Access Manager function for WebLogic users:

Authenticator: This security provider uses Oracle Access Manager authentication services to authenticate users who access WebLogic applications. Users are authenticated based on their credentials, such as user name and password.

The security provider also offers user and group management functions. It enables the creation and deletion of users and groups from the BEA WebLogic Server. It also provides single sign-on between WebGates and portals.

Identity Asserter: Like the Authenticator, this security provider uses Oracle Access Manager authentication services to validate already-authenticated Oracle Access Manager users using the ObSSOCookie and to create a WebLogic-authenticated session.

Whether you use the Authenticator or the Identity Asserter depends on your deployment scenario. See "Integration Architecture" on page 10-3 for details.

Authorizer: This security provider uses Oracle Access Manager authorization services to authorize users who are accessing a protected resource. The authorization is based on Oracle Access Manager policies.

Role Mapper: This security provider returns security roles for a user. These roles are defined in Oracle Access Manager, and they are provided by Oracle Access Manager using return actions on a special authentication policy. This authentication policy contains a resource with a URL prefix of /Authen/Roles. Role Mapper maps these roles to predefined security roles in WebLogic.

Deployment Provider: This security provider monitors the applications that are deployed or undeployed on the WebLogic Server and writes information about these applications to either NetPointDeployPolicy.txt or NetPointUndeployPolicy.txt. The files store details about the resources, like resource type, Role names to which access is

allowed and the like. Details will vary based on the type of the resource. The NetPointPolicyDeployer tool uses these files to create access policies in the Policy Manager.

A special authorization rule for administrators in Oracle Access Manager provides access to the WebLogic applications described in NetPointDeployPolicy.txt. This means that the policies created by the NetPointPolicyDeployer tool in the Policy Manager will have authorization rules in which only administrator have allow access rights.

Integration Architecture

With the Security Provider, you can use Oracle Access Manager to protect WebLogic resources, including Web applications, EJBs, JNDIs, and so on. You can configure single sign-on for Web applications, such that a user who has authenticated to WebLogic can access Oracle Access Manager-protected resources (including WebLogic and non-WebLogic resources) without re-authentication. You can also configure single sign-on such that a user who has authenticated to Oracle Access Manager can access WebLogic resources without re-authentication.

> **Note:** With the WebLogic Portal, the Security Provider is used only to provide authentication to WebLogic resources. Role Mapping and Authorization are taken care by the Weblogic Portal. Therefore, the Authorizer and Role Mapper must not be configured in NetPointRealm.

The following applies only to the WebLogic Server. For Weblogic Portal authentication, see "Authentication for the Portal" on page 10-7.

You can use the integration with WebLogic in the following ways:

- To provide authentication for mixed Web and non-Web resources:

  This scenario assumes that the environment configuration does not use a proxy server running a WebGate. In this scenario, you protect resources in Oracle Access Manager using username and password authentication. This method can be used to protect both HTTP resources on the Web and to protect resources such as EJBs, JNDIs, and other types of applications. In this type of scenario, all J2EE Web application deployment descriptors must be configured to be BASIC or FORM. This method requires use of the Authenticator service provider.

  See "Authentication for Mixed Web and Non-Web Resources" on page 10-4 for details.

- To provide authentication for Web resources only using a proxy server with WebGate installed:

  In this scenario, you protect resources using a variety of authentication schemes, such as form, SecurID, and so on. For this type of authentication to work, all J2EE Web applications must have an authen-method deployment descriptor configured to CLIENT-CERT. This configuration uses the Identity Assertion security provider, where authentication is performed based on an external token (in this case, the ObSSOCookie). See "Authentication for Web-Only Resources" on page 10-6 for details.

  WebGate protects only Web resources (HTTP). When a user attempts access to a Web resource protected by WebGate:

- WebGate queries the Access Server, which authenticates and authorizes access to the Web resource.
- WebGate sets the ObSSOCookie in the HTTP header.
- The Identity Asserter (authentication provider) reads the ObSSOCookie and sets the Subject. The Subject remains until the user session expires. Further authentication is not required to access any type of resource.

Non-Web applications must use the Authenticator service provider rather than the Identity Assertion security provider.c. When Web resource accesses include mixed resources such as JNDI internally, the Authorization Provider handles access decisions.

A WebGate is a Web server plug-in access client that intercepts HTTP requests for Web resources and forwards them to the Access Server for authentication and authorization. An AccessGate is a form of access client that processes requests for Web and non-Web resources (non-HTTP) from users or applications.

With this integration, whether you have Web resources, non-Web resources, or a mix of Web and non-Web resources the question of which component performs authentication is based on the type of deployment. For example:

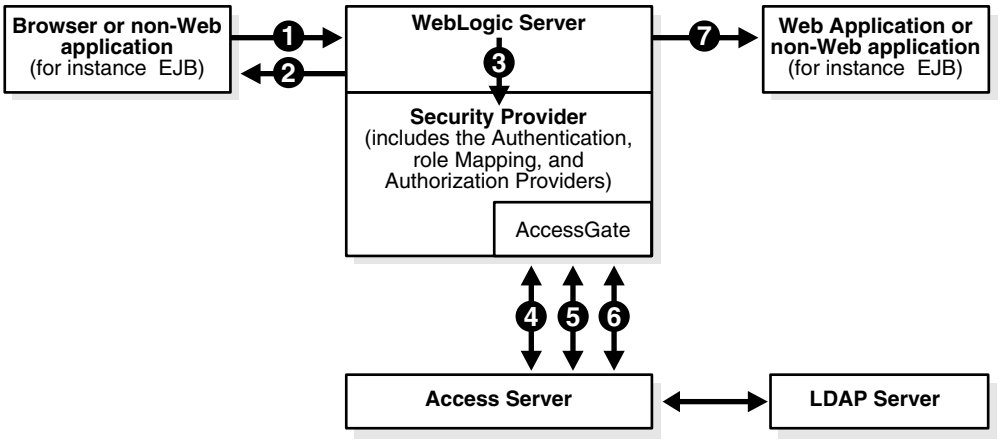
- WebGate comes into play when you have a WebLogic proxy installed and resources deployed on the WebLogic Server are accessed through it.
- For non-Web resources, only the Authenticator is involved (the Identity Asserter is not used at all).
- When you have a mix of Web and non Web resources, the resource is authenticated by either the Authenticator or the Identity Asserter based upon your deployment descriptor settings

  After the user is authenticated a session is created and the Subject is set in the request context. Subsequent access to non-Web resources will not require authentication until the user session expires. For more information on this scenario, see "Authentication for Mixed Web and Non-Web Resources" next.

Authentication for Mixed Web and Non-Web Resources

In this scenario, WebLogic resources, including both Web and non-Web applications, are protected using Oracle Access Manager authentication and authorization schemes. The J2EE Web applications have been configured with deployment descriptors for basic or form authentication. The Identity Asserter need *not* be configured. The ObSSOCookie is required to achieve the SSO functionality between Web resources. AccessGate generates the ObSSOCookie when the user is authenticated, and AccessGate handles all the communications between the Security Provider and the Access Server. AccessGate processes the ObSSOCookie and sets it in the HTTP header. The Security Provider then uses this cookie and authenticates and authorizes the user.

Figure 10–1 Mixed Web and non-Web Resources (Basic and Form Authentication)



Process overview: User authentication, mixed resource types

1. A user attempts to access a Oracle Access Manager-protected WebLogic resource.
2. The WebLogic Server challenges the user for a username and password (not Oracle Access Manager) using a predefined WebLogic login form because the application's deployment descriptor requires authentication from the container.

   You may use your own login form, which must be customized for WebLogic as described in "Adding Authentication Methods to web.xml" on page 10-45.
3. The WebLogic Server forwards the username and password to the Security Provider for authentication and authorization.
4. The Authentication Provider uses the AccessGate to communicate with the Access Server to verify the user's identity.
5. If authentication is successful, the Role Mapping Provider uses the AccessGate to communicate with the Access Server to determine what Oracle Access Manager-defined roles are assigned to this user. These roles are mapped to security roles in WebLogic. In Oracle Access Manager, these roles are configured as a return action when getting an authorization policy for /Authen/Roles. The return actions can be configured in two ways:
   - **Static:** By entering constant values for name and return value.
   - **Dynamic:** By configuring a user profile attribute as the return value.
6. The Authorization Provider uses the AccessGate to ask the Access Server to verify that the user has permission to access the requested resource.

   The policies that protect resources are specified in the Policy Manager application in Oracle Access Manager. Policies that are defined in web.xml are not be honored when the Oracle Access Manager's NetPoint Authorization Provider is in effect. This provider supports retrieving external data for authorization. See "Authorization Data from an External Source" on page 10-50 for details.
7. If authorization is successful, the WebLogic Server enables the user to access the requested resource. The ObSSOCookie is set so that when the user attempts to access additional Oracle Access Manager-protected non-WebLogic resources, re authentication is not performed.

In this scenario, if the ObSSOCookie is already set and the user has logged in using form-based authentication, the user is logged in without being challenged. You can

configure this type of integration by providing additional logic as illustrated in the sample file WebLogic login.jsp. You provide the additional logic in a file called NetPointSSO.jsp, which is provided in the installation directory.

> **Note:** When any WebLogic resource is requested, control goes to WebLogic Server and the login page of the application is launched. The Authenticator comes into play only after the login page is submitted. You need to modify JSP only if the ObSSOCookie is set already.

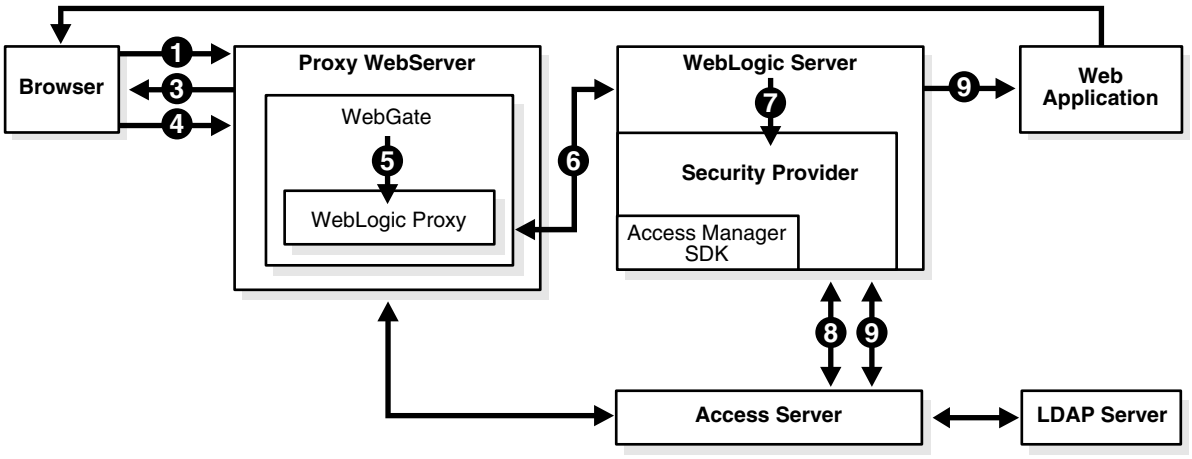
For details about the ObSSOCookie, see the Oracle Access Manager Access System Administration Guide.

Authentication for Web-Only Resources

In this scenario, all types of authentication schemes supported by Oracle Access Manager can be used, including those that require identity assertion (also called perimeter authentication), using the ObSSOCookie as the basis of the authentication. A proxy server running WebGate is installed to protect the WebLogic Server. The WebGate performs all of the authentications and authorizations. Identity assertion is used for authenticating Web applications in WebLogic.

This scenario only supports Web applications.

Figure 10–2 Security Provider Scenario for Client Cert Authentication



Process overview: User authentication, Web-only applications

1. A user attempts to access an Oracle Access Manager-protected Web application that is deployed on the WebLogic server.

   The application has an authen-method deployment descriptor configured to CLIENT-CERT.

2. WebGate intercepts the request and queries the Access Server to check if the resource is protected.

3. If the resource is protected, WebGate challenges the user for credentials based on the type of Oracle Access Manager authentication scheme configured for the resource.

4. The user presents credentials such as user name and password, or a certificate.
5. If the user authenticates successfully, WebGate generates an ObSSOCookie, appends this in an HTTP header; the Web server forwards this HTTP request to the WebLogic proxy plug-in which forwards the request to the WebLogic server.

   In this case, the HTTP response object is set in the header, not in the ObSSOCookie.
6. The WebLogic proxy plug-in passes the cookie in the HTTP header to the WebLogic Server.
7. The WebLogic Server's security service invokes the Identity Assertion Provider.

   The Identity Assertion Provider expects the ObSSOCookie as an external token for validating the user. The Asserter sets the cookie in the HTTP response object once it validates the token.
8. The Identity Assertion Provider extracts the ObSSOCookie information from the HTTP header, validates the ObSSOCookie, and retrieves the user identity from the Access Server using a return action defined in a special authentication scheme in Oracle Access Manager. A resource with the URL Prefix /Authen/Basic is protected by Oracle Access Manager and is used by the Security Provider SSPI connector internally to authenticate users.
9. The remaining steps are the same as for "Process overview: User authentication, mixed resource types" on page 10-5, step 5 - step 7.

Authentication for the Portal

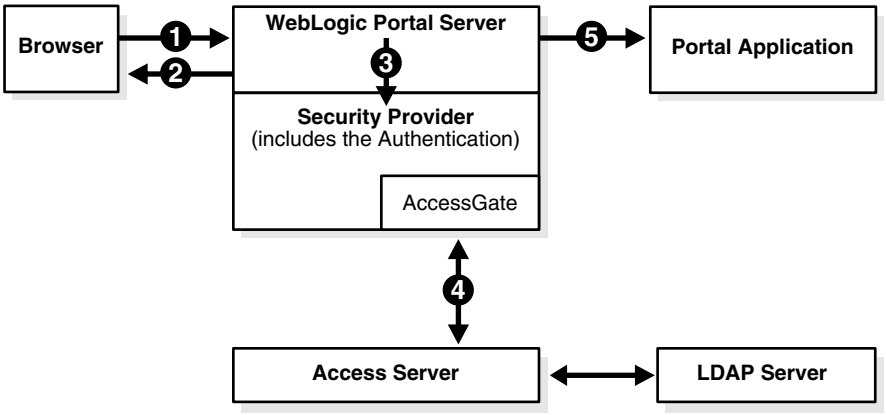
Oracle Access Manager 10*g* (10.1.4.0.1) supports integration with the WebLogic Portal.

The following scenario assumes that the environment configuration does not use a proxy Server running a WebGate. In this scenario, the resource is not required to be a Oracle Access Manager-protected resource because the Security Provider does not participate in authorizing access to Weblogic Portal resources.

The Security Provider internally authenticates the resource against the configured resource /Authen/Basic. The Portal application needs to be an application protected by WebLogic with username and password authentication.

As mentioned earlier, when a request is made for any of the resources deployed on WebLogic Server, control goes to WebLogic Server which in turn launches the login page for the application. In the case of the Weblogic Portal, Oracle does not implement the Authorization Provider and therefore does not have access to the HTTP response object. As a result, Oracle Access Manager is not be able to set the ObSSOCookie once a user is authenticated. Instead the Security Provider uses its own login filter and you need to modify the JSPs. For more information, see "Process overview: User authentication for the Portal" on page 10-8.

Figure 10–3 User authentication for the Portal



Process overview: User authentication for the Portal

1. A user attempts to access a WebLogic resource that is protected by Oracle Access Manager.
2. The WebLogic Server challenges the user for username and password.
3. The WebLogic Server forwards the username and password to the Security Provider for authentication.
4. The Authentication Provider uses the AccessGate to communicate with the Access Server to verify the user's identity.

   If authentication is successful, the Authentication Provider sets the subject correctly and passes control to WebLogic for Role Mapping and Authorization.
5. WebLogic displays the Portal application, on the basis of the authorization granted to the various portlets, etc. in the Portal Application.

Single sign-on: Single sign-on between Oracle Access Manager-protected non-WebLogic resources to WebLogic Resources and vice-versa can be achieved for Portal Web Applications that are authenticated using a login portlet.

To achieve this, additional logic must be added in form of NetpointPortalSSO.jsp to the login jsp used by the login portlet, and the POST action of the login form must be configured to invoke the Oracle Access Manager login filter class. For single sign-on setup details are given in section, see "Configuring Single Sign-On for the WebLogic Portal" on page 10-44.

Process overview: Single sign-on between Oracle Access Manager-protected non-WebLogic resources to WebLogic resources

1. A user accesses Oracle Access Manager-protected non-WebLogic resources and the ObSSOCookie is set.
2. A user accesses a WebLogic resource.
3. The NetpointPortalSSO.jsp, which is included as a part of login jsp, intercepts the ObSSOCookie and authenticates using the ObSSOCookie.

   In this case login form present in login jsp is not displayed.
4. WebLogic authorizes the resources in the Portal Application.

Process overview: Single sign-on between WebLogic resources to Oracle Access Manager-protected non-WebLogic resources

1. A user tries to access a WebLogic resource.
2. The NetpointPortalSSO.jsp, which is included as a part of login jsp, checks for the ObSSOCookie.

   In this case, no cookie is set and the login form present in login jsp is displayed. Authentication occurs using the user login credentials supplied in login.jsp.
3. The user enters their credentials in the login form; the credentials get posted to a Login Filter (ObLoginFilter configured in web.xml).
4. The Login Filter (ObLoginFilter) authenticates the user with Access Manager SDK; if authentication is successful, the Login Filter sets the ObSSOCookie and redirects to the main resource.

   Authentication is not done using WebLogic authentication.
5. The control reaches to login jsp.

   With the ObSSOCookie set, the flow mentioned in "Process overview: Single sign-on between Oracle Access Manager-protected non-WebLogic resources to WebLogic resources" on page 10-8 is followed and WebLogic Portal is accessed.
6. The user accesses Oracle Access Manager-protected non-WebLogic resources with the ObSSOCookie set and no re-authentication performed.

Supported Versions and Platforms

Any references to specific versions and platforms in this chapter are made for demonstration purposes.

You can find support and certification information at the following URL:

`http://www.oracle.com/technology/documentation/`

You must register with OTN to view this information.

Also, you can see the supported versions and platforms for this integration on Metalink, as follows.

To view information on Metalink

1. In your browser, enter the following URL:

   `https://metalink.oracle.com`
2. Log in to MetaLink.
3. Click the **Certify** tab.
4. Click **View Certifications by Product**.
5. Select the **Application Server** option and click **Submit**.
6. Choose **Oracle Identity Management** and click **Submit**.
7. Click **Oracle Identity Management Certification Information 10g (10.1.4.0.1) (html)** to display the Oracle Identity Management page.
8. Click the link for **Section 6, Oracle Access Manager Certification** to display the certification matrix.

Online Assistance

Information about installing and configuring the components required for integration of Oracle Access Manager and WebLogic is provided in the following sections of this chapter and in a readme file. For access to the readme, go to:

Security_Provider_install_dir

where *Security_Provider_install_dir* is the directory where the where the Security Provider for WebLogic SSPI is installed.

Installing and Configuring the Security Provider

The following sections provide the information needed to install and configure the Security Provider.

> **Tip:** For information on upgrading this integration, see the *Oracle Access Manager Upgrade Guide*.

Task overview: Installing and configuring the Security Provider

1. Before you install Security Provider for WebLogic, you must complete various tasks.

   See "Preparing the Environment" on page 10-11 for details.

2. Perform the installation procedure, depending on the platform where you are installing Oracle Access Manager.

   You perform a typical installation to supply default values for the Security Provider configuration. You perform advanced installation to override the default configuration.

   See "Installing the Security Provider" on page 10-11, "Completing a Typical Installation" on page 10-12, and "Completing Advanced Installation" on page 10-13 for details.

3. After installing Oracle Access Manager and WebLogic, you define Oracle Access Manager policy domains that provide a method for protecting WebLogic applications.

   See "Setting Up WebLogic Policies in Oracle Access Manager" on page 10-15 for details.

4. After you have created your resource types and authentication schemes, you can run the NetPoint Policy Deployer, or you can configure the policies manually in Oracle Access Manager.

   See "Running the NetPoint Policy Deployer" on page 10-19 or "Manually Configuring WebLogic Policies in Oracle Access Manager" on page 10-22 for details.

5. Configure the NetPointResourceMap.conf file to create mappings of WebLogic resources to Oracle Access Manager resources.

   See "Mapping WebLogic Resources to Oracle Access Manager Resources" on page 10-28 for details.

6. Configure the WebLogic environment so that the Security Provider is recognized by the WebLogic Server.

   See "Preparing the WebLogic Environment" on page 10-31 for details.

7. Configure the Identity Server.

   See "Configuring the Identity Server" on page 10-36 for details.

8. Configure multiple WebPass instances for failover purposes.

   See "Configuring Multiple WebPass Instances" on page 10-37 for details.

Preparing the Environment

Before you install Security Provider for WebLogic, you must complete the following tasks:

Task overview: Before installing the Security Provider for WebLogic

1. Install and set up the WebLogic Portal and WebLogic Server as described in your vendor documentation.

   Note: Oracle Access Manager supports integration with the WebLogic Portal.

2. Be sure you are using JDK 1.4.

3. Install and set up Oracle Access Manager, as described in the *Oracle Access Manager Installation Guide*, then:

   In the Access System Console, add an AccessGate and associate it with the Access Server that you installed.

   Add and associate an AccessGate with an Access Server for the Security Provider. You may want to name the AccessGate accordingly; for example, WebLogicProvider.

   Note: If you are going to use the deployment tool discussed in "Running the NetPoint Policy Deployer" on page 10-19, you must turn on the Access Management Service for the AccessGate, as well as all Access Servers associated with the AccessGate.

4. Create a user in the Identity System who is a WebLogic administrator and give this person delegated administrative rights. See the chapter on policy domains in the *Oracle Access Manager Access System Administration Guide* for details.

5. Install Security Provider as described in "Installing the Security Provider" on page 10-11.

Installing the Security Provider

The installation procedure depends on the platform on which you are installing Oracle Access Manager. The following example occurs on a Windows system. However, installation is the same after you launch the installation package for your platform.

To install the Security Provider for WebLogic

1. Locate and launch the Security Provider installation package. For example:

   Oracle_Access_Manager10_1_4_0_1_Win32_BEA_WL_SSPI

   The install wizard launches and the Welcome screen appears.

If the AccessGate fails during installation, you can run the tool configureAccessGate after installation, which is located in:

Security_Provider_install_dir\oblix\tools\configureaccessgate

where *Security_Provider_install_dir* is the directory where the Security Provider for WebLogic SSPI is installed. See the *Oracle Access Manager Access System Administration Guide* for information on AccessGates and the configureAccessGate tool.

2. On the Welcome screen, click Next.
3. Confirm that you are logged in as a user with administrative rights, then click Next.
4. Select an installation directory and click Next.
5. View the confirmation screen and click Next.

   A set of files are installed. When the installer has completed, you are prompted as to whether you want a Typical or Advanced installation.
6. Select Typical or Advanced and click Next.
7. Continue with the procedure that is appropriate for your environment:
   - "Completing a Typical Installation" on page 10-12.
   - "Completing Advanced Installation" on page 10-13.

Completing a Typical Installation

Typical installations supply default values for the Security Provider configuration.

A typical installation prompts you for a transport security mode. The transport security mode that you select for the Security Provider must match the transport security mode for the Access Server. Information on the prompts for installing in simple and certificate mode are available in the *Oracle Access Manager Installation Guide*. The prompts for configuring the transport security mode for the Security Provider are similar to those presented when installing any other Oracle Access Manager component.

To finish a typical installation

1. Complete the WebPass details, as follows:
   a. Enter the hostname where webpass is installed.
   b. Enter the webpass port number.
   c. Indicate whether the webpass is protected by a webgate.

   You complete step 2 when the WebPass is protected by a WebGate. Otherwise, proceed to step 3. If you have chosen to use WebGate to protect WebPass, the assumption is that you are protecting the Oracle Access Manager applications with policy domains. Therefore, it is also assumed that single sign-on between these components has been configured correctly.
2. **WebPass Protected by WebGate:** Complete the following steps.
   a. Enter the cookie domain for the WebGate (for example, .domain.com). The ObSSOCookie is then recognized by all servers within this domain.
   b. Enter the cookie path (/).
3. Complete directory-specific information, as follows:

a. Specify whether WebPass requires an HTTPS connection.

   This is the SSL for secure connection when WebPass runs on HTTPS.

b. Specify the user attribute.

   This attribute must be the same as the attribute configured for the Login semantic type in the Identity Server or a unique attribute in the user's profile such as uid.

c. Specify the user search attribute.

   This attribute must be the same as the attribute configured for the DN Prefix semantic type for the person object class in the Identity System. The person object class type must a structural object class. The administrator of your directory server sets this search attribute. The user attribute and the user search attribute cannot be the same.

d. Specify the group search attribute.

   This attribute must be the same as the attribute configured for the DN Prefix semantic type for the group object class in the Identity System. The group object class a structural object class. The administrator of your directory server sets the group search attribute.

4. Select a transport security mode:

   - **Open:** If you select open mode, all data is in plain text.
   - **Simple:** If you select simple mode, you are prompted to supply a global pass phrase. As in Cert mode, you secure the private key with a Privacy Enhanced Mail (PEM) pass phrase. Before an AccessGate or Access Server can use a private key, it must have the correct PEM pass phrase. The PEM pass phrase is stored in a encrypted file called password.lst. For Simple mode, the PEM pass phrase is the same for each WebGate and Access Server instance.
   - **Cert:** If you select cert mode, you are prompted to supply a global pass phrase. You are then asked if you wish to request a certificate or install a certificate.

5. Supply information regarding the AccessGate and Access Server that you have installed.

6. Review the readme that appears.

   The information in this readme is covered in this chapter also.

7. Confirm the installation.

8. Continue with "Setting Up WebLogic Policies in Oracle Access Manager" on page 10-15.

Completing Advanced Installation

Advanced installation permits you to override the default configuration. All the configuration options that you can set in an advanced installation are provided in the sample configuration file described in "NetPointProvidersConfig.properties" on page 10-54. This gives you the opportunity to customize your installation, which can be useful if you have configured several versions of the Security Provider authentication and authorization schemes.

Note: Do not attempt an advanced installation unless you are familiar with creating policy domains and policies in Oracle Access Manager and have run though at least one typical installation of the Security Provider.

To finish an advanced installation

1. Complete the screen subtitled "Oracle Security Provider" to use a special policy to authenticate users in WebLogic. Specify the following:

   - **Resource Type:** This is the name of a resource type used by the Security Provider to authenticate users.

     See "To configure the WebLogic resource types" on page 10-16 for details.

   - **Resource Name:** This is the URL prefix for the resource used by the Security Provider to authenticate users.

     See "To add resources to the domain in Oracle Access Manager" on page 10-23 for details.

   - **Resource Name used for Anonymous Access:** This is the URL prefix for the resource used when allowing anonymous access to certain resources.

     See "To add resources to the domain in Oracle Access Manager" on page 10-23 for details.

   - **Resource operation:** This is the operation specified on the resource type definition.

     The operation is performed to authenticate users.

   - **Login Parameter for credential_mapping Plug-in of Authentication Scheme:**

     See "To create authentication schemes for WebLogic" on page 10-17 for details.

   - **Password Parameter User for password_validation Plug-in of Authentication Scheme:** See "To create authentication schemes for WebLogic" on page 10-17 for details.

   - **Action Type:** Action is configured to get the login ID from the ObSSOCookie. This is the action configured on the authorization rule.

     See "To add authorization and authentication rules to the domain" on page 10-24 for details.

   - **Action Name:** Action is configured to get the login ID from the ObSSOCookie.

     See "To add authorization and authentication rules to the domain" on page 10-24 for details.

   - **Dummy Username:** For Form Login with SSO when there is No WebGate on Proxy HTTP Server.

     This is used if you are protecting both Web and non-Web resources and you are using form login. If the login.jsp is modified to include NetPointSSO.jsp, and the user has already logged in to an Oracle Access Manager-protected resource, the user has received a token. The next time the user tries to access a protected resource, Oracle Access Manager uses the dummy user name as the user name and the token is used as the password. The default is obdummyuser. Oracle recommends that you use the default name.

- **WebLogic resource types used for web applications (comma separated)**: These are the resource types that WebLogic uses for Web applications.

  See "Mapping WebLogic Resources to Oracle Access Manager Resources" on page 10-28 for details.

2. Complete the screen subtitled "Oracle Security Provider" to use a special policy to get roles for a user.

   Specify the following configuration to set up this policy:

   - **TTL (time to live) of elements in roles cache:** This is the amount of time the action is preserved in the cache.
   - **Time to delete expired elements of cache (in seconds):** This is the time interval for freeing the memory used for expired elements in the cache.
   - **Resource type:** This is the name of a resource type used by the Security Provider to get roles. See "To configure the WebLogic resource types" on page 10-16 for details.
   - **Resource name:** This is the URL prefix for the resource used by the Security Provider to get roles. See "To add resources to the domain in Oracle Access Manager" on page 10-23 for details.
   - **Resource operation**: This is the operation specified on the resource type definition. The operation is performed to authenticate users.
   - **Action Type in authorization rule to get roles:** This is the action configured on the authorization rule to get user roles. See "To add authorization and authentication rules to the domain" on page 10-24 for details.

3. Complete the screen, "Configuration for Oracle Security Provider for WebLogic":

   - **Default access to resources not protected by Oracle Access Manager (deny, allow, abstain):** Allow grants access, deny forbids it. Abstain means that if there are multiple security providers, WebLogic goes to the next security provider to decide what to do.
   - **Map the authorization result ABSTAIN to (allow, deny):** A result of abstain can be automatically reset to allow or deny.
   - **Set debugging**: Debug logs are written to the WebLogic log file.

4. Finish the installation by completing the steps described in "Completing a Typical Installation" on page 10-12.

Setting Up WebLogic Policies in Oracle Access Manager

After installing Oracle Access Manager and WebLogic, you need to define Oracle Access Manager policy domains that provide a method for protecting WebLogic applications. The basics of defining a policy domain are:

- **Creating resource types:** This enables Oracle Access Manager to identify the kinds of WebLogic resources that it should protect and the operations (such as GET) associated with the resource.
- **Creating authentication schemes:** This enables Oracle Access Manager to verify user identities.
- **Creating authorization schemes:** This enables Oracle Access Manager to grant users access to the resources that you have defined.

- **Creating a policy domain**: This creates a container for your WebLogic-related policies.
- **Creating policies:** These are directives for protecting specific WebLogic resources. Policies are an amalgam of resource type definitions, URLs identifying the resource locations, and the authentication and authorization schemes to apply when users access the resources.

The first step in setting up a policy domain is to define your resources and authentication schemes. These tasks are discussed in the following sections, which assume a basic knowledge of Oracle Access Manager. The *Oracle Access Manager Access System Administration Guide* provides details on tasks described in the following sections.

Note: The resource type wl_svr described in this document is available if you want to protect access to starting and stopping the WebLogic server. To do this, you can define a policy (as described in "To create policies for the domain" on page 10-26) that uses this resource type. You can find the information you need for this policy in the isAccessAllowed entries in the debug logs that contain the string <svr>. Note that you must create this policy manually. The deployment tool provided with the Oracle Access Manager Security Provider does not create this policy for you. Also, the resource type wl_ejb is used when you deploy EJB applications. You can create policies that use this type of resource manually, or you can use the deployer tool to create these policies.

To configure the WebLogic resource types

1. From the Access System Console, click the Access System Configuration tab, then click Common Information Configuration, Resource Type Definitions.

   The List All Resource Types page appears.

2. From the List All Resource Types page, click Add.

   The Define a new Resource Type page appears.

3. Define and save the first resource type:

   Name: wl_url

   Display name: wl_url

   Resource matching: case insensitive

   Resource operation: GET

   Resource operation (second): POST

4. Define and save the second resource type:

   Name: wl_svr

   Display name: wl_svr

   Resource matching: case insensitive

   Resource operation: BOOT

   Resource operation (second): DEFAULT

5. Define and save the third resource type:

Third resource:

Name: wl_adm

Display name: wl_adm

Resource matching: case insensitive

Resource operation: DEFAULT

6. Define and save the fourth resource type:

   Name: wl_ejb

   Display name: wl_ejb

   Resource matching: case insensitive

   Resource operation: EXECUTE

7. Define and save the fifth resource type:

   Name: wl_authen

   Display name: wl_authen

   Resource matching: case insensitive

   Resource operation: LOGIN

8. Define and save the sixth resource type:

   Name: http

   Display name: http

   Resource matching: case insensitive

   Resource operation: GET

   Resource operation: POST

   Resource operation: PUT

   Resource operation: HEAD

   Resource operation: DELETE

   Resource operation: TRACE

   Resource operation: OPTIONS

   Resource operation: CONNECT

   Resource operation: OTHER

To create authentication schemes for WebLogic

1. From the Access System Console, click Access System Configuration, Authentication Management, Add.
2. Create the first authentication scheme to be used by the Security Provider to authenticate users, as follows (which uses the Oracle Access and Identity authentication scheme as a template).
   a. Configure the General tab:

      Name: Oracle WebLogic Access and Identity authentication scheme

      Description: Used to authenticate users who access WebLogic resources.

      Level: 1

Challenge Method: Basic

Challenge Parameter: realm:Oracle Access and Identity

SSL Required: No

Challenge Redirect: (Leave blank)

Enabled: (Leave as is)

Note: The realm: string on the challenge parameter is required. The text after this parameter can be configured. Also, in the Name field, the "l" in WebLogic must be lowercase to match the name in a NetPointWeblogicTools.properties file that is part of the integration solution. In general, the name of this authentication scheme should be identical (with case sensitivity) to the ObWLAuthenticationScheme.Name parameter in the NetPointWeblogicTools.properties file.

---

b. Save the information on the General tab by clicking Save.

c. Click the Plugins tab, and use the credential_mapping and validate_password plugins from the existing Oracle Access Manager authentication schemes. In the credential_mapping plug-in, be sure the mapping base and mapping filter use objects that are specific to your environment. Examples:

```
credential_mapping
   obMappingBase="o=company,c=us",obMappingFilter=
   "(&(&(objectclass=inetorgperson)(uid=%userid%))
   (|(!(obuseraccountcontrol=*))
   (obuseraccountcontrol=ACTIVATED)))"
 validate_password obCredentialPassword="password"
```

where place-holders such as o=company,c=us, and inetorgperson are replaced with values that are valid for your organization.

After you create at least one plug-in, default steps and a default authentication flow are created automatically.

d. After creating a plug-in, you can enable the authentication scheme by clicking the General tab, Modify, selecting the Enable option, and clicking Save.

3. Create the second authentication scheme for un-protecting certain resources, such as gif images in WebLogic resources using the following details and the Anonymous authentication scheme as a template:

General tab:

Name: Oracle WebLogic Anonymous Authentication

Description: Used to un-protect gifs, and so on.

Level: 0

Challenge Method: Anonymous

Parameter: (Leave blank)

SSL Required: No

Challenge Redirect: (Leave blank)

Enabled: Yes

Note: In the Name field, the "l" in WebLogic must be lowercase to match the name in a NetPointWeblogicTools.properties file that is part of the integration solution. In general, the name of this authentication scheme should be identical (including case sensitivity) to the ObWLNoneAuthenticationScheme.Name parameter in the NetPointWeblogicTools.properties file.

Plugins tab:

Use the credential_mapping plug-in from the pre-configured Anonymous authentication schemes. In the credential_mapping plug-in, be sure the mapping base and mapping filter use objects that are specific to your environment. Use OblixAnonymous as the mapping filter. Example:

```
credential_mapping obMappingBase="o=company,c=us",obMappingFilter="
(uid=OblixAnonymous)"
```

where place-holders such as o=company,c=us, and uid should be replaced with values appropriate for your environment.

4. Restart the Access Server.

   You are now ready to create the following policies:

   - **Basic Authentication Policy**: This policy is used internally to authenticate users by evaluating the user name and login. The policy protects resources with a URL prefix of /Authen/Basic.
   - **Role-based Authentication Policy:** This policy gets user roles. This policy protects resources with a URL prefix of /Authen/Role.
   - **Anonymous Authentication Policy:** This policy provides anonymous access to gifs, and other resources. This policy protects resources with a URL prefix of /Authen/Anonymous.
   - **Anonymous Authentication Policy (second):** This policy enables anonymous access for users. This policy protects resources with a URL prefix of /Authen/Anonymous.
   - **WebLogic Administrator Policy:** This policy enables access to the WebLogic administration console.

   The Oracle-provided Policy Deployer (known as the NetPoint Policy Deployer) can automate this process, or you can create these policies manually.

5. Continue with one of the following discussions:

   - "Running the NetPoint Policy Deployer" on page 10-19
   - "Manually Configuring WebLogic Policies in Oracle Access Manager" on page 10-22

Running the NetPoint Policy Deployer

After you have created your resource types and authentication schemes, you can run the NetPoint Policy Deployer. This tool enables you to:

- Create the policy domain and policies during initial setup of the Security Provider for WebLogic.

This policy domain uses the resource type wl_authen created in "To configure the WebLogic resource types" on page 10-16.

- Create and delete policy domains and policies that protect WebLogic applications.

You need to run this tool at least once for initial setup. Afterwards, you can either manually create policies for applications deployed in WebLogic, or you can run this tool to automatically create them. Refer to the following procedures:

- To prepare for running the Policy Deployer Tool
- To run the Policy Deployer Tool for the first time
- To run the Policy Deployer after the first time

Note: If you do not want to use the Policy Deployer tool, you must manually configure WebLogic policies as described in "Manually Configuring WebLogic Policies in Oracle Access Manager" on page 10-22.

To prepare for running the Policy Deployer Tool

1. Add the following to CLASSPATH:
   - *Security_Provider_install_dir*/oblix/tools/npWLTools
   - *Security_Provider_install_dir*/oblix/tools/npWLTools/npWLTools.jar
   - *Security_Provider_install_dir*/oblix/lib/jobaccess.jar
2. Add the following:

   Security_Provider_install_dir/oblix/lib

   On Windows, you add this to the PATH. On Solaris, you add this to LD_LIBRARY_PATH. On HP-UX, you add this to SHLIB_PATH.
3. Ensure that the following configuration files are copied from *Security_Provider_install_dir* to the WebLogic domain directory:
   - NetPointProvidersConfig.properties

     See "NetPointProvidersConfig.properties" on page 10-54 for details.
   - NetPointResourceMap.conf

     See "Mapping WebLogic Resources to Oracle Access Manager Resources" on page 10-28 for details on configuring this file.

To run the Policy Deployer Tool for the first time

1. Review the following configuration file:

   Security_Provider_install_dir/oblix/tools/npWLTools/NetPointWeblogicTools.properties

   Where *Security_Provider_install_dir* is the directory where the Security Provider for WebLogic SSPI is installed.

   Be sure that ObWLTools.setupInitialNetPointSSPIPolicies is set to true (the default).
2. If you are running the WebLogic Web applications in Identity Assertion mode, configure the following parameters in the NetPointWeblogicTools.properties configuration file:

- ObWLWebResource.usingIdentityAssertion
- ObWLWebResource.proxyPrefix

See "NetPointWeblogicTools.properties" on page 10-60 for details.

3. From the command line, enter the following:

   ```
   java com.oblix.weblogic.tools.NetPointPolicyDeployer userid
   password
   ```

   where *userid* and *password* belong to the login ID of the Master Oracle Access Manager Administrator. For all the policies that this tool creates, it initially grants access to this userID only. Use JDK 1.4 to ensure that this command works as expected.

4. Go to the Policy Manager and check if the policies are created.

   See "To create a policy domain in Oracle Access Manager" on page 10-22 for details on the policy domains and how they should be configured.

5. After running the tool, go to the Access System Console and provide the proper access to the policies that have been created.

   Note: Setting up security policies in Oracle Access Manager is only required for protection of resources deployed on the BEA WebLogic Server. The Security Provider only supports authentication for portals. If you are using the Security Provider in the portal domain, then only the policies required for authentication need to be created. These are created when you run the Policy Deployer for the first time with the ObWLTools.SetupIntialNetPointSSPIPolicies parameter set to true and the ObWLTools.DeployPolicy and ObWLTools.UnDeployPolicy parameters set to false in the NetPointWeblogicTools.properties file. Therefore, for the portal domain, you can ignore the procedure "To run the Policy Deployer Tool for the first time" on page 10-20.

6. Ensure that the policy domain that was just created is enabled in Oracle Access Manager.

7. In the Policy Manager, click My Policy Domains, then click the WebLogic policy domain.

   The status of the policy domain should be enabled.

8. If the status of the policy domain is not enabled, click Modify and enable it.

To run the Policy Deployer after the first time

1. Open the following configuration file:

   Security_Provider_install_dir/oblix/tools/npWLTools/

   NetPointWeblogicTools.properties

   where *Security_Provider_install_dir* is the directory where the Security Provider for WebLogic SSPI is installed.

2. Be sure that NetPointDeployPolicy.txt and NetPointUndeployPolicy.txt are in the WebLogic domain directory.

   NetPointDeployPolicy.txt is created when applications are deployed from the WebLogic server. NetPointUndeployPolicy.txt is created when applications are undeployed from the WebLogic server. The Security Provider writes security

policy data into this file when J2EE applications are deployed or undeployed. The Policy Deployer reads this file to create policies.

3. Set the following:
   - ObWLTools.SetupIntialNetPointSSPIPolicies=false
   - ObWLTools.DeployPolicy=true
   - ObWLTools.UnDeployPolicy=true

   Note: If the tool completes successfully but there are no policies in the Oracle Access Manager Policy Manager, look in the NetPointDeployPolicy.txt file and the NetPointUnDeployPolicy.txt files to see if the following settings exist for the same resource: ObWLTool.DeployPolicy=true, ObWLTool.UnDeployPolicy=true

   This can cause the policies for the same resource to be created and deleted from the Oracle Access Manager Policy Manager.

   If this occurs, set ObWLTool.DeployPolicy=true and ObWLTool.UnDeployPolicy=false while creating policies using the NetPointPolicyDeployer tool.

4. Follow step 2 - step 5 of "To run the Policy Deployer Tool for the first time" on page 10-20.
5. Continue with "Manually Configuring WebLogic Policies in Oracle Access Manager" on page 10-22.

Manually Configuring WebLogic Policies in Oracle Access Manager

If you do not wish to use the Policy Deployer tool described in "Running the NetPoint Policy Deployer" on page 10-19, you can configure the policies manually in Oracle Access Manager.

Task overview: Manually configuring WebLogic Policies in Oracle Access Manager

1. Create a WebLogic policy domain, as described in "To create a policy domain in Oracle Access Manager" on page 10-22.
2. Add resources to the domain, as described in "To add resources to the domain in Oracle Access Manager" on page 10-23.

   Multiple resources can be defined for each resource type configured in "To configure the WebLogic resource types" on page 10-16. Resources provide URL prefixes under which various policies can be defined.
3. Add authorization and authentication rules to the domain, as described in "To add authorization and authentication rules to the domain" on page 10-24.
4. Define policies within the domain, as described in "To create policies for the domain" on page 10-26.

   The URL prefix in the resource and the URL pattern in the policy together form the definition of the data to be protected by the policy.

To create a policy domain in Oracle Access Manager

1. Log in to the Access System.

2. From the Policy Manager, click Create Policy Domain.
3. Define the following policy domain:

   Name: SecuProvForWebLogic

   Description: Policy domain for WebLogic resources.
4. Save this policy domain.

To add resources to the domain in Oracle Access Manager

1. From the Policy Manager, click My Policy domains and select the new policy domain.
2. Click the Resources tab for the policy domain.
3. Click add, then configure, and save the resource for user authentication:

   Resource type: wl_authen

   URL prefix: /Authen/Basic

   Description: Resource used by the Security Provider to authenticate users.

   The resource type was defined in "To configure the WebLogic resource types" on page 10-16.
4. Click add, then configure, and save the resource for anonymous access:

   Resource type: wl_authen

   URL prefix: /Authen/Anonymous

   Description: Resource used for anonymous authentication, where a session is created for anonymous users.
5. Click add, then configure, and save the resource for returning user roles:

   Resource type: wl_authen

   URL prefix: /Authen/Roles

   Description: Resource used when the policy is configured to return roles that are mapped to security roles in WebLogic.
6. Click add, then configure, and save the resource for protecting the WebLogic administration console:

   Resource type: wl_url

   URL prefix: /console

   Description: Optional. Protects the WebLogic administration console.
7. Click add, and save the resource for server administration:

   Resource type: wl_svr

   URLprefix: /servername

   Description: Optional. Restricts access for users who perform server administration tasks, such as starting and stopping the server.

Note: If you get an error, be sure that these resources are not already used by another policy domain.

To add authorization and authentication rules to the domain

1. In the Policy Manager, add the resources that you defined in "To configure the WebLogic resource types" on page 10-16 to this policy domain. For example:

   Policy Manager, My Policy domains, *policy_domain*

   Next, you add authorization rules.

2. Click the Authorization Rules tab, click Add, and create the rule for administrators:

   General tab:

   Name: Authz rule for admin

   Description: Authorization rule for an administrator. This rule provides administrator access to WebLogic applications.

   Enabled: Yes

   Actions tab: Leave blank.

   Allow Access tab:

   People: Add users who are allowed to be WebLogic administrators.

3. Click Add, and create the rule for anonymous access:

   General tab:

   Name: Authz rule for anyone

   Description: Provides anonymous access to resources

   Enabled: Yes

   Actions tab: Leave blank.

   Allow Access tab:

   Role: Anyone

4. Click Add, and create the general rule for access:

   General tab:

   Name: Authz Rule for Authen

   Description: Returns the user ID from the ObSSOCookie using the return action configured on the Actions tab.

   Enabled: Yes

   Actions tab:

   Redirect to: Leave blank On Authorization Success

   Return Type: WL_REALM

   Name: uid

   Return Attribute: login ID attribute where this is the attribute in your directory for the user login ID.

   On Authorization Failure Return: Leave blank

Allow Access tab:

Role: Anyone.

5. Click Add, and create the rule for returning the user WebLogic role:

   General tab:

   Name: Authz rule for role

   Description: Returns the user's role. These roles are hard-coded on the return actions, and they match administrative roles in WebLogic.

   Enabled: Yes

   Actions tab:

   Redirect to: Leave blank

   On Authorization Success Return:

   Type: WL_REALM

   Name: role1

   Return Value: Admin

   Type: WL_REALM

   Name: role2

   Return Value: Operator

   Type: WL_REALM

   Name: role3

   Return Value: Monitor

   Type: WL_REALM

   Name: role4

   Return Value: Deployer

   On Authorization Failure Return: Leave blank

   Allow Access tab:

   People: People who are allowed to be the WebLogic administrator.

   > **Note:** The WebLogic administration console requires the administrator to have certain roles. These are hard-coded on the return actions of Authz rule for role, defined in this step. As an alternative, you can allow access to everyone rather than just the administrator, and control access by using a return attribute such as a user profile attribute or a special attribute called obMyGroups that returns all the groups that a user belongs to.

6. Create a default authorization rule that allows anonymous access, as follows:
   a. Click the Default Rules tab, Authorization Rules, Add.
   b. Select the rule for anonymous access that you created in the previous step.

This is the rule called Authz rule for anyone.

c. Click Save.

If no policy is evaluated, the default rule provides anonymous access to everyone. This can be changed to meet the requirements of your environment.

7. Create a default rule that authenticates users for access to all resources that do not fall under a specific policy, as follows:

a. Click the Default Rules tab, Authentication Rule, Add.

The General page appears.

b. On this page, add the following default rule (or configure another one, if needed for your environment):

Name: Oracle WebLogic Anonymous Authentication.

Authentication scheme: Use the authentication scheme you created in "To create authentication schemes for WebLogic" on page 10-17.

To create policies for the domain

1. Create policies for this domain from the Policy Manager. For example:

Policy Manager, *policy_domain*, Policies tab

2. Add the basic policy for this policy domain:

General tab:

Name: Basic authentication policy

Description: Authentication using basic LDAP username and password.

Resource type: wl_authen

Resource operation: LOGIN

Resource: /Authen/Basic

Authentication Rule tab:

Name: Basic authentication rule

Authentication Scheme: Select the basic authentication scheme Oracle WebLogic Access and Identity that you created in "To create authentication schemes for WebLogic" on page 10-17.

Authorization Rule tab: Add the rule Authz Rule for Authen that you created in "To add authorization and authentication rules to the domain" on page 10-24.

Note: If you are using identity assertion as the authentication mechanism that protects Web applications, see the notes in "Preparing the WebLogic Environment" on page 10-31 after configuring your authentication policies.

---

3. Add the anonymous access policy for this policy domain:

Policies, General tab:

Name: Anonymous authentication policy

Description: Authenticates anonymous users.

Resource type: wl_authen

Resource operation: LOGIN

Resource: /Authen/Anonymous

Authentication Rule tab:

Name: Anonymous authentication rule

Authentication Scheme: Select the anonymous authentication scheme Oracle WebLogic Anonymous Authentication that you created in "To create authentication schemes for WebLogic" on page 10-17.

Authorization Rule tab: Add the rule Authz rule for anyone that you created in "To add authorization and authentication rules to the domain" on page 10-24.

4. Add the user role policy for this policy domain:

   General tab:

   Name: Role-based authentication policy

   Description: Authenticates users and gets their WebLogic roles

   Resource type: wl_authen

   Resource operation: LOGIN

   Resource: /Authen/Roles

   Authentication Rule tab:

   Name: Role authentication rule

   Authentication Scheme: Select the Oracle WebLogic Access and Identity rule that you created in See "To create authentication schemes for WebLogic" on page 10-17.

   Authorization Rule tab: Add the Authz rule for role that you created in "To add authorization and authentication rules to the domain" on page 10-24.

5. Add the "unprotect" policy for this policy domain:

   General tab:

   Name: Unprotect policy for gifs and other files

   Description: Allow anonymous access to gif files

   Resource type: wl_url

   Resource operation: GET,POST

   Resource: all (if there are no resources defined, this defaults to all)

   URL pattern: /.../*.gif

   Authentication Rule tab:

   Name: WebLogic Domain Anonymous authentication rule

Authentication Scheme: Select the Oracle WebLogic anonymous authentication rule that you created in "To create authentication schemes for WebLogic" on page 10-17.

Authorization Rule tab: Add the Authz rule for anyone that you created in "To add authorization and authentication rules to the domain" on page 10-24.

6. Add the WebLogic administration console policy for this policy domain:

   General tab:

   Name: Policy for WebLogic admin console

   Description: Allow administrator access to the WebLogic admin console

   Resource type: wl_url

   Resource operation: GET

   Resource: /console

   Authentication Rule tab:

   Name: WebLogic Domain default authentication rule

   Authentication Scheme: Select the Oracle WebLogic Access and Identity rule that you created in "To create authentication schemes for WebLogic" on page 10-17.

   Authorization Rules tab: Add the Authz rule for admin that you created in "To add authorization and authentication rules to the domain" on page 10-24.

   For additional information, see the *Oracle Access Manager Access System Administration Guide*.

7. Continue with "Mapping WebLogic Resources to Oracle Access Manager Resources" on page 10-28.

Mapping WebLogic Resources to Oracle Access Manager Resources

The NetPointResourceMap.conf file contains mappings of WebLogic resources to Oracle Access Manager resources. These mappings allow both products to recognize each other's resource definitions. Oracle Access Manager recognizes only URLs, whereas each WebLogic resource has different set of elements associated with it. By mapping these resource elements to URLs and operations, all types of resources can be protected through Oracle Access Manager.

> **Note:** The NetPointResourceMap.conf file is used only for the WebLogic Server integration, not for WebLogic Portal integration

After defining your Oracle Access Manager resources, policy domain, and so on, you need to be sure that the WebLogic resources that you want to protect will correspond to the resources that you defined in Oracle Access Manager.

To map WebLogic resources to Oracle Access Manager resources

1. Locate the NetPointResourceMap.conf file. in the directory where the Security Provider is installed.

2. Review the format of this file in "NetPointResourceMap.conf File Format" on page 10-29.
3. Edit the file using information on Oracle Access Manager resource type definitions in "Setting Up WebLogic Policies in Oracle Access Manager" on page 10-15.
4. Continue with "Preparing the WebLogic Environment" on page 10-31.

NetPointResourceMap.conf File Format

Resources that have mapping entries in this file are the only ones protected. Resources that do not have a mapping entry in this file are allowed access by default.

The format of entries in the NetPointResourceMap.conf file is as follows:

Weblogic Resource Type:Oracle Access Manager Resource Type:
enabled | disabled:URL prefix:URL pattern:Operation

where:

- Weblogic Resource Type: The WebLogic resource type. For example: <url>.
- Oracle Access Manager Resource Type: The Oracle Access Manager resource type that is mapped to the WebLogic resource type. For example: wl_url.
- enabled | disabled: If enabled, all resources of the specified WebLogic resource type are protected by Oracle Access Manager. If disabled:
  - All users can access the resource if the status is disabled, allow.
  - No users can access the resource if the status is disabled, deny.
- **URL prefix:** The WebLogic elements that form a Oracle Access Manager URL prefix under which all resources are protected. This URL prefix is specified in a Oracle Access Manager policy. Each element in the URL prefix is a type of resource. For example, the following URL prefix can be used for EJBs:

  ```
  application/module/ejb
  ```

  where application and module are a specific WebLogic application and module.
- **URL pattern:** The WebLogic elements that form a more granular Oracle Access Manager URL pattern than is specified by a URL prefix. This pattern is specified in an Oracle Access Manager policy. For example, to control access to users based on a particular method, you would specify:

  ```
  methodInterface/method
  ```
- **Operation:** This maps to a WebLogic resource element such as HTTPMETHOD. If you specify a value in angle brackets ("<>"), the policy returns the matching string. If you omit the brackets, the policy returns the value associated with the parameter.

This example lists the content of the NetPointResourcemap.conf file

```
#############################################################################
# This file maps Weblogic resources to Oracle Access Manager resources.
# Oracle Access Manager only understands urls whereas each Weblogic resource has
# different set of elements associated with it. By mapping these resources to
# urls & operations all kinds of resources can be protected via
# Oracle Access Manager.
#
# The format of entries is as follows.
# Weblogic Resource Type:Oracle Resource Type: enabled/disabled:
# URL prefix:URL pattern:Operation
# For example <url> : wl_url : enabled : contextPath : uri : httpMethod
```

```
# If the resource is configured disabled, the default action can be configured.
# For example disabled,allow
#
# If the operation doesn't come from Weblogic resource field, and is fixed then
# it can be configured by putting the value between <> . For example <execute>
# If the value for operation is not specified then it defaults to "<default>"
#
# Leading & trailing white spaces in the fields and blank lines are allowed.
# Comments can be put by starting the line with #
#################################################################################
##### COMMONLY used resources #####
# HTTP resource. Available keys: application, contextPath, uri, httpMethod,
# transportType
<url>:wl_url:enabled:contextPath:uri:httpMethod

# ejb - EJB resource. Available keys: application, module, ejb, method,
# methodInterface, signature
# signature is ignored here for performance reasons.
# You can include it if you want to.
<ejb>:wl_ejb:enabled:application/module/ejb:methodInterface/method:<execute>

# Web resource. Available keys: application, uri, webResource, httpMethod,
# transportType
# This resource is deprecated by BEA in WLS 8.1. <url> replaces this resource
# type.
# You can enable it if you want to. Refer to WLS 8.1 documentation for details.
<web>::disabled,deny:uri:webResource:httpMethod

# Server resource. Available keys: server, action
# Typically server=<wls server name>
<svr>:wl_svr:enabled:server::action

# Admin resource. Available keys: category, realm, action
# Typically for admin console category=Configuration. realm is ignored in default
# configuration.
<adm>:wl_adm:enabled:category::action

##### LESS used resources #####

# JDBC resource. Available keys: resourceType, resource, action
#<jdbc>:wl_jdbc:enabled:resourceType:resource:action
<jdbc>::disabled,allow:::
# JMS resource. Available keys: destinationType, resource, action
#<jms>:wl_jms:enabled:destinationType:resource:action
<<jms>::disabled,allow:::
# JNDI resource. Available keys: path, action
#<jndi>:wl_jndi:enabled:path::action
<jndi>::disabled,allow:::
```

Note: JNDI, JDBC, JMS resource protection is disabled by default. These can be enabled in NetPointResourceMap.conf.

The tool can be used to create policies automatically only for Web and ejb resources. It cannot be used for JNDI, JDBC, JMS, and other types of resources. These need to be created by hand in the Access System Console. To find the URL pattern and operation used for the policy, set the log level to debug for the WebLogic SSPI package and look into the logs for the string "Entering OblixDatabase.isProtected for". For example:

```
Entering OblixDatabase.isProtected for resource Type=wl_jndi, isEnabled=true,
URL=/weblogic/jms/MessageDrivenBeanConnectionFactory, operation=lookup
```

For this example, a resource type wl_jndi needs to be first created in Access System Console with a resource operation of "lookup". Then policy needs to be created for the url /weblogic/jms/MessageDrivenBeanConnectionFactory (or parts of it such as /weblogic/jms) in the weblogic policy domain. For more information about protecting resources, see the *Oracle Access Manager Access System Administration Guide*.

Preparing the WebLogic Environment

The following procedure describes how to configure the WebLogic environment so that the Security Provider is recognized by the WebLogic Server.

> **Note:** When you deploy an application on Weblogic 9.2, be sure that you deploy it with the appropriate deployment descriptors for Web applications. The deployment descriptors for Web applications are web.xml and weblogic.xml. Also be sure to deploy the application with deployment descriptors for EJB applications. The files ejb-jar.xml and weblogic-ejb-jar.xml are the deployment descriptors for EJB applications.

To prepare the environment

1. Copy the mbean jar file from one of the following locations:

   From

   install_dir/oblix/lib/mbeantypes

   to

   WebLogic_Home/server/lib/mbeantypes

   > **Note:** If you are using WebLogic 9.2, copy `wl8NetPointSecurityProviders_Upgraded.jar`. If you are using WebLogic 8.1, copy `wl8NetPointSecurityProviders.jar`. If you are using WebLogic 7.0 SP2 and later, copy `wl7NetPointSecurityProviders.jar`.

2. Copy the following files from your *Security_Provider_install_dir* to your WebLogic domain folder:

   NetPointProvidersConfig.properties

   NetPointResourceMap.conf: only for the WebLogic Server domain

3. Ensure that the following Admin credentials are set in clear text in the NetPointProvidersConfig.properties file:

   OB_AdminUserName=admin

   OB_AdminUserCreds=password

   If the NetPointProvidersConfig.properties file has a clear text password, the SSPI reads in the password, encrypts it, and rewrites the properties file with the encrypted password.

Note: NetPointProvidersConfig.properties file formatting is lost when Oracle Access Manager rewrites the file with the encrypted password. You may want to save a copy of the NetPointProvidersConfig.properties file. Also, ensure that all parameters are correctly filled as mentioned in "NetPointProvidersConfig.properties" on page 10-54.

You complete the next step if the SSPI talks to a WebPass that is protected by a WebGate. Otherwise, skip to step 5.

4. **WebPass Protected by WebGate:** Complete the following activities when the Oracle Access Manager SSPI talks to a WebPass protected by a WebGate:

   a. In the NetPointProvidersConfig.properties file, ensure that OB_WebPassIsProtected is set to true. The OB_CookiePath and OB_CookieDomain parameters are configured correctly.

   b. From the Access System Console, click Access System Configuration, click AccessGate Configuration in the left navigation pane, click the link for the WebGate that protects the WebPass, and in the IPValidation field select the Off option.

      In Oracle Access Manager 10.1.4, the WebGateStatic.lst file no longer exists. The options in this file have moved to the Access System Console. See *Oracle Access Manager Access System Administration Guide* for details.

      Note: If you want to set IPValidation to True, configure the IPValidationExceptions parameter to contain the IP address.

      ---

   c. Restart the Web server.

      Note: Ensure that the security level in this authentication scheme is the same level or a lower level than the one specified in the WebLogic authentication scheme

      ---

      Next, you need to determine if the machine hosting WebPass is running SSL. If it is, complete step 5. Otherwise, skip to step 6.

5. **WebPass Host SSL-Enabled:** Determine if the machine hosting WebPass is running SSL, and if so, complete the following steps:

   a. Open the NetPointProvidersConfig.properties file and set OB_WebPassSSLEnabled = True.

   b. Obtain the CA certificate from the certificate authority to which the Web server hosting the WebPass or WebGate running in SSL mode has registered, and place it in ca.cer file.

   c. Use the keytool in JAVA_HOME\bin or JAVA_HOME\jre\bin to add the following ca certificate to cacerts keystore present in:

      ```
      JAVA_HOME\jre\lib\security folder for weblogic jdk
      keytool -import -alias ca -file ca.cer -keystore JAVA_HOME\jre\lib\
      security\cacerts
      ```

6. Add the following environment variables in the WebLogic Server startup script before the command that starts the server:

   Add the following to the CLASSPATH:

   ```
   /install_dir/oblix/lib/wlNetPoint.jar
   /install_dir/oblix/lib/bcprov-jdk14-125.jar
   /install_dir/oblix/lib/xerces.jar
   /install_dir/oblix/lib/jobaccess.jar
   ```

7. Add the following environment variables in the WebLogic Server startup script before the command that starts the server:

   Windows: Add the following to PATH:

   ```
   install_dir\oblix\lib
   ```

   Solaris and Linux: Add the following to LD_LIBRARY_PATH:

   ```
   install_dir/oblix/lib
   ```

   HP-UX: Add the following to SHLIB_PATH:

   ```
   install_dir/oblix/lib
   ```

   Portal Domain: The CLASSPATH and PATH variables should be added just after the SAVE_JAVA_OPTIONS environment variable in the startWebLogic.cmd script (On Unix, it is the startWebLogic.sh script).

8. On Linux, set the LD_ASSUME_KERNEL environment variable to 2.4.19, as follows:

   ```
   LD_ASSUME_KERNEL=2.4.19
   export LD_ASSUME_KERNEL
   ```

9. Remove the boot.properties file from the WebLogic domain directory.

   This will cause the startWebLogic script described in the next step to prompt for username and password.

10. In the WebLogic domain directory, edit the appropriate startup script:

    Windows: The script is startWeblogic.cmd

    Unix: The script is startWeblogic.sh

    Ensure the following paths are set in the script:

    ```
    /install_dir/oblix/lib/wlNetPoint.jar
    /install_dir/oblix/lib/bcprov-jdk14-125.jar
    /install_dir/oblix/lib/xerces.jar
    /install_dir/oblix/lib/jobaccess.jar
    ```

11. In the WebLogic domain directory, start the WebLogic Server using the appropriate startup script:

    Windows: This command is startWeblogic.cmd

    Unix: This command is startWeblogic.sh

    Using the WebLogic 8.1 Domain Configuration Wizard, you can create instances of a new WebLogic 8.1 domain, for example, mydomain, and a new WebLogic 8.1 server, for example, myserver. You can also create instances of a new WebLogic 8.1.3 Portal domain, for example, portalDomain, and a new WebLogic 8.1.3 portal, for example, portalServer.

12. Set up a Realm that uses Oracle Access Manager security providers, as follows:

    a. Open a new console window and set the Weblogic environment by executing setEnv.cmd.

       Unix: Source the setEnv.sh script present in the server domain directory.

       Portal Domain: Use the setDomainEnv.cmd script (on Unix it is the setDomainEnv.sh script).

    b. Run the following script and ensure that it has the correct username, password, and URL values:

       Windows: *install_dir*/setupNetPointRealm.cmd

       Unix: *install_dir*/setupNetPointRealm.sh

       Note: To use policies based on roles for Web and EJB applications in WebLogic SSPI, run the `setupNetPointRealm` tool with the `sspi_role` parameter.

       For example:

       ```
       Install_dir\setupNetPointRealm.cmd sspi_role
       ```

       Portal Domain: Run the script with parameter "portal".

       WebLogic Server 7.0: The script does not work and NetPointRealm must be set manually.

       WebLogic Server 9.2 on Unix: Set the `domName` variable in the `install_dir/setupNetPointRealm.properties` file. Then run the `Install_dir/setupNetPointRealm_wl92.sh` script.

    c. For WebLogic 9.2, do the following:

       For the portal server, verify that both the Weblogic Default and Oblix Authentication provider are configured.

       Change the control flag for Oblix authentication provider to OPTIONAL, and verify that the control flag for the Weblogic authentication provider is REQUIRED.

    d. Log in to the WebLogic Admin Console, navigate to Domain, Security, Realms and do the following:

       * Verify that NetPointRealm is set as the default.
       * Verify that the security providers are set properly in NetPointRealm.

       Use the following steps for WebLogic Server 9.2:

       * Click Lock and Edit in the WebLogic Admin Console.
       * Navigate to NetpointRealm, Providers, Certification Path, WebLogicCertPathProvider. Select the Current Builder option to use the WebLogicCertPathProvider as the current builder. Click Activate Changes to activate all changes.
       * Set NetPointRealm as the default realm.

         In the left pane, select your domain to open the Settings page for your domain. Click the Security tab. Click General. Select NetPointRealm as

the default security realm. Click Save. Click Activate Changes to activate all changes.

e. **If the script fails:** If the script fails, you must manually add the Oracle Access Manager security realm (NetPointRealm):

   * Go to Domain, Security, Realms and select "Configure a new Realm".
   * For the option "Check Roles and Policies for", ensure that "All Web Applications and EJBs" is selected.
   * Navigate to Providers, Authentication, and configure a new Authenticator and Identity Asserter.
   * **Identity Asserter:** Select the Token Type ObSSOCookie and in the Details tab, uncheck "Base64Decoding Required".
   * **Portal Domain:** Set the control flag of Authenticator to OPTIONAL and also configure a Default Authenticator.
   * Navigate to Providers, Authorization and configure a new Authorizer(for the portal domain, only configure a Default Authorizer).

     For role based policies, you also need to configure a Default Authorization Provider. Navigate to Providers, Authorization and configure a Default Authorization Provider.
   * For role based policies, navigate to Providers, Adjudication and configure a new Adjudication Provider.
   * Navigate to Providers, Role Mapping and configure a new Role mapper (for the portal domain, only configure a Default Role mapper).
   * Navigate to Providers, Credential Mapping and configure a new Default Credential mapper.
   * Navigate to Domain, Security and select this realm as the default realm.

13. **Portal Server Domain:** Complete the following steps to configure a WebLogic Portal domain:

    a. Restart the server using the same WebLogic credentials that were used earlier.

    b. In the WebLogic Server Console, navigate to Domain, Security, Realms, NetPointRealm, Providers, Authentication, and:

       * Remove the Default Authenticator.
       * Change the control flag for Authenticator to REQUIRED.

    c. Using the Group Manager, create a group in Oracle Access Manager that maps to the Admin role in the BEA WebLogic Server and contains all the administrators for the BEA Portal.

       For example:

       BEA_Administrators

    d. Create a user (portaladmin) and add it to the BEA_Administrators group; later you login as this user (portaladmin) when restarting the server.

    e. In the WebLogic Server Console Admin Console, navigate to Security, Realms, NetPointRealm and:

       * Click Groups to display all Oracle Access Manager groups.

* Search for the BEA Admin group that was created in this step. You can use a wild card in the search.
* Copy the group name.

f. Click Global Roles, Admin role, Conditions tab and:

* Add a Role Condition where the caller is a member of the group.
* Paste in the group name you copied.

g. Change the role condition from "and" to "or", then click Apply.

h. Repeat this procedure for the PortalSystemAdministrator role.

Note: Other BEA roles can be mapped to Oracle Access Manager groups/users. When you restart the WebLogic Server, it is important that you are logged in as a user in the Oracle Access Manager group associated with the BEA Admin role.

---

14. Restart the WebLogic Server.

    The next time you log in to the WebLogic console, provide Master Oracle Access Manager Administrator credentials. You will be authenticated using NetPointRealm.

15. If you are using identity assertion as the authentication mechanism that protects Web applications:

    a. Install a WebGate on the proxy Web server. See "Authentication for Web-Only Resources" on page 10-6 for an illustration of this type of installation.

    b. Configure the Oracle Access Manager policies that protect the Web applications to use HTTP as the resource type instead of wl_url.

    Note: There is one exception to the resource type configuration. The WebLogic administration console always uses form login. The /console policy must use the resource type wl_url.

    ---

16. If anything other than an Oracle Access Manager form-based authentication scheme protects the policies configured with the HTTP resource type, configure a challenge redirect parameter to redirect the user to another Web server that has WebGate installed.

    Note: If you do not complete this step, the user will have to refresh the browser to access the desired page because the ObSSOCookie set by the WebGate in the HTTP request has not yet been sent to the WebLogic server.

    ---

17. Continue with the following procedures, as needed:

    - "Configuring the Identity Server" on page 10-36.
    - "Configuring Multiple WebPass Instances" on page 10-37.

Configuring the Identity Server

Next, you complete the following procedure to configure the Identity Server.

To configure the Identity Server

1. Open the oblixappparams.xml file and set the searchstringMinimumLength to zero:

   IdentityServer_install_dir\identity\oblix\apps\common\bin\

   oblixappparams.xml

   ```
   <NameValPair ParamName="searchstringMinimumLength" Value="0"/>
   ```

   where *IdentityServer_install_dir* is the directory where you installed Identity Server.

2. Open the groupservcenterparams.xml file and set the groupMemberSearchStringMinimumLength to zero:

   IdentityServer_install_dir\identity\oblix\apps\groupservcenter\bin\

   groupservcenterparams.xml

   ```
   <NameValPair ParamName="groupMemberSearchStringMinimumLength" Value="0"/>
   ```

3. Restart the Identity Server.

   The next step must be completed after Identity System setup.

4. From the Identity System Console, create an administrator with the required View and Delegated Administration rights.

   Note: This administrator should be the one used for "OB_AdminUserName" parameter in the NetPointProvidersConfig.properties. For more information about configuring administrators, see the *Oracle Access Manager Identity and Common Administration Guide*.

Configuring Multiple WebPass Instances

Oracle Access Manager uses failover to maximize performance and provide uninterrupted service to end users. Failover redirects requests when a server fails. You may want to configure multiple WebPass instances for failover purposes.

This section assumes that you have already installed more than one instance of WebPass for the Security Provider. See the *Oracle Access Manager Deployment Guide* for more information on failover.

To configure multiple WebPass instances

1. Open the NetPointProvidersConfig.properties file in the WebLogic domain directory.
2. Enter the WebPass fully-qualified hostname with the domain name and port number using a comma-separated list.

   For example:

   # WebPass webserver host name and port number

   OB_WebPassHost=foo.domain.com,bar.doman.com OB_WebPassPort=81,80

   In this example, the valid WebPass host:port combinations are:

   o foo.domain.com:81

   o bar.domain.com:80

Using Role Based Policies

The Security Provider for WebLogic SSPI now supports WebLogic role names as policy names. Roles defined for your Web and EJB applications deployed in WebLogic can be mapped to policies in Oracle Access Manager.

Before you can create create policies in Oracle Access Manager that are based on WebLogic roles, you need to set `obRoles.useRoleBasedPolicies=true` in the `NetPointProvidersConfig.properties` file.

You can use the following methods to create role based policies in Oracle Access Manager:

- Use the NetPoint Policy Deployer Tool
- Manually Configure WebLogic Role Based Policies in Oracle Access Manager

Use the NetPoint Policy Deployer Tool

The NetPoint Policy Deployer tool creates the policies in Policy Manager. See "Running the NetPoint Policy Deployer" on page 10-19 for details on running the NetPoint Policy Deployer tool.

Example 10–1 shows the resource, authorization rule, and policy created for a Web application deployed on WebLogic.

Example 10–1 Oracle Access Manager Policy Based on WebLogic Role

The following shows an excerpt from the deployment descriptor (`web.xml` file) used by a Web application deployed on WebLogic:

```
<security-role>
    <description>
      Broker
    </description>
    <role-name>
      Broker
    </role-name>
</security-role>
```

Logical roles in the `web.xml` file are mapped to physical groups or users in the `weblogic.xml` file. The following code shows the corresponding `weblogic.xml` file excerpt:

```
<security-role-assignment>
     <role-name>Broker</role-name>
     <principal-name>BrokerGroup</principal-name>
</security-role-assignment>
```

After running the NetPoint Policy Deployer tool, the following resource, authorization rule, and policy are created for the application in Policy Manager:

Resource name: /BrokerGroup

Resource Details:

Resource type: J2EE_Role

URL prefix: /BrokerGroup

Description: Resource used by the role /BrokerGroup

Authorization Rule Name: Authorization rule for BrokerGroup

Authorization Rule Details:

General tab:

Name: Authorization rule for /BrokerGroup

Description: Authorization rule for role /BrokerGroup

Enabled: Yes

Actions tab: Leave blank.

Allow Access tab: user1k2

Deny Access tab: user1k3

Policy Name: Policy for BrokerGroup

Policy Details:

General tab:

Name: Policy for J2EE_Role /BrokerGroup

Description: Policy for J2EE_Role /BrokerGroup

Resource Type: J2EE_Role

Resource Operation: IN

Resource:/BrokerGroup

Authentication Rules tab:

Name: Use Default

Authorization Expression tab: Authorization Rule for / BrokerGroup

Manually Configure WebLogic Role Based Policies in Oracle Access Manager

If you do not wish to use the Policy Deployer tool, of if you run into errors, you can manually configure the role based policies in Oracle Access Manager. The following steps discuss the operations that you need to perform:

1. Configuring a New J2EE_Role Resource Type and Operation
2. Adding Resources to the Policy Domain
3. Adding Authorization Rules to the Policy Domain
4. Creating Role Based Policies for the Domain

Configuring a New J2EE_Role Resource Type and Operation

Use the following steps to configure a new J2EE_Role resource type and operation:

1. From the Access System Console, click the **Access System Configuration** tab, then click **Common Information Configuration**, **Resource Type Definitions**.

   The List All Resource Types page appears.

2. From the List All Resource Types page, click **Add**.

   The Define a New Resource Type page appears.

3. Define and save the new resource type. For example:

Resource Name: J2EE_Role1

Display Name: J2EE_Role1

Resource matching: Case Insensitive

Resource operation: IN

Note: You can assign any name to the `J2EE_Role` resource type. You should use the same value for the `ObRoles.J2EEResourceType` parameter in the `NetPointProvidersConfig.properties` file.

Adding Resources to the Policy Domain

Next, you need to add resources to the policy domain for WebLogic. Add the resource for user authentication. Also add resources corresponding to the Admin, Operator, Deployer, and Monitor roles in WebLogic. You would also need to add resources corresponding to roles defined in the deployment descriptors of Web applications deployed in WebLogic. Use the following steps to add the resources:

1. From the Policy Manager, click **My Policy Domains** and select the WebLogic (SecuProvForWeblogic) policy domain.
2. Click the **Resources** tab for the policy domain.
3. Click **Add**, then configure, and save the resource for user authentication:

   Resource type: wl_authen

   URL prefix: /Authen/Basic

   Description: Resource used by the Security Provider to authenticate users

   The resource type was defined in "To configure the WebLogic resource types" on page 10-16.
4. Click **Add**, then configure, and save the resource for role Admin:

   Resource type: J2EE_Role1

   URL prefix: /Admin

   Description: Resource used by the role Admin
5. Click **Add**, then configure, and save the resource for role Operator:

   Resource type: J2EE_Role1

   URL prefix: /Operator

   Description: Resource used by the role Operator
6. Click **Add**, then configure, and save the resource for role Deployer:

   Resource type: J2EE_Role1

   URL prefix: /Deployer

   Description: Resource used by the role Deployer
7. Click **Add**, then configure, and save the resource for role Monitor:

   Resource type: J2EE_Role1

   URL prefix: /Monitor

   Description: Resource used by the role Monitor

8. Repeat the preceding step for any application role defined in WebLogic.

   For example, if an application defines the Broker role in its `web.xml` file and the Broker role is mapped to the principal BrokerGroup in the `weblogic.xml` file, then you would need to perform the following step:

   Click **Add**, then configure, and save the resource for role BrokerGroup:

   Resource type: J2EE_Role1

   URL prefix: /BrokerGroup

   Description: Resource used by the role BrokerGroup

Adding Authorization Rules to the Policy Domain

Next, you need to add authorization rules to the policy domain for WebLogic. Use the following steps to add authorization rules:

1. From the Policy Manager, click **My Policy Domains** and select the WebLogic (SecuProvForWeblogic) policy domain name.
2. Click the **Authorization Rules** tab for the policy domain.
3. Click Add, and create the authorization rule for the Admin role. For example:

   General tab:

   Name: Authorization rule for Admin

   Description: Authorization rule for role Admin

   Enabled: Yes

   Actions tab: Leave blank.

   Allow Access tab: Add users whom you want to allow access.

   Deny Access tab: Add users whom you want to deny access.
4. Click Add, and create the authorization rule for the Deployer role. For example:

   General tab:

   Name: Authorization rule for Deployer

   Description: Authorization rule for role Deployer

   Enabled: Yes

   Actions tab: Leave blank

   Allow Access tab: Add users whom you want to allow access.

   Deny Access tab: Add users whom you want to deny access.
5. Click Add, and create the authorization rule for the Operator role. For example:

   General tab:

   Name: Authorization rule for Operator

   Description: Authorization rule for role Operator

   Enabled: Yes

   Actions tab: Leave blank.

   Allow Access tab: Add users whom you want to allow access

   Deny Access tab: Add users whom you want to deny access

6. Click Add, and create the authorization rule for the Monitor role. For example:

   General tab:

   Name: Authorization rule for Monitor

   Description: Authorization rule for role Monitor

   Enabled: Yes

   Actions tab: Leave blank.

   Allow Access tab: Add users whom you want to allow access

   Deny Access tab: Add users whom you want to deny access

7. Repeat the preceding step for any application role defined in WebLogic.

   For example, if an application defines the Broker role in its `web.xml` file and Broker role is mapped to the principal BrokerGroup in the `weblogic.xml` file, then you must perform the following step:

   Click **Add**, and create the authorization rule for the BrokerGroup role. For example:

   General tab:

   Name: Authorization rule for BrokerGroup

   Description: Authorization rule for role BrokerGroup

   Enabled: Yes

   Actions tab: Leave blank.

   Allow Access tab: Add users whom you want to allow access

   Deny Access tab: Add users whom you want to deny access

Creating Role Based Policies for the Domain

The next step is to create the role based policies for the domain. Use the following steps to add policies:

1. From the Policy Manager, click **My Policy Domains** and select the WebLogic (SecuProvForWeblogic) policy domain name.
2. Click the **Policies** tab for the policy domain.
3. Click **Add**, and add the basic authentication policy. For example:

   General tab:

   Name: Basic authentication policy

   Description: Authentication using basic LDAP username and password

   Resource Type: wl_authen

   Resource Operation: LOGIN

   Resource:/Authen/Basic

   Authentication Rules tab:

   Name: Basic authentication rule

   Scheme: Select the basic authentication scheme, NetPoint WebLogic Basic Over LDAP

Authorization Expression tab: Add the rule, Authz Rule for Authen

4. Click **Add**, and add the policy for the Admin role. For example:

   General tab:

   Name: Policy for J2EE_Role Admin

   Description: Policy for J2EE_Role Admin

   Resource Type: J2EE_Role1

   Resource Operation: IN

   Resource:/Admin

   Authentication Rules tab:

   Name: Use Default

   Authorization Expression tab: Authorization Rule for Admin

5. Click **Add**, and add the policy for the Deployer role. For example:

   General tab:

   Name: Policy for J2EE_Role Deployer

   Description: Policy for J2EE_Role Deployer

   Resource Type: J2EE_Role1

   Resource Operation: IN

   Resource:/Deployer

   Authentication Rules tab:

   Name: Use Default

   Authorization Expression tab: Authorization Rule for Deployer

6. Click **Add**, and add the policy for the Operator role. For example:

   General tab:

   Name: Policy for J2EE_Role Operator

   Description: Policy for J2EE_Role Operator

   Resource Type: J2EE_Role1

   Resource Operation: IN

   Resource:/Operator

   Authentication Rules tab:

   Name: Use Default

   Authorization Expression tab: Authorization Rule for Operator

7. Click **Add**, and add the policy for the Monitor role. For example:

   General tab:

   Name: Policy for J2EE_Role Monitor

   Description: Policy for J2EE_Role Monitor

   Resource Type: J2EE_Role1

   Resource Operation: IN

Resource:/Monitor

Authentication Rules tab:

Name: Use Default

Authorization Expression tab: Authorization Rule for Monitor

8. Repeat the preceding step for any application role defined in WebLogic.

   For example, if an application defines the Broker role in its `web.xml` file and Broker role is mapped to principal BrokerGroup in `weblogic.xml`, then you would need to use the following step:

   Click **Add**, and add the policy for the BrokerGroup role. For example:

   General tab:

   Name: Policy for J2EE_Role BrokerGroup

   Description: Policy for J2EE_Role BrokerGroup

   Resource Type: J2EE_Role1

   Resource Operation: IN

   Resource:/Monitor

   Authentication Rules tab:

   Name: Use Default

   Authorization Expression tab: Authorization Rule for BrokerGroup

Configuring Single Sign-On for the WebLogic Portal

Oracle Access Manager supports integration with the WebLogic Portal.

To enable single sign-on between Portal Web Applications and Oracle Access Manager protected resources, the Portal Web Application must be set up for ObSSOCookie handling. The prerequisite to support single sign-on for the Portal Web Application is that it should be using a form-based login portlet for authentication. The "sampleportal" Web Application module a part of "portalApp" Web Application, which is shipped with Weblogic Portal 8.1.3 and is considered as an example.

Following is an outline of the procedures you need to complete to set up single sign-on for the Portal Web Application module.

Task overview: Configuring single sign-on for the WebLogic Portal

1. Edit the web.xml file, as described in "Configuring web.xml to Add Filter-related Nodes" on page 10-45.
2. Configure the login jsp, as described in "Configuring the login or groupspace jsp used by the Login Portlet" on page 10-46.
3. Copy the login filter class, as described in "Copying ObLoginFilter.class in the WEB_INF/classes" on page 10-48.
4. Finish setup, as described in "Completing Setup" on page 10-48.
5. Test your configuration, as described in "Testing Single Sign-On for the WebLogic Portal" on page 10-48.

Configuring web.xml to Add Filter-related Nodes

You need to include filter-related nodes at the start of other filter nodes.

> **Note:** The mapping mentioned in the filter and the POST action URL set in the form should be the same. The only difference is that the action URL will include the context root too and the Oblogin_validate.jsp name will be present.

To add filter-related nodes in WebLogic's web.xml file

1. Locate the Portal Application's Web module's WEB-INF/web.xml.
2. Add the following filter related nodes at the start of other filter nodes:

```
<!-- Login Servlet Filter, required for single sign-on between Portal and
Netpoint  -->
   <filter>
         <filter-name>OblixLoginFilter</filter-name>
         <filter-class>ObLoginFilter</filter-class>
    </filter>
```

3. Add the following filter mapping node at the start of other filter mapping nodes:

```
<filter-mapping>
      <filter-name>OblixLoginFilter</filter-name>
      <!-- Configure this mapping to invoke the Oblix Login filter  -->
         <url-pattern>/portlets/login_validate/*</url-pattern>
     </filter-mapping>
```

Adding Authentication Methods to web.xml

You may configure different authentication methods in the web.xml (deployment descriptor) of a Web application. The following information explains how you specify the authentication schemes in the <auth-method> tag under <login-config>. The following authentication schemes may be configured:

- Form
- Basic
- CLIENT_CERT

To add form-based authentication to WebLogic's web.xml file

1. Add the following to the web.xml file:

```
<login-config>
<auth-method>FORM</auth-method>
<form-login-config>
<form-login-page>/login.jsp</form-login-page> -- location of the jsp page that will accept user credentials
<form-error-page>/fail_login.html</form-error-page> -- location of the page to which request must be redirected upon unsuccessful login
</form-login-config>
</login-config>
```

2. Ensure that the login form contains fields for entering username and password, which must be named j_username and j_password, respectively.
3. Ensure that the form posts j_username and j_password values to j_security_check logical name:

The following example shows how the form should be coded into an HTML page:

```
<form method="POST" action="j_security_check">
<input type="text" name="j_username">
<input type="password" name="j_password">
</form>
```

To add basic authentication to WebLogic's web.xml file

1. Add the following lines to web.xml:

```
<login-config>
     <auth-method>BASIC</auth-method>
     <realm-name> Your_realm_name </realm-name>
</login-config>
```

   where the <auth-method> tag specifies the authentication method (in this case, BASIC), and the <realm-name> tag contains the text that should appear on the dialog box requesting credentials.

2. Save web.xml.

Configuring the login or groupspace jsp used by the Login Portlet

For WebLogic 8.1, you must configure the following in the login.jsp that is used by the login portlet. For WebLogic 9.2, you configure the groupspace.jsp that is used by the login portlet.

To configure the login or groupspace jsp for the Login Portlets

1. For WebLogic 9.2, include the following at the start of the groupspace.jsp:

```
<%@ page import="com.bea.portlet.PostbackURL,
com.bea.netuix.servlets.controls.content.JspContentContext" %>
<% // Set this url as per your setting %>
<%@include file="/portlets/NetPointPortalSSO.jsp" %>
<%
      // Included to get the Base URl for redirection after Authnetication.
   PostbackURL url=PostbackURL.createPostbackURL(request,response);
%>
```

2. For WebLogic 8.1, set the form's action URL as follows:

```
<form method="post" action="/sampleportal/portlets/
 login_validate/Oblogin_validate.jsp" type="POST">
```

   For WebLogic 9.2, set the form's action URL as follows:

```
<form method="post"
action="/groupspace/portlets/login_validate/Oblogin_validate.jsp"
tyle="margin:0px; padding: 0px;">
```

3. For WebLogic 8.1, include following at the start of the login jsp:

```
<%@ page import="com.bea.portlet.PostbackURL,
                com.bea.netuix.servlets.controls.content.JspContentContext" %>
   <% // Set this url as per your setting %>
      <%@include file="/portlets/NetPointPortalSSO.jsp" %>
<%
// Included to get the Base URl for redirection after Authnetication.
JspContentContext jspContentContext =
JspContentContext.getJspContentContext(request);
PostbackURL url = jspContentContext.getBaseUrl(request, response, "");
```

```
%>
```

4. For WebLogic 8.1, set the form's action url as following:

```
<form method="post" action="/sampleportal/portlets/
login_validate/Oblogin_validate.jsp" type="POST">
```

> **Note:** The action URL needs to start with the context root included.

5. Set the user input fields used in login jsp to get the username and password to username and password, respectively.
6. Include a new variable in the login input form:

```
<tr>
 <td align="left"> <input type="hidden" name="targeturl" value=<%= url %> >
</td>
</tr>
```

7. If your form provides logout functionality, set the logout url to Oblogout.jsp. Else in your logout logic include following code to kill the ObSSOCookie

```
<%@ page import="com.oblix.weblogic.configuration.NPConfiguration"%>
<%@ page import="com.oblix.weblogic.logging.ObDebug"%>
<%
    // begin block to kill ObSSOCookie
    // Check if the user has ObSSOCookie
   ObDebug.getInstance().debug("Inside logout.jsp");
    Cookie[] cookies = request.getCookies();
    if ( cookies != null ){
       String obSSOcookie = null;
       for (int i = 0; i < cookies.length; i++) {
   if (cookies[i].getName().equals("ObSSOCookie")) {
   obSSOcookie = cookies[i].getValue();
   // if ObSSOCookie is not null and is not 'loggedout' then
   // kill it by making it loggedout
if (obSSOcookie != null && ! obSSOcookie.equals("") && !
obSSOcookie.equals("loggedout")){

Cookie killedSSOCookie = new Cookie("ObSSOCookie", "loggedout");
   String cookieDomain = NPConfiguration.getCookieDomain();
   if(cookieDomain != null && cookieDomain.length() > 0)
   killedSSOCookie.setDomain(cookieDomain);
   killedSSOCookie.setPath("/");
   response.addCookie(killedSSOCookie);
ObDebug.getInstance().debug("Logout jsp - ObSSOCookie set to loggedout with
domain [" + cookieDomain + "]");
   }
   break;
   }
         }
    }
    // end block to kill ObSSOCookie
%>
```

Copying ObLoginFilter.class in the WEB_INF/classes

The ObLoginFilter.class expects that the name of user input fields used in login form are "username" and "password". However, other names can be used.

To use other names

1. Modify the file provided.
2. Compile the file and include it under the WEB-INF/classes folder.
3. Proceed to "Completing Setup" on page 10-48.

Completing Setup

Use the following procedure to complete the setup process for this implementation.

In the sampleportal example, under the PortalApp application that is shipped as an example with Weblogic 8.1 SP3, Oracle provides a login_validate folder under the portlets folder in the context root. The files Oblogout.jsp and NetpointPortalSSO.jsp are located under the portlets folder. These files are included in the installation directory as follows:

oblix/examples/src/webapp/portalApp/sampleportal

Except for Oblogout.jsp, Oblogin_validate.jsp and NetPointPortalSSO.jsp, you must configure the jsp files for each application.

For WebLogic 9.2, in the context root, portlets folder, Oracle provides a login_validate folder. The files Oblogout.jsp and NetpointPortalSSO.jsp are locaetd under the context root. These files are located in the installation directory as follows:

- oblix/examples/src/webapp/portalApp/sampleportal/Oblogin_validate.jsp
- oblix/examples/src/webapp/portalApp/sampleportal/Oblogout.jsp
- oblix/examples/src/webapp/portalApp/sampleportal/ObloginFilter.class
- oblix/examples/src/webapp/portalApp/groupspace/groupspace.jsp
- oblix/examples/src/webapp/portalApp/groupspace/NetpointPortalSSO.jsp
- oblix/examples/src/webapp/portalApp/groupspace/web.xml

To complete setup

1. Copy Oblogout.jsp under context root of the application.
2. Copy NetpointPortalSSO.jsp under context root of the application.
3. Under context root of your application module, create a folder "login_validate" and copy the Oblogin_validate.jsp.

   This file's contents is displayed only when the filter is not invoked.
4. Continue with "Testing Single Sign-On for the WebLogic Portal" on page 10-48.

Testing Single Sign-On for the WebLogic Portal

You can test the examples provided in the following directory:

Security_Provider_install_dir/examples

These samples allow you to test single sign-on for Web applications and EJBs either using or not using identity assertion for authentication.

There are readmes in the example directories. These readmes provide instructions for testing different types of resources.

For the WebLogic Portal: The examples\src\webapp\portalApp\sampleportal does not contain a full sample that can be deployed.

To test single sign-on for the WebLogic Portal

1. Configure the *WebLogic_install_dir*\samples\domains\portal domain to use the Oracle Access Manager Security Provider.
2. Locate the deployed sampleportal example for WebLogic 8.1 as follows:

   WebLogic_install_dir\samples\portal\portalApp\sampleportal

   Locate the deployed sampleportal example for WebLogic 9.2 as follows:

   WebLogic_install_dir\samples\portal\portalApp
3. Copy and replace the following files:

   For WebLogic 8.1, copy and replace from:

   Security_Provider_install_dir\examples\src\webapp\portalApp\ sampleportal

   To:

   WebLogic_install_dir\samples\portal\portalApp\sampleportal

   Replace the files as indicated in Table 10–1:

Table 10–1 Files To Be Replaced for WebLogic 8.1

New file (in the Oracle Access Manager installation directory)	File to be replaced
login.jsp	\portlets\login.jsp
NetpointPortalSSO.jsp	\portlets\NetpointPortal.jsp
Oblogout.jsp	\portlets\Oblogout.jsp
web.xml	\WEB-INF\web.xml
ObLoginFilter.class	\WEB-INF\classes\ObLoginFilter.class
Oblogin_validate.jsp	\portlets\login_validate\Oblogin_validate.jsp

   For WebLogic 9.2, copy and replace from:

   Deployment path: *Weblogic_install_dir*/weblogic92/samples/portal/portalApp

   To the following :

   Library path: *Domain_dir*//servers/AdminServer/tmp/_WL_user/wlp-groupspace-web-lib/6nsdg

   Replace the files as indicated in Table 10–2:

Table 10–2 Files To Be Replaced for WebLogic 9.2

New file (in the Oracle Access Manager installation directory)	File to be replaced
oblix/examples/src/webapp/portalApp/groupspace/groupspace.jsp	*library path*/groupspace.jsp

Table 10–2 (Cont.) Files To Be Replaced for WebLogic 9.2

New file (in the Oracle Access Manager installation directory)	File to be replaced
oblix/examples/src/webapp/portalApp/groupspace/NetpointPortalSSO.jsp	*library path*/NetpointPortalSSO.jsp
oblix/examples/src/webapp/portalApp/sampleportal/Oblogout.jsp	*library path*/Oblogout.jsp
oblix/examples/src/webapp/portalApp/sampleportal/ObLoginFilter.class	*library path*/WEB_INF/classes/ObloginFilter.class
oblix/examples/src/webapp/portalApp/groupspace/Web.xml	*library path*/WEB_INF/web.xml
oblix/examples/src/webapp/portalApp/sampleportal/oblogin_validate.jsp	*library path*/portlets/login_validate/Oblogin_validate.jsp

4. Either restart the WebLogic Portal or redeploy the PortalApp example.

Authorization Data from an External Source

You can configure the Security Provider for WebLogic to perform an authorization in which the authorization scheme uses external data that resides in the HTTP servlet request for Web applications to determine if a user is allowed access. For example, the authorization scheme can determine that a request must come from a particular IP address, such as the user's home or work machine. The access decision can be based on other factors such as rules. For example, you can allow access only if the user belongs to the Engineering role.

See the *Oracle Access Manager Access System Administration Guide* and the *Oracle Access Manager Developer Guide*for details on authorization schemes that use external data.

Information from a user's role and the ContextHandler is sent to the Access Server based on the configuration of a custom authorization scheme. The ContextHandler is an object that WebLogic passes to the SSPI. The Access Server passes this information to a custom authorization plug-in that makes the authorization decision. It supports three formats for the user parameter field in the authorization scheme definition:

- **RA_roles**: Roles that the user belongs to appear as a comma-separated string.
- **RA_http.method**: This executes the method on an HTTPServletRequest object and returns a string value. For example, RA_http.getRemoteAddr() gets the IP address of the machine where the user sent the request.
- **RA_http.session.method**: This executes the method on an HTTPSession object and returns string values. For example, RA_http.session.getAttribute(myattribute) gets the value of the session attribute named myattribute and returns it as a string. For example, if you want to provide a coupon to a user who has $1,000 of purchases in a shopping cart, you can detect the spending level on a session attribute.

Context-sensitive authorization can be done for EJBs based on the method parameter values apart from the roles. This is done using the reverse action functionality provided by the Access Server. A custom authorization scheme is required for this purpose. You can demonstrate this for a stateless EJB using the following procedure, which limits access to the buy method for the BEAS stock only.

To implement an example

1. Copy the shared library req_context, as follows:

From: *install_dir*/examples/src/webapp/contextAuthz

To: Access Server host, *AccessServer_install_dir*/oblix/lib

2. Navigate to the Access System Console, Access System Configuration, Authorization Management, then click the Add button.
3. In the Shared Library field, enter the path to the req_context.

   For example:

   ```
   c:\OracleAccessManager\access\oblix\lib\req_context
   ```

4. In the User Parameter field, add RA_ejb.Parameter1.
5. In the Required Parameter field, add the following name value pair.

   paramName_1 ejb.Parameter1

   paramValue_1 BEAS

6. Saving the scheme, then restart the Access Server.
7. Create an authorization definition using this authorization scheme in the domain used for SSPI.
8. Modify the policy for the EJB buy method by setting the authorization rule to the authorization definition that you created in the previous step.
9. Run the EJB client.

   You should get the following output, where the buy of BEAS shares goes successful but the buy of MSFT shares fail due to access control.

```
run:
[java]
    [java] Beginning statelessSession.Client...
    [java]
    [java] user: admin
    [java] Creating a trader
    [java] Buying 100 shares of BEAS.
    [java] Buying 200 shares of MSFT.
    [java] There was an exception while creating and using the Trader.
    [java] This indicates that there was a problem communicating with the
server: java.rmi.AccessException:
[EJB:010160]Security Violation: User: 'admin' has insufficient permission to
access EJB: type=<ejb>,
application=_appsdir_ejb20_basic_statelessSession_ear,
module=ejb20_basic_statelessSession.jar, ejb=statelessSession,
method=buy, methodInterface=Remote, signature={java.lang.String,int}.
    [java]
    [java] End statelessSession.Client...
    [java]
```

A custom plug-in similar to this can be written in C to implement the business logic according to your needs.

There is an example of context-specific authorization provided with the security Web application sample in the installation directory. Details for how to configure the authorization scheme is provided in the readme.

Audit Files

To enable auditing, you need to add an auditing provider in the WebLogic security realm. There is a default provider installed when you install WebLogic.

The default auditing provider writes the audit records to the following file:

WebLogic_domain_directory/server/DefaultAuditRecorder.log

where WebLogic_domain_directory is the name of the WebLogic domain and server is the server name.

WebLogic audits all authentication and authorization successes and failures. The Security Provider for WebLogic audits the reason for any failures. The failure logs appear before the WebLogic failure log.

Debug Log Files

WebLogic writes debug logs to the following files:

WebLogic_domain_director/server/server.log

Where *WebLogic_domain_directory* is the name of the WebLogic domain and *server* is the server name.

For debugging, set the configuration parameter ObDebugMode=true in the NetPointProvidersConfig.properties file. See "NetPointProvidersConfig.properties" on page 10-54 for details. The change is takes effect after 60 seconds. Debug logs are written to the server.log file. You can also configure debug logs to print to stdout from the WebLogic administration console.

For more information about a log from the catalog, use WebLogic's CatInfo utility. Set the WebLogic environment by running the setdomainEnv.cmd (or .sh for Unix), and then run:

java weblogic.i18ntools.CatInfo -id message ID

This command also lists the cause action details for the log message.

User Creation/Deletion and Group Creation

The SSPI uses workflows defined in the Identity System to create users and groups and delete users. The data available to be passed in a workflow request is limited by the SSPI interface of WebLogic, as follows:

- **Create User**: The Userid, Password, and Description parameters are available to the SSPI while making a Create User request.
- **Delete User**: The only attribute that is available while deleting a user is Userid.
- **Create Group**: The parameters available while creating a group are Name and Description.

It is possible to define a workflow that uses values for these attributes. It is also possible to send constant values for more attributes as shown, in the following sample workflow definition in Table 10–3.

Table 10–3 Workflow Fields and Values

Workflow Field	Sample Workflow Values
Workflow Name	Name generated by the Oracle Access Manager BEA SSPI Create User Workflow.
Workflow Type	Create User

Table 10–3 (Cont.) Workflow Fields and Values

Workflow Field	Sample Workflow Values
Workflow DN	obworkflowid=wfqs20020806T0907402920,obcontainerId=workflo wDefinitions,OU=Oblix,OU=Company,DC=qalab-vduong,DC=oblix ,DC=com
Workflow Status	Enabled
Description	Workflow generated for COREidBEASSPI
Target	Company:OU=Company,DC=qalab-vduong,DC=oblix,DC=com
Workflow Domain	OU=Company,DC=qalab-vduong,DC=oblix,DC=com
Workflow Steps	Step 1: Name: Initiate Attribute Name: LoginID (Required) Attribute Name: Password (Required) Attribute Name: Name (Required) Participant: admin Step 2: Name: Enable Entry Condition: 1. true:false

Following are the corresponding parameters from the NetPointProvidersConfig.properties file:

```
OBWebPass.CreatUserWorkFlowID=wfqs20020806T0907402920,
obcontainerId=workflowDefinitions, OU=Oblix,
OU=Company,DC=qalab-vduong,DC=oblix,DC=com
OBWebPass.CreatUserWorkFlowDomain=OU=Company,DC=lab-vduong,DC=oblix,DC=com
In this file, $UID$ and $PASSWD$ denote value of login attribute and password,
respectively. The placeholders are passed to the workflow as is and are written to
the user profile. At runtime, both $UID$ and $PASSWD$ are replaced with values
obtained for the login attribute and password.
OBWebPass.CreatUserWorkFlowNumOfFields=3
OBWebPass.CreatUserWorkFlowAttrName_1=cn
OBWebPass.CreatUserWorkFlowAttrValue_1=Name of $UID$
OBWebPass.CreatUserWorkFlowAttrName_2=uid
OBWebPass.CreatUserWorkFlowAttrValue_2=$UID$
OBWebPass.CreatUserWorkFlowAttrName_3=userPassword
OBWebPass.CreatUserWorkFlowAttrValue_3=$PASSWD$
OBWebPass.CreatUserWorkFlowComment=Added user $UID$ from WebLogic portal server.
```

If the workflow is modified to use different attributes, the preceding sample lines in the NetPointProvidersConfig.properties file need to change. If workflow is modified to use another attribute, the DS attribute name must be specified in this file.

For example, if you change the first attribute from cn Name to cn Mail (that is obmail in DS) then do the following:

```
OBWebPass.CreatUserWorkFlowAttrName_1=obmail

OBWebPass.CreatUserWorkFlowAttrValue_1= $UID$@company.com
```

Similarly, if this is a new parameter being added to the workflow, you need to increase the number of fields and add two new lines for attribute:

Increase the number of fields as indicated in the following:

```
OBWebPass.CreatUserWorkFlowNumOfFields=4
```

Add two new lines, as indicated in the following:

```
OBWebPass.CreatUserWorkFlowAttrName_4=obmail

OBWebPass.CreatUserWorkFlowAttrValue_4=$UID$@company.com
```

The same approach is to be followed during group creation and user deletion. For group deletion, determination of the workflow to be used is made by Oracle Access Manager. The delete group operation requires only the Group DN, which is obtained from the Portal Admin interface at runtime.

Configuration Files

The following configuration files are required for the integration of Oracle Access Manager and WebLogic:

- NetPointProvidersConfig.properties
- NetPointWeblogicTools.properties

NetPointProvidersConfig.properties

The NetPointProvidersConfig.properties file describes all common configuration items. There are also items written by the installer when you select the Typical/Advanced install of the Security Provider for WebLogic. A sample file containing all of the parameters in the

A sample file containing all of the parameters in the SampleNetPointProvidersConfig.properties is located in:

Security_Provider_install_dir/examples

where *Security_Provider_install_dir* is the directory where the Security Provider for WebLogic is installed. Table 10–4 describes these parameters.

Table 10–4 NetPointProvidersConfig.properties

Parameter	Description and Value
ObDebugMode	Specifies whether Oracle Access Manager debugging information is recorded in the WebLogic log files. Default=false
ObRoles.Cache.TTL	Specifies the length of time (in seconds) for which security roles are cached. Default=60 seconds
ObRoles.Cache.CleanupSchedule	Specifies the length of time (in seconds) after which expired items in the Roles cache are flushed. This is to reclaim memory. Default=60 seconds
ObAuthorization.ActionOn UnprotectedResource	Specifies default access to resources not protected by Oracle Access Manager. Default=allow

Table 10–4 (Cont.) NetPointProvidersConfig.properties

Parameter	Description and Value
Ob_InstallDir	Specifies the installation directory for the Oracle Access Manager Security Provider.
OB_AuthnSchemeResourceTypeName	The Security Provider uses a special policy to authenticate users in WebLogic. The resource type specified on this parameter is used in this special authentication policy. This resource type contains a resource with a URL prefix of /Authen/Basic. Value: Name of the resource type.
OB_AuthnSchemeResourceName	The Security Provider uses a special policy to authenticate users in WebLogic. The resource specified on this parameter is used in this special authentication policy. This resource contains a URL prefix of /Authen/Basic. Value: Name of the resource.
ObAuthentication.Anonymous.ResourceName	The name of the resource used in the policy for anonymous authentication.
OB_AuthnSchemeOperation	The operations specified on the resource for this policy. Example: LOGIN
ObAuthentication.LoginIdParam	The userID challenge parameter that is used in the authentication scheme for the policy. This value is the same as the userID challenge parameter specified in the credential_mapping plug-in. Example: userid
ObAuthentication.passwordParam	The password challenge parameter that is used in the authentication scheme for the policy. This value is the same as the password challenge parameter specified in the validate_password plug-in. Example: password
OB_AuthzActionType	The authorization action that retrieves the user's login ID from the ObSSOCookie. Example: WL_REALM
OB_AuthzActionName	The action that retrieves the user's login ID from the ObSSOCookie. Example: uid
ObFormAuthenticationSSO.DummyUserName	This dummy user name is used with a form login when single sign-on is configured. In this scenario, there is no WebGate on the proxy HTTP server. See "Completing Advanced Installation" on page 10-13 for details.
ObWebAppResourceTypes	WebLogic resource types that are used for Web applications.
ObRoles.ResourceType	This parameter specifies the resource type defined for the policy that retrieves the user's role information. Example: wl_authen

Table 10–4 (Cont.) NetPointProvidersConfig.properties

Parameter	Description and Value
ObRoles.J2EEResourceType	This parameter supports J2EE_Role type resource in Security Provider for WebLogic. Example: J2EE_Role1
ObRoles.ResourceName	This parameter specifies the URL prefix of the resource (not the resource type) defined for the policy that retrieves the user's role information. Example: /Authen/Roles
ObRoles.ResourceOperation	The operation specified on the resource in this policy. Example: LOGIN
ObRoles.J2EEResourceOperation	This parameter supports J2EE_Role type resource operation for J2EE_Role type resource in Security Provider for WebLogic Example: IN
ObRoles.ActionType	The action specified on the authorization rule for the policy that obtains the user's role. Example: WL_REALM
ObRoles.UseRoleBasedPolicies	Set this to true if you want to create role based policies for Web and EJB applications. Set this to false for resource based policies. **Important:** Do not change the value once this parameter has been set.
ObWebAndEjbResourceTypes	This parameter provides a list of Web and EJB resource types. Value: <url>,<web>,<ejb>
ObAuthorization.AbstainMapsTo	If you do not want to use the Abstain result of an isAuthorized call, you can map the result to allow or deny. See "Completing Advanced Installation" on page 10-13 for details.
ObAuthorization.OnDenyRedirectTo Url	If access to a resource is denied, then you can redirect the user to this page. This is optional.
OB_LogLevel	The logging level that is recorded in the log file. Values are none, info, and debug. This is optional.
OB_LogFileName	The file name for log messages. Default = *install_dir*/log. This is optional.
OB_LogMilliSeconds	The data/time format of log messages in the file specified with OB_LogFileName. When true, log messages are time formatted in milliseconds. Default =true. This is optional.
OB_WebPassHost	The WebPass server host machine name. The host name must be fully qualified; for example, OB_WebPassHost=hostname.acme.com. To configure multiple WebPass instances for failover purposes, separate the names with a comma. For example: OB_WebPassHost=foo.domain.com, bar.domain.com Note that the host name corresponds to the port number in the specified order. See the example in the Ob_WebPassPort description section. This is optional.

Table 10–4 (Cont.) NetPointProvidersConfig.properties

Parameter	Description and Value
OB_WebPassPort	The port number of the host machine. To configure multiple WebPass instances, separate the port numbers with a comma, for example: OB_WebPassPort=80, 81. Note that the host name corresponds to the port number in the specified order. In the example provided in the previous paragraph, the *hostname*:*port* number pairing is as follows: foo.domain.com:80 bar.domain.com:81 For failover to work, all other variables such as user name, credentials and webgate protection must be the same. This is Mandatory.
OB_WebPassIsProtected	Values are true and false. If WebPass is protected, set value=true. This is mandatory.
OB_AdminUserName	Oracle Access Manager requires the Admin username and password to make IdentityXML calls to the WebPass. This is mandatory.
OB_AdminUserCreds	Oracle Access Manager requires the Admin username and password to make IdentityXML calls to the WebPass. Without the password the Security Provider will not work. Note: You need to enter a clear-text password, which the program will encrypt and rewrite to the properties file after the first run. This is mandatory.
OB_CookieDomain	The cookie domain specified in the WebGate installer configuration. Needed if WebPass is protected. For example, .xyz.com. This is mandatory.
OB_CookiePath	The cookie path specified in the WebGate configuration. Needed if WebPass is protected. Default = /. This is mandatory.
OB_WebPassSSLEnabled	Specifies whether WebPass needs HTTPS connection. Values are true and false. Default = false. This is Mandatory.
OB_UserAttr	The unique user identification (for example, uid). This is mandatory.
OB_UserSearchAttr	The DN prefix for users from LDAP (for example, cn). This is mandatory.
OB_GroupSearchAttr	The DN prefix for groups from LDAP (for example, cn). This is mandatory.

Table 10–4 (Cont.) NetPointProvidersConfig.properties

Parameter	Description and Value
OB_WebPassADDomain	Optional. The domain of the Admin user. To be used in case of Active Directory Forest with multiple domains. For example, OB_WebPassADDomain=ou=company,dc=qalab,dc=acme,dc=com The ADDomain must be the same as the default defined in the Identity System. This is optional.
OB_WebPassXPIRecordsReturned	Optional. The number of records to return for getUsers or getGroups. Default = return all. This is optional.
OB_UserGroupsCache_enabled	Enables caching of list groups of which the user is a member. Values are true and false. Maintains a cache of all the groups a logged in user belongs to. This is optional.
OB_UserGroupsCache_timeout	The timeout for cache of the list of groups for a user. The timeout is per user. This value should not be very high. If the user's group membership changes the new membership will only take affect at cache timeout. For example, a value of 3600 equates to 1 hour. This is optional.
OB_GroupMembersCache_enabled	Enables caching of list of groups and list of members in each group. Values are true and false. Stores members for each groups (not a frequently used cache). This is optional.
OB_GroupMembersCache_timeout	Specifies the timeout for cache of list of groups and the list of members in each group. This is optional.
OB_UserAttributesCache_enabled	Enables Caching of User Attributes. Values are true and false. This is optional.
OB_UserAttributesCache_timeout	The timeout for the cache of user attributes. Timeout is for the whole cache. This is optional.
OB_UserAttributesCacheElement_timeout	The timeout for the cached user attributes. The Timeout is per user. This is Optional.
OB_GroupAttributesCache_enabled	Enables Caching of Group Attributes. Values are true and false. This is optional.
OB_GroupAttributesCache_timeout	The timeout for the cache of group attributes. Timeout is for the whole cache. This is optional.
OB_GroupAttributesCacheElement_timeout	The timeout for the cached group attributes. The Timeout is per group. This is optional.

Table 10–4 (Cont.) NetPointProvidersConfig.properties

Parameter	Description and Value
OB_Keystore	Specifies the keystore file used by the Security Provider for WebLogic SSPI when it makes SSL connections to HTTPS WebPass. The keystore contains the requestor's public and private key pairs, X.509 certificate, and certificates for Certificate Authorities trusted to certify responder servers. The keystore is managed using the JDK keytool. For example: *install_dir*/oblix/config/jssecacerts. This is optional.
OB_KeystorePassword	The password for the keystore. This is optional.
OB_UserTabId	For future use. Do not change the default. Default = Employees
OB_GroupTabId	For future use. Do not change the default. Default = Groups
OB_NestedGroupsEnabled	Values are true and false. The default is true. To improve GroupSrvCenter performance when nested groups are not used, set the value to false. ■ Nested groups will not be included in the search; the uniquemember attribute will not be requested in a group search when OB_NestedGroupsEnabled=false. ■ A value of true retrieves the uniquemember attribute in the group search, uses this for nested group computation, then removes it before the group is recorded. This is optional.
OB_DynamicGroupsEnabled	Values are true and false. To improve GroupSrvCenter performance when you are not using dynamic groups, set the value to false. Dynamic groups will not be included in the search. This is optional.
OB_UserPasswordAttr	User Password Attribute. Example: userpassword. This is mandatory.
OB_UserDescriptionAttr	User Description Attribute. Example: description. This is mandatory.
OBDisableEveryoneGroupCheck	Parameter to enable or disable the everyone group check. Values are true and false.The default value is false. Setting this parameter to false enables everyone group check. Setting this parameter to true disables everyone group check.
OBWebPass.CreatUserWorkFlowID	WorkflowID for create user workflow. This is mandatory if user creation is supported.
OBWebPass.CreatUserWorkFlow Domain	WorkflowDomain for create user workflow. This is mandatory if user creation is supported.

Table 10–4 (Cont.) NetPointProvidersConfig.properties

Parameter	Description and Value
OBWebPass.CreatUserWorkFlow NumOfFields	Number of attributes present in the create user workflow. This is mandatory if user creation is supported.
OBWebPass.CreatUserWorkFlow AttrName_{number} OBWebPass.CreatUserWorkFlow AttrValue_{number} pair	Name/Value pair for an attribute during user creation.
OBWebPass.CreatUserWorkFlow Comment	Comment while creating a user.
OBWebPass.DeactivateUserWorkFlow ID	WorkflowID for delete user workflow. This is mandatory if user deletion is supported.
OBWebPass.DelUserWorkFlowNum OfFields	Number of attributes present in the delete user workflow. This is mandatory if user deletion is supported.
OBWebPass.DelUserWorkFlowAttr Name_{number} OBWebPass.DelUserWorkFlowAttr Value_{number} pair	Name/ Value pair for an attribute during user deletion.
OBWebPass.DelUserWorkFlow Comment	Comment while deleting a user.
OB_GroupIDAttr	Group Name Attribute. Example: cn. This is mandatory.
OB_GroupDescriptionAttr	Group Description Attribute. Example: description. This is mandatory.
OB_GroupUniqueMemberAttr	Group Uniquemenber Attribute. Example: uniquemenber. This is mandatory.
OBWebPass.CreatGroupWorkFlowID	WorkflowID for create group workflow. This is mandatory if group creation is supported.
OBWebPass.CreatGroupWorkFlow Domain	WorkflowDomain for create group workflow. This is mandatory if group creation is supported.
OBWebPass.CreatGroupWorkFlow NumOfFields	Number of attributes present in the create group workflow. This is mandatory if group creation is supported.
OBWebPass.CreatGroupWorkFlow AttrName_{number} OBWebPass.CreatGroupWorkFlow AttrValue_{number} pair	Name/ Value pair for an attribute during group creation.
OBWebPass.CreatGroupWorkFlow Comment	Comment while creating a group.

NetPointWeblogicTools.properties

Table 10–5 describes the NetPointWeblogicTools.properties file located in:

SecurityProvider_install_dir/oblix/tools/npWLTools

where *SecurityProvider_install_dir* is the directory where the Security Provider for WebLogic is installed.

This file contains information on the WebLogic policy domain.

Table 10–5 SampleNetPointWebLogicTools.properties Parameters

Parameter	Description and Value
ObWLTools.Debug	Creates a debug file. Default = true
ObWLTools.LogFile	Creates a log file. Default = NetPointWeblogicTools.log
ObPolicyDomain.Name	The WebLogic domain name created by the Policy Deployer. Default = SecuProvForWeblogic
ObPolicyDomain.Description	Description of the WebLogic Domain.
ObPolicyDomain.LoginAttribute	The return attribute in the action of an authentication policy that retrieves the user's login ID from the ObSSOCookie. The return attribute is necessary for Oracle Access Manager single sign-on. Default = uid
ObWLTools.SetupInitialNetpointSSPIPolicies	Sets up the initial Oracle Access Manager policies for WebLogic such as /Authen/Bank. Use the default value when you first run the Oracle Access Manager Policy Deployer tool. When you run the tool subsequently, change value to false. Default = true
ObWLTools.DeployPolicy	If set to true, the tool reads the NetPointDeployPolicy.txt file and creates the policies in Oracle Access Manager.
ObWLTools.UnDeployPolicy	Deletes policies. Default = false
ObWLSDomain.Dir	The local directory where the WebLogic domain is located. Default = c:/bea/user_projects/mydomain
ObWLAuthenticationScheme.Name	The authentication scheme used for WebLogic. This should be created manually before running the tool. Default = Oracle WebLogic Access and Identity
ObWLNoneAuthenticationScheme.Name	The Anonymous authentication scheme used for WebLogic. This should be created manually before running the tool. Default = Oracle WebLogic Anonymous Authentication
ObWLWebResource.usingIdentityAssertion	Whether identity assertion is used to protect Web applications. Default=false.
ObWLWebResource.proxyPrefix	The string trimmed from the beginning of the URL that the user originally specifies, before the request is forwarded to the WebLogic Server. For example, if the URL http://myWeb.server.com/weblogic/foo is requested, the URL forwarded to the WebLogic Server is: http://myWeb.server.com:7001/foo The part of the request that is trimmed is what you specify on the ObWLWebResource.proxyPrefix parameter.

Implementation Notes for Active Directory

The following are issues to consider when implementing the Security Provider for WebLogic SSPI on Active Directory.

- Configuring Security Provider for WebLogic
- Setting a Domain in NetPointProvidersConfig.properties

Configuring Security Provider for WebLogic

The steps to configure Security Provider for WebLogic SSPI for an Active Directory Forest follow.

To configure the Security Provider for an Active Directory forest

1. In the Access System Console, create a new Access and Identity authentication scheme for a domain in the Active Directory Forest.

   The base credentials that you specify in the Plugin(s) field must be the same as the search base that you specified in the directory server profile. See "Supported Versions and Platforms" on page 10-9.

   You need to complete step 2 only if you did not create an administrator during pre-installation setup. Otherwise, skip to step 3.

2. Create a WebLogic administrator in Oracle Access Manager with View and Delegated Administration rights and ensure that the administrator's login identification is unique.

3. Specify the WebLogic administrator as the administrator for the Active Directory forest domain.

   This domain must be the same as the one for which you created the authentication scheme in step 1. To do this, specify values for the OB_WebPassADDomain parameter in the NetPointProvidersConfig.properties file as described in"NetPointProvidersConfig.properties" on page 10-54.

   You can search for users in the parent domain but you cannot search for users in sibling or children domains.

Note: You do not need to create an administrator for every domain in an Active Directory Forest.

Setting a Domain in NetPointProvidersConfig.properties

If you are running Active Directory using multiple domains, you must manually edit the NetPointProvidersConfig.properties file to include a value for the OBWebPassADDomain parameter. For example:

OBWebPassADDomain=dc=xyz, dc=acme, dc=com

The domain must be the same as the domain defined for the default directory server in the Identity System.

See the *Oracle Access Manager Identity and Common Administration Guide* for more information.

To Prepare the BEA WebLogic Server 8.1.x

1. Ensure that your BEA WebLogic server 8.1.3 installation includes Java 1.4, which is required for this integration.
2. On an HP-UX 11i machine, ensure that the following Java-related patches are installed:

Patch numbers			
PHCO_26060	PHCO_26111	PHCO_27731	PHCO_28425
PHCO_29633	PHCO_29959	PHKL_18543	PHKL_20228
PHKL_23226	PHKL_23409	PHKL_24064	PHKL_26008
PHKL_27207	PHKL_27282	PHKL_28488	PHKL_28766
PHKL_29434	PHKL_30073	PHKL_30190	PHNE_23003
PHNE_29473	PHSS_17535	PHSS_24303	PHSS_26972
PHSS_26974	PHSS_26976	PHSS_28879	PHSS_29369
PHSS_29744	PHSS_30010	PHSS_30048	PHSS_30260
PHSS_30500			

   Detailed information about these patches is available at the following web site:

   `http://www.hp.com/products1/unix/java/patches`

3. After installing the BEA SSPI package for HP-UX 11.11, you must set up NetPointRealm in the WebLogic domain by running a script, which, by default, does not have execute permission. Therefore, you must manually assign permission on HP-UX 11.11, then run the setupNetPointRealm.sh script.
4. The NetPointIdentityAsserter provider authenticates the user through the ObSSOCookie. To facilitate this, you must set up a proxy server such as Apache with a WebGate installed on it along with a WebLogic module loader.

   When the WebGate successfully authenticates the user, it generates an ObSSOCookie. Using the WebLogic module loader, Apache redirects the request to the WebLogic server with the ObSSOCookie set, at which point the NetPointIdentityAsserter is invoked.

   However, the WebLogic module loader for Apache 1.3.x is not shipped with BEA WebLogic Server 8.1.3. To test this functionality, you need a proxy server on a platform, such as Solaris. This functionality of the Oracle Access Manager SSPI has been tested against a proxy server on Solaris. It involves the following steps:

   a. Set up a proxy server (Apache, for example) on a platform other than WebLogic (Solaris, for example).
   b. Install a WebGate along with a WebLogic module loader on this proxy server.
   c. Test user authentication and redirection to the WebLogic server.

About Parameter Names in the NetPointProvidersConfig.properties file

You can manage Oracle Access Manager users and groups through the WebLogic Server Console.

If you are migrating or patching the Security Provider for WebLogic SSPI from Oracle Access Manager v7.0.2 (or an earlier version) to Oracle Access Manager 10*g* (10.1.4.0.1), take care to ensure that all the parameters in

NetPointProvidersConfig.properties file are set up as specified in"NetPointProvidersConfig.properties" on page 10-54. Also, read the section "Preparing the WebLogic Environment" on page 10-31.

Setting up Cookies and Header Attributes in SSPI

The procedures for setting cookies and header attributes for SSPI are similar to those for a WebGate.

The cookie and header name and values are specified as return actions for policies defined in NetPoint. SSPI recognizes WL_COOKIE and WL_HeaderVar as return types for cookies and header attributes respectively.

Note that attributes and not actual headers are set in the http request. The attributes set in the request can be extracted by the target application using the httpservletrequest getAttribute () method. To ensure that these attributes can be extracted, make sure that the target application has access to the J2EE interface httpservletrequest.

Tips

The following list is organized alphabetically according to the title in bold and are useful to understand. See also "WebLogic Portal Admin Console Changes" on page 10-66.

changeUserPassword Method: The old password will not be checked as all IDXML calls are made with admin credentials. In any case, the changePassword method is not called by the Portal Admin Console. The Portal Admin Console uses resetPassword method, which has been implemented.

Character Restrictions for User Names: The WebLogic Server Console imposes certain restrictions on characters allowed in user names. While all users are displayed correctly in the Console, user creation/user to role mapping must bind with these restrictions.

Escape Mechanism for BEA WebLogic Server Role Mapping Provider: The BEA WebLogic Server Role Mapping provider is used for defining roles in weblogic. Using the Role Mapping provider, users/groups can be assigned to roles. See "Additional Resources" on page 10-72 for more details about WebLogic Security Roles.

This role mapper cannot handle various special characters in user and group names, even though LDAP permits these characters. To overcome this limitation, the Security Provider implements an escape mechanism that enables you to map any Oracle Access Manager group to WebLogic roles. Use Table 10–6 as a reference to read Oracle Access Manager groups in WebLogic Server/Portal Console.

Table 10–6 Read Oracle Access Manager Groups in WebLogic Server/Portal Console

Special Character	Escaped to Sequence
:	:A
(space)	:B
, (comma)	:C
-	:D
=	:E
?	:F
>	:G

Table 10–6 (Cont.) Read Oracle Access Manager Groups in WebLogic Server/Portal

Special Character	Escaped to Sequence
#	:H
' (single inverted comma)	:I
" (double inverted comma)	:J
*	:K
<	:L
&	:M
~	:N
(	:O
)	:P
{	:Q
}	:R
\t	:S
%	:T
;	:U
|	:V
$	:W

Group Description: Group description will not be available for display immediately after creating a group.

Group Members: Only immediate members of the group can be removed from the group. Only User members will be listed as members of group. This holds for nested groups too.

Group With Latin Characters: In the WebLogic Server Console, when trying to assign a group with "Latin Characters" as a parent group of another user/group, the operation fails because this interface in the console does not handle "Latin Characters".

Group Within a Group: In the WebLogic Portal Console, if you try to create a group inside another group, and the new group is created, it is not added to the parent group. You can add the created group to the desired group using the "Add group to Group" interface in the Console.

User/Group Workflows: In create user/group workflows only one value for each attribute (parameter) should be specified in the configuration file.

User/Group Display: Unique userid (uid) will be used for all display operations for users. Group DN will be used for all display operations for groups.

Wild Card Searches: These searches are supported for users and groups; the match is not case sensitive:

- *: returns all user/groups
- Foo*: returns all users/groups whose name starts with Foo
- *Foo: returns all users/groups whose name ends with Foo
- *Foo*: returns all users/groups whose name contains Foo
- FooBar: only returns a user/group named "FooBar"

Note: For users, a unique user identifier is matched against the pattern. However, for groups a common name is matched. For exact group searches, specify the entire group DN without escape characters.

See also "WebLogic Portal Admin Console Changes" on page 10-66.

WebLogic Portal Admin Console Changes

The following WebLogic Portal Admin Console changes have occurred.

Building a Group Hierarchy Tree for the Oracle Access Manager Authentication Provider: When trying to search for groups in the WebLogic Portal Admin Console, a message may be displayed saying:

The Authentication Provider, Oracle Access Manager Authenticator, has not been configured for GUI tree mode

In this case, only one group can be searched at a time. You can use the following procedure to list all the groups in the Admin Console.

To enable the listing of all groups in the Admin Console

1. From the WebLogic Administration Portal main menu, select Service Administration.
2. In the left pane, select Authentication Hierarchy Service.
3. In the right pane, Provider to Add to Build List field, enter the name of Oracle Access Manager authentication provider exactly (including case sensitivity):

   For example:

   NetPointAuthenticator

   Note: You can find the name of Oracle Access Manager Authentication provider by selecting the Security Provider's tool and expanding the Authentication Providers node.

4. Click Update & Build Tree.

   Now, group hierarchies will be displayed for all operations where groups are involved.

Delegated Administration: In order for delegated administration roles to take effect, the PortalSystemDelegator role needs to be associated with the Oracle Access Manager group that was associated with the Admin role in the WebLogic Server Console during product setup.

Configuring Multiple Policy Domains for Different WebLogic Servers

Using the SSPI, you can configure only one Oracle Access Manager policy domain per WebLogic server instance. However, you can configure and apply a different policy domain to each WebLogic server. A role must have a uniform definition in all policy domains, although the role can be used in different ways in different policy domains.

SSPI maintains the following two configuration files in the WebLogic server domain directory:

NetPointProvidersConfig.properties
NetPointResourceMap.conf

The parameters in these two files facilitate setup of different policy domains on different WebLogic servers. Thus, all the weblogic servers in an organization need not use the same policy domain.

The SSPI integration with WebLogic Server performs three main tasks: Authentication, Role Mapping, and Authorization.

For authentication, SSPI uses the OB_AuthnSchemeResourceName parameter in *NetPointProvidersConfig.properties*. You define a policy in Oracle Access Manager to protect the resource specified by this parameter. Specifying different names for this authentication parameter facilitates different authentication policies for different policy domains. By default, this parameter is /Authen/Basic.

The same principle applies to the policy defined for anonymous authentication through ObAuthentication.Anonymous.ResourceName. The default value for this parameter is /Authen/Anonymous.

Also, different role mapping policies can be defined in different policy domains through the role mapping parameter ObRoles.ResourceName, which resides in NetPointProvidersConfig.properties. By default, this parameter is /Authen/Roles.

For authorization, NetPointResourceMap.conf contains information about mapping WebLogic resources to Oracle Access Manager resources. For example, a WebLogic resource of type url maps to a Oracle Access Manager resource of type wl_url. To make different WebLogic servers refer to resource protection policies defined in different domains, specify different resource mappings for your various WebLogic servers.

For example, one WebLogic server can map url to wl_wrl for HTTP resources and protect resources of type wl_url in its policy domain. A second weblogic server can map url to weblogic_wrl for HTTP resources and protect resources of type weblogic_url in its policy domain.

Note: For a given WebLogic server, the authentication and role mapping policies are picked up from a single policy domain, whereas for authorization, the policies used to protect resources are picked up from any domain that has a policy defined for the resource type being accessed.

Troubleshooting the Security Provider for WebLogic

Problem: I get the following error when starting the WebLogic Server:

```
<May 23, 2003 1:44:07 AM PDT> <Error> <OblixSecurityProviders> <700021>
<Authentication failed for user [admin]. Reason - Unprotected resource LOGIN
Authen:/Authen/Basic used in an ObAuthenticationScheme or ObUserSession
constructor.>
```

Solution: The authentication policy may be disabled or absent. See "Setting Up WebLogic Policies in Oracle Access Manager" on page 10-15 for setup information.

Problem: I created a policy to protect web application (for example, /security) with an authorization rule to allow access to some users. But, in reality everyone is allowed access.

Solution: Ensure the following:

- Enable the policy.
- Do not use host identifiers.
- Use the Access Tester to ensure that the policy is being evaluated for this resource and the user who is allowed to access the resource.

Note that the default behavior of Access Server is to allow access if the resource is not protected. This default behavior can be changed by modifying the "denyOnProtect" flag in the Access Server configuration.

Problem: I want to use the methods of the principal ObWLSUser object in an EJB.

Solution: You can use all the methods exposed by the Principal object, but not methods like "isAuthorized" that are present only in ObWLSUser object. This is because they make JNI calls to Access Manager SDK, which is not present on the client side except when the EJB is being run from another WebLogic Server using Oracle Providers. If you really want to execute methods like "isAuthorized" on the client side, then you must install the Access Manager SDK and call ObConfig.Init() before making any calls. A suggested solution is to get the session token from the ObWLSUser object and then use Access Manager SDK methods to execute methods such as isAuthorized.

Problem: The server does not start up. The log has the following entry

```
####<May 29, 2003 2:07:48 PM PDT> <OFF> <Unknown> <vjain> <examplesServer> <main>
<<WLS Kernel>> <> <000000> <Message text not found - Can't locate bundle for
class>
```

Solution: Security Provider's message resources are bundled in wlNetpoint.jar. Make sure that wlNetpoint.jar is present in the classpath. Refer to "Preparing the Environment" on page 10-11 for details on setting up the required environment variables.

Problem: I get the following error:

```
<May 29, 2003 2:36:40 PM PDT> <Error> <OblixSecurityProviders> <700053> <Exception
encountered when isProtected() called for the resource - Type=wl_svr,
isEnabled=true, URL=/examplesServer, operation=default. Reason - The requested
resource could not be mapped to a policy domain in the Policy database. Check if
the corresponding directory service is up.>
```

Solution: The wl_svr resource type is probably enabled in NetpointResourceMap.txt but the resource type is not yet created in the Access System.

Problem: The installer does not write the complete file and the code expects some default values.

Solution: This problem occurs when some of the values are not specified in the properties file. Use the file in *install_dir*\weblogic8\examples and modify it for your environment.

For example, if an action is not specified for authen to get name later:

```
,Roles:{Regular=Regular} ,Resource:type=<url>, application=security,
contextPath=/security, uri=/admin/edit.jsp, httpMethod=GET
,ContextHandler:HttpServletRequest ,ContextHandler:HttpServletResponse>
<Jun 10, 2003 3:37:21 PM PDT> <Debug> <OblixSecurityProviders> <000000>
<OblixResource got from cache>
<Jun 10, 2003 3:37:21 PM PDT> <Debug> <OblixSecurityProviders> <000000> <Entering
OblixDatabase.isProtected for resource Type=wl_url, isEnabled=true,
URL=/security/admin/edit.jsp, operation=GET>
<Jun 10, 2003 3:37:21 PM PDT> <Debug> <OblixSecurityProviders> <000000> <Is
resource protected? true>
```

```
<Jun 10, 2003 3:37:21 PM PDT> <Debug> <OblixSecurityProviders> <000000> <Entering
OblixDatabase.isAccessAllowed for cn=Drusy Sails,ou=LHuman Resource,ou=Los
Angles,ou=Dealer1k1,ou=Latin America,ou=Ford,o=Company,c=US on resource
Type=wl_url, isEnabled=true, URL=/security/admin/edit.jsp, operation=GET>
<Jun 10, 2003 3:37:21 PM PDT> <Debug> <OblixSecurityProviders> <000000>
<OblixDatabase.isAccessAllowed returned PERMIT>
<Jun 10, 2003 3:37:21 PM PDT> <Debug> <OblixSecurityProviders> <000000> <Found an
authenticated principal of type ObWLSUser>
<Jun 10, 2003 3:37:21 PM PDT> <Debug> <OblixSecurityProviders> <000000> <Got a SSO
token>
. . .
```

Solution: Set values for the following parameters as shown in the NetPointProvidersConfig.properties file:

ObAuthenticationScheme.AuthorizationRule.ActionType=WL_REALM

ObAuthenticationScheme.AuthorizationRule.ActionName=uid

Problem: I get the following stack.

```
<Jun 9, 2003 7:43:04 PM PDT> <Debug> <OblixSecurityProviders> <000000> <userName=
[weblogic_system]>
<Jun 9, 2003 7:43:04 PM PDT> <Debug> <OblixSecurityProviders> <000000> <Entering
OblixDatabase.login() for user weblogic_system>
<Jun 9, 2003 7:43:04 PM PDT> <Debug> <OblixSecurityProviders> <000000>
<Authentication failed for [weblogic_system] with message Unprotected resource
LOGIN Authen:/Authen/Basic8 used in an ObAuthenticationScheme or ObUserSession
constructor.
com.oblix.access.ObAccessException: Unprotected resource LOGIN
Authen:/Authen/Basic8 used in an ObAuthenticationScheme or ObUserSession
constructor.
at com.oblix.access.ObUserSession.initCppSideAuthenticate(Native Method)
at com.oblix.access.ObUserSession.<init>(ObUserSession.java:222)
at com.oblix.weblogic.internal.OblixDatabase.login (OblixDatabase.java:185)
. . .
```

Solution: Check the resource, operation and policy and authorization rules. Some resource is not protected.

Problem: I get the following stack.

```
at com.oblix.weblogic.security.providers.authentication.
OblixLoginModuleImpl.login(OblixLoginModuleImpl.java:161)
at weblogic.security.service.DelegateLoginModuleImpl.login
(DelegateLoginModuleImpl.java:71)
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at sun.reflect.NativeMethodAccessorImpl.invoke (NativeMethodAccessorImpl.java:39)
at sun.reflect.DelegatingMethodAccessorImpl.invoke
(DelegatingMethodAccessorImpl.java:25)
at java.lang.reflect.Method.invoke(Method.java:324)
at javax.security.auth.login.LoginContext.invoke (LoginContext.java:675)
at javax.security.auth.login.LoginContext.access$000 (LoginContext.java:129)
at javax.security.auth.login.LoginContext$4.run (LoginContext.java:610)
at java.security.AccessController.doPrivileged(Native Method)
at javax.security.auth.login.LoginContext.invokeModule (LoginContext.java:607)
at javax.security.auth.login.LoginContext.login (LoginContext.java:534)
at weblogic.security.service.PrincipalAuthenticator.authInternal
(PrincipalAuthenticator.java:329)
at weblogic.security.service.PrincipalAuthenticator.authenticate
(PrincipalAuthenticator.java:282)
at weblogic.security.service.SecurityServiceManager. doBootAuthorization
```

```
(SecurityServiceManager.java:913)
at weblogic.security.service.SecurityServiceManager.initialize
(SecurityServiceManager.java:1036)
at weblogic.t3.srvr.T3Srvr.initializeHere(T3Srvr.java:783)
at weblogic.t3.srvr.T3Srvr.initialize(T3Srvr.java:627)
at weblogic.t3.srvr.T3Srvr.run(T3Srvr.java:337)
at weblogic.Server.main(Server.java:32)
>
<Jun 9, 2003 7:43:04 PM PDT> <Error> <OblixSecurityProviders> <700021>
<Authentication failed for user [weblogic_system]. Reason - Unprotected resource
LOGIN Authen:/Authen/Basic8 used in an ObAuthenticationScheme or ObUserSession
constructor.>
<Jun 9, 2003 7:43:04 PM PDT> <Debug> <OblixSecurityProviders> <000000>
<ObUserSession ctor from username, password failed
com.oblix.access.ObAccessException: Unprotected resource LOGIN
Authen:/Authen/Basic8 used in an ObAuthenticationScheme or ObUserSession
constructor.
at com.oblix.access.ObUserSession.initCppSideAuthenticate(Native Method)
at com.oblix.access.ObUserSession.<init>(ObUserSession.java:222)
at com.oblix.weblogic.internal.OblixDatabase.login (OblixDatabase.java:185)
at com.oblix.weblogic.security.providers.authentication. OblixLoginModuleImpl.
login(OblixLoginModuleImpl.java:161)
at weblogic.security.service.DelegateLoginModuleImpl.login
(DelegateLoginModuleImpl.java:71)
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at sun.reflect.NativeMethodAccessorImpl.invoke (NativeMethodAccessorImpl.java:39)
at sun.reflect.DelegatingMethodAccessorImpl.invoke
(DelegatingMethodAccessorImpl.java:25)
at java.lang.reflect.Method.invoke(Method.java:324)
at javax.security.auth.login.LoginContext.invoke (LoginContext.java:675)
at javax.security.auth.login.LoginContext.access$000 (LoginContext.java:129)
at javax.security.auth.login.LoginContext$4.run (LoginContext.java:610)
at java.security.AccessController.doPrivileged(Native Method)
at javax.security.auth.login.LoginContext.invokeModule (LoginContext.java:607)
at javax.security.auth.login.LoginContext.login (LoginContext.java:534)
at weblogic.security.service.PrincipalAuthenticator.authInternal
(PrincipalAuthenticator.java:329)
at weblogic.security.service.PrincipalAuthenticator.authenticate
(PrincipalAuthenticator.java:282)
at weblogic.security.service.SecurityServiceManager. doBootAuthorization
(SecurityServiceManager.java:913)
at weblogic.security.service.SecurityServiceManager.initialize
(SecurityServiceManager.java:1036)
at weblogic.t3.srvr.T3Srvr.initializeHere(T3Srvr.java:783)
at weblogic.t3.srvr.T3Srvr.initialize(T3Srvr.java:627)
at weblogic.t3.srvr.T3Srvr.run(T3Srvr.java:337)
at weblogic.Server.main(Server.java:32)
>
<Jun 9, 2003 7:43:04 PM PDT> <Debug> <OblixSecurityProviders> <000000> <Entering
OblixLoginModuleImpl.abort>
<Jun 9, 2003 7:43:04 PM PDT> <Critical> <Security> <BEA-090402> <Authentication
denied: Boot identity not valid; The user name and/or password from the boot
identity file (boot.properties) is not valid. The boot identity may have been
changed since the boot identity file was created. Please edit and update the boot
identity file with the proper values of username and password. The first time the
updated boot identity file is used to start the server, these new values are
encrypted.>
****************************************************************
The WebLogic Server did not start up properly.
Reason: weblogic.security.SecurityInitializationException: Authentication denied:
```

```
Boot identity not valid; The user name and/or password from the boot identity file
(boot.properties) is not valid. The boot identity may have been changed since the
boot identity file was created. Please edit and update the boot identity file with
the proper values of username and password. The first time the updated boot
identity file is used to start the server, these new values are encrypted.
****************************************************************
. . .
```

Solution: You are probably missing the Oracle Access Manager configuration file in the WLS Domain directory.

Problem: I get the following stack.

```
<Jun 9, 2003 6:11:39 PM PDT> <Info> <WebLogicServer> <BEA-000377> <Starting
WebLogic Server with Java HotSpot(TM) Client VM Version 1.4.1_02-ea-b01 from Sun
Microsystems Inc.>
<Jun 9, 2003 6:11:41 PM PDT> <Info> <Configuration Management> <BEA-150016> <This
server is being started as the administration server.>
<Jun 9, 2003 6:11:41 PM PDT> <Info> <Management> <BEA-141107> <Version: WebLogic
Server 8.1  Thu Mar 20 23:06:05 PST 2003 246620
WebLogic XMLX Module 8.1  Thu Mar 20 23:06:05 PST 2003 246620 >
<Jun 9, 2003 6:11:43 PM PDT> <Notice> <Management> <BEA-140005> <Loading domain
configuration from configuration repository at /
export/home/bea81/user_projects/mydomain/./config.xml.>
<Jun 9, 2003 6:11:57 PM PDT> <Info> <Logging> <000000> <FileLogger Opened at
./myserver/myserver.log>
<Jun 9, 2003 6:12:03 PM PDT> <Error> <Unknown> <000000> <Unable to access
undefined message, id=700025>
****************************************************************
The WebLogic Server did not start up properly.
java.lang.ExceptionInInitializerError
at sun.reflect.NativeConstructorAccessorImpl.newInstance0(Native Method)
at sun.reflect.NativeConstructorAccessorImpl.newInstance
(NativeConstructorAccessorImpl.java:39)
at sun.reflect.DelegatingConstructorAccessorImpl.newInstance
(DelegatingConstructorAccessorImpl.java:27)
at java.lang.reflect.Constructor.newInstance(Constructor.java:274)
at java.lang.Class.newInstance0(Class.java:306)
at java.lang.Class.newInstance(Class.java:259)
at weblogic.security.service.SecurityServiceManager. createSecurityProvider
(SecurityServiceManager.java:1686)
at weblogic.security.service.RoleManager.initialize (RoleManager.java:147)
at weblogic.security.service.RoleManager.<init> (RoleManager.java:93)
at weblogic.security.service.SecurityServiceManager.doRole
(SecurityServiceManager.java:1417)
at weblogic.security.service.SecurityServiceManager. initializeRealm
(SecurityServiceManager.java:1271)
at weblogic.security.service.SecurityServiceManager.loadRealm
(SecurityServiceManager.java:1216)
at weblogic.security.service.SecurityServiceManager. initializeRealms
(SecurityServiceManager.java:1338)
at weblogic.security.service.SecurityServiceManager.initialize
(SecurityServiceManager.java:1018)
at weblogic.t3.srvr.T3Srvr.initializeHere(T3Srvr.java:783)
at weblogic.t3.srvr.T3Srvr.initialize(T3Srvr.java:627)
at weblogic.t3.srvr.T3Srvr.run(T3Srvr.java:337)
at weblogic.Server.main(Server.java:32)
Caused by: com.oblix.weblogic.configuration.ConfigurationException: Unable to
access undefined message, id=700025
at com.oblix.weblogic.configuration.OblixConfiguration. loadProperties
(OblixConfiguration.java:106)
```

```
at com.oblix.weblogic.configuration.OblixConfiguration.<init>
(OblixConfiguration.java:61)
at com.oblix.weblogic.configuration.OblixConfiguration.getInstance
(OblixConfiguration.java:55)
at com.oblix.weblogic.internal.OblixDatabase.<init> (OblixDatabase.java:47)
at com.oblix.weblogic.internal.OblixDatabase.getInstance (OblixDatabase.java:39)
at com.oblix.weblogic.security.providers.roles.
OblixRoleMapperProviderImpl.<clinit>(OblixRoleMapperProviderImpl.j ava:51)
        ... 18 more
****************************************************************
```

Solution: The Oblix Jar file is not correctly packaged.

Additional Resources

The following URLs are provided for background:

- **Terminology for the Newly Introduced Concepts:**
  http://e-docs.bea.com/wls/docs81/secintro/terms.html
- **Security Introduction:**

  http://e-docs.bea.com/wls/docs81/secintro/
- **Security framework whitepaper:**

  http://www.bea.com/content/news_events/white_papers/BEA_WL_Server_TechSecurity_wp.pdf
- **Types of WebLogic Resources:**

  http://e-docs.bea.com/wls/docs81/secwlres/types.html
- **Roles:**

  http://e-docs.bea.com/wls/docs81/secwlres/secroles.html
- **Web application security deployment descriptors:**

  http://e-docs.bea.com/wls/docs81/security/thin_client.html#1045984
- **EJB security deployment descriptors:**

  http://e-docs.bea.com/wls/docs81/security/ejb_client.html#1033936
- **WebLogic Security Roles:**
  http://e-docs.bea.com/wls/docs81/secwlres/secroles.html

11

Integrating with IBM WebSphere

The Oracle Access Manager Connector for WebSphere enables you to integrate applications running on IBM's WebSphere Application Server with the Access System's authentication and authorization services. The connector also makes users and groups managed by Oracle Access Manager available for authentication and authorization within WebSphere.

This chapter describes how prepare your environment, then install, set up, and test the Oracle Access Manager Connector for WebSphere, and configure your WebSphere Application Server for Oracle Access Manager.

This chapter covers the following topics:

About the Connector for WebSphere

The Oracle Access Manager Connector for WebSphere enables WebSphere Application Server administrators to integrate applications running on WebSphere with Access System authentication and authorization services.

Using the Connector for WebSphere, users who try to access Access System-protected WebSphere resources are challenged and authenticated by the Access System. The connector also makes Identity System-managed users and groups available for authentication and authorization.

Advantages of using the Connector for WebSphere include:

- Providing information about Identity System-managed users and groups to the WebSphere Security Server for authentication and authorization.
- Using the Access System to authenticate users who access WebSphere resources such as JSPs, EJBs, and Servlets.

  You can use Access System authentication schemes to provide single sign-on for Web and non-Web enabled applications.
- Authorizing users who access WebSphere resources.

  Oracle Access Manager supports WebSphere's security role-based authorization. The WebSphere's role-based authorization grants access to a protected resource based on a role (such as Manager) for a user or group. Oracle Access Manager integrates WebSphere Server security roles with the Access System's policy-based authorization.
- Protecting WebSphere Server resources such as administration tools, events, servlets, passwords, JDBC connection pools, JMS destinations, and JNDI contexts.

  You can use WebSphere's role-based access policies to control access to WebSphere resources.
- Enabling single sign-on between an Access System-protected resource and a WebSphere Server resource that is protected with WebSphere Security constraints.

The following is an overview of the integration.

Task overview: Integrating with the WebSphere Application Server

1. Prepare your environment, as described in "Preparing Your Environment" on page 11-11.
2. Install the Connector, as described in "Installing the Connector for WebSphere" on page 11-20.
3. Set up and test the Connector, as described in "Completing Connector Setup" on page 11-26.
4. Configure your WebSphere Application Server, as described in "Configuring WebSphere Application Server v5" on page 11-29.
5. Integrate with the WebSphere Portal Server, if this is part of your environment, as described in "Integrating with WebSphere Portal" on page 11-49.

Oracle Access Manager supports the WebSphere Application Server Network Deployment architecture. The Network Deployment architecture comprises of a set of Application Server nodes managed together as a cell. Use the following steps to integrate Oracle Access Manager with the Network Deployment architecture:

1. Complete the preceding steps, numbered 1 to 5, to separately integrate each standalone node with Oracle Access Manager. You must complete the integration before configuring the Network Deployment architecture.
2. Set up the Network Deployment architecture. For more information, refer to the IBM WebSphere InfoCenter documentation at:

   `http://publib.boulder.ibm.com/infocenter/wasinfo/v5r0/index.jsp?topic=/com.ibm.websphere.base.doc/info/welcome_base.html`

As you complete activities in this chapter, you will see the following path name formats:

Identity_install_dir: The directory where you installed the Identity Server. In your installation, for example, this may be C:\OracleAccessManager. During installation the \identity subdirectory is added to the specified destination making the full path to the directory where you installed the Identity Server *Identity_install_dir**identity*.

WebGate_install_dir: The directory where you installed the Access System WebGate. In your installation, for example, this may be C:\OracleAccessManager\Webcomponent. During installation the \access subdirectory is added to the specified destination making the full path to the directory where you installed the WebGate *WebGate_install_dir**access*.

CWS_install_dir: The directory where the Connector for WebSphere was installed. In your installation, for example, this may be C:\OracleAccessManager\NetPointWASRegistry.

WAS_install_dir: The directory where the WebSphere Application Server is installed. In your installation, for example, this may be C:\IBM\WebSphere\AppServer.

WPS_install_dir: The directory where the WebSphere Portal Server is installed. In your installation, for example, this may be C:\IBM\WebSphere\PortalServer. This directory is used for the Portal logs such as, *WPS_install_dir*/log/appserver-out.log.

When you see one of the notations shown in the previous paragraphs, substitute the appropriate path name for your environment as you complete the step.

The rest of this section discusses the following topics:

- WebSphere Components
- Connector for WebSphere Components

WebSphere Components

For complete support information, see "Supported Versions and Platforms" on page 11-10. The following WebSphere components are used in the integration between WebSphere and Oracle Access Manager:

WebSphere Application Server (WAS): The WebSphere Application Server (WAS) enables secure, high volume transactions and Web services.

- WAS 5.0/5.1 is J2EE 1.3 compliant
- WAS 6 and 6.1 are J2EE 1.4 compliant

> **Note:** Both implement authorization using the EJB1.1 specification for security roles. A security role is a set of permissions for access to Web resources and specific EJB methods.

WebSphere Portal Server (WPS): The WebSphere Portal Server provides single sign-on for portlets based on the Java Authentication and Authorization Services (JAAS). The single sign-on implementation enables portlet developers to extract:

- A user's username and password, and distinguished name (DN)
- The DNs of any groups that the user belong to
- The WebSphere Application Server CORBA Credential
- The LTPA token

Any combination of these objects can be used to provide single sign-on to the portlet's back end. For example, the username and password may be used to create a Basic

Authentication HTTP header. Or the LTPA Token may be used to provide single sign-on to another WebSphere Application Server in the same security domain.

Web Trust Association Interceptor (TAI): The TAI is used for third-party proxy authentications. The WebSphere Application Server uses the TAI to enforce the trust policy between WebSphere and third-party security providers for single sign-on. The TAI enables the Access System to authenticate users who try to access resources on the WebSphere Application Server.

Application Assembly Tool (AAT): The AAT is used to build security-aware applications. You use the AAT to define WebSphere roles and bind them to Identity System users and groups.

Connector for WebSphere Components

The Connector for WebSphere uses the Trust Association Interceptor (TAI), the Identity System, the Access System, and the following components:

NetPointWASRegistry: This is the Connector for WebSphere. The NetPointWASRegistry is a user data store implementation of the WebSphere CustomRegistry in Oracle Access Manager. The NetPointWASRegistry serves as a plug-in to the WebSphere Application Server (WAS).

The WebSphere CustomRegistry is also known as a custom user registry (CUR). The CustomRegistry defines the methods that the WAS uses to perform security operations for applications configured to use them. For example, the WebSphere CustomRegistry may be used to identify attributes such as username and password, and to combine user information from diverse data sources.

The NetPointWASRegistry consists of the Access Manager SDK and Identity XML. The NetPointWASRegistry establishes a native connection between the WAS and Oracle Access Manager, enabling WebSphere administrators to use policy-based security features to control user access to business applications.

IdentityXML: The Connector for WebSphere uses IdentityXML calls to get user and group information from the Identity Server. Typically, you use IdentityXML to integrate the Identity System with external software systems and to perform Identity System functions programmatically rather than using the Identity System GUI.

Access Manager SDK: The Access Manager Software Developer's Kit (SDK) enables you to create an interface that can be built into WebSphere and to create an AccessGate that communicates with the Access Server for authentication purposes. The SDK is installed automatically when you install the Connector for WebSphere. The SDK is used by the TAI.

Custom Member Repository (CMR): The CMR is an extension of the Oracle Access Manager component called NetPointWASRegistry (a custom user registry). It resides on the WebSphere Portal Server. The WebSphere Portal Server uses WebSphere Application Server security for authentication when logging in to the Portal. The WebSphere Portal Server enables users to customize and personalize their experience and uses a component called Member Services to manage information about users, user accounts, user profile attributes, and group memberships.

The CMR is an instance of a Member Services component. The CMR connects the WebSphere Portal Server to the Identity System users and groups. The CMR implements the IBM WebSphere MemberRepository interface, and is used to assign and determine access control to the portlets. The CMR stores user.baseattributes and group.baseattributes. It supports only read operations, not create or modify or delete operations.

The WebSphere Portal Server will use the CMR to make IdentityXML queries like getAttributes for a user for personalization, getGroupMemberships, search users by attribute, and similar functions.

> **Note:** The Connector for WebSphere does not support getGroupMemberships. As a result, in the case of Nested Groups, if you check for inner group membership the parent group details will not be displayed.

For more information, see the "Supported Versions and Platforms" on page 11-10.

Integration Architecture

The integration between WebSphere and Oracle Access Manager can vary depending on if you use only the NetPointWASRegistry or if you also use the Access System's single sign-on. For details, see:

- "Scenario 1: Use of NetPointWASRegistry" on page 11-5.
- "Scenario 2: Architecture for Single Sign-On" on page 11-6.

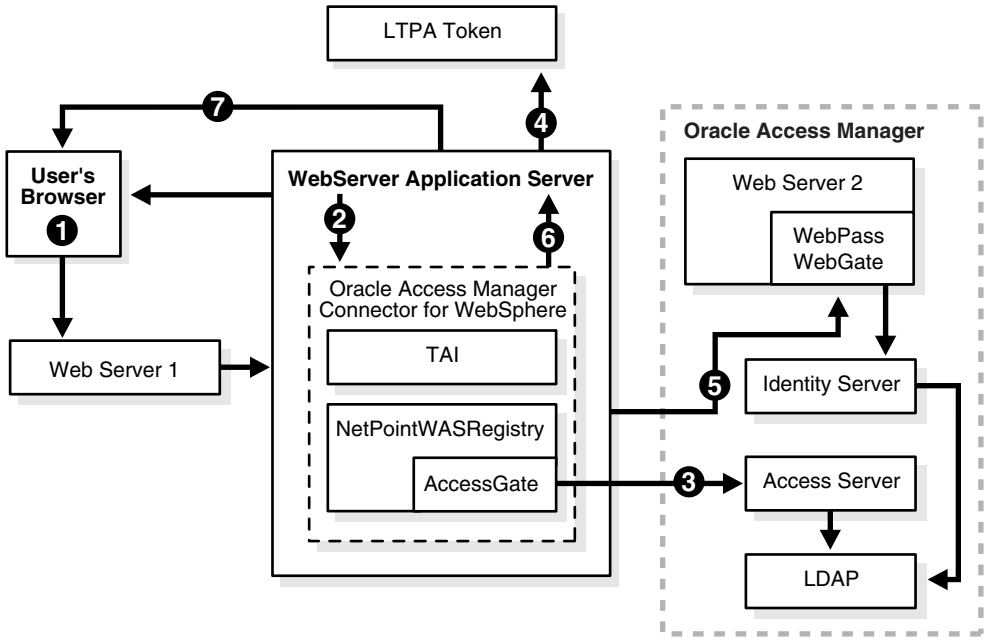
For additional information, see "Mapping Users and Groups to Security Roles in WAS" on page 11-8. See also "Integration Scenario with the Oracle Access ManagerCMR" on page 11-9.

Scenario 1: Use of NetPointWASRegistry

The NetPointWASRegistry obtains Identity System-managed user and group information and performs authentication based on that information.

Figure 11–1 illustrates an implementation of the Connector for WebSphere using the NetPointWASRegistry.

Figure 11–1 Integrating the WebSphere Application Server with the NetPointWASRegistry



In this scenario, use of a WebGate is optional. The WebGate is needed only for single sign-on or to protect the WebPass.

> **Note:** In this scenario, The WAS and both the Web servers must belong to the same domain.

Process overview: Login using WAS with the NetPointWASRegistry

1. A user tries to access a WebSphere resource through a browser.
2. The WAS forwards the user's request to the Connector for WebSphere.
3. The Connector for WebSphere checks with the Access Server and authenticates the user.
4. If single sign-on is enabled in the WAS, an LTPA token is generated.
5. The Connector for WebSphere queries the Identity Server via WebPass for a list of groups to which the user belongs.

   The Identity Server checks the directory and returns the information to the Connector for WebSphere.
6. The Connector for WebSphere returns this information to the WAS.
7. The WAS checks the deployment descriptor for a user-security or group-security role mapping.

   If the user or group belongs to a security role that is allowed to access the resource, the WAS enables the user to access the resource.

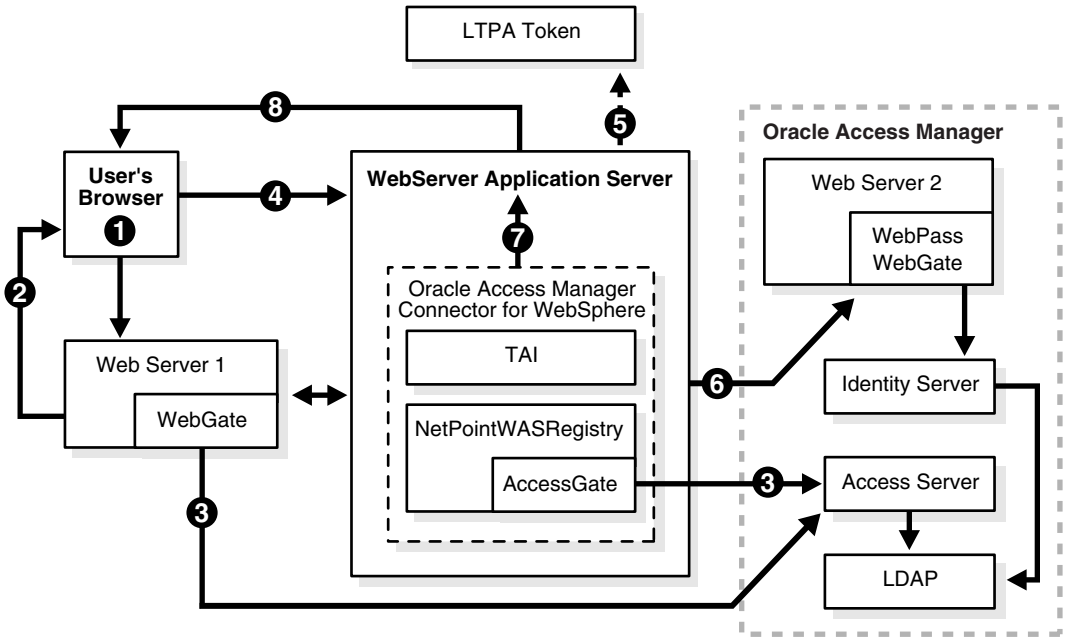
Scenario 2: Architecture for Single Sign-On

The Access System's single sign-on feature enables authenticated users to access protected resources without having to re-authenticate. To use the Access System's

single sign-on, you must enable the TAI and install an AccessGate plug-in on the Web server servicing WAS.

Figure 11–2 illustrates WAS using Access System single sign-on.

Figure 11–2 Single sign-on with the WebSphere Application Server



Note: In this scenario, a WebGate is required for single sign-on.

Process overview: Login using the WAS with Access System single sign-on

1. The user attempts to access an WebSphere resource that is protected by the Access System.
2. WebGate (or AccessGate) intercepts the request and prompts for a username and password, using the Basic challenge method.
3. WebGate passes the user's credentials to the Access Server.

   The Access Server checks the user data store (the directory) and authenticates the user. WebGate sets an ObSSOCookie in the request.
4. The Web server forwards the user request to the WAS.

   The Oracle Access Manager TAI gets the request and confirms that the user has been authenticated.
5. WAS recognizes that the Access System has authenticated the user and creates an LTPA token.

   The remaining steps in this process are the same as steps 5-7 in Scenario 1. The WAS and both Web servers must belong to the same domain.
6. The Connector for WebSphere queries the Identity Server via WebPass for a list of groups to which the user belongs.

The Identity Server checks the directory and returns the information to the Connector for WebSphere.

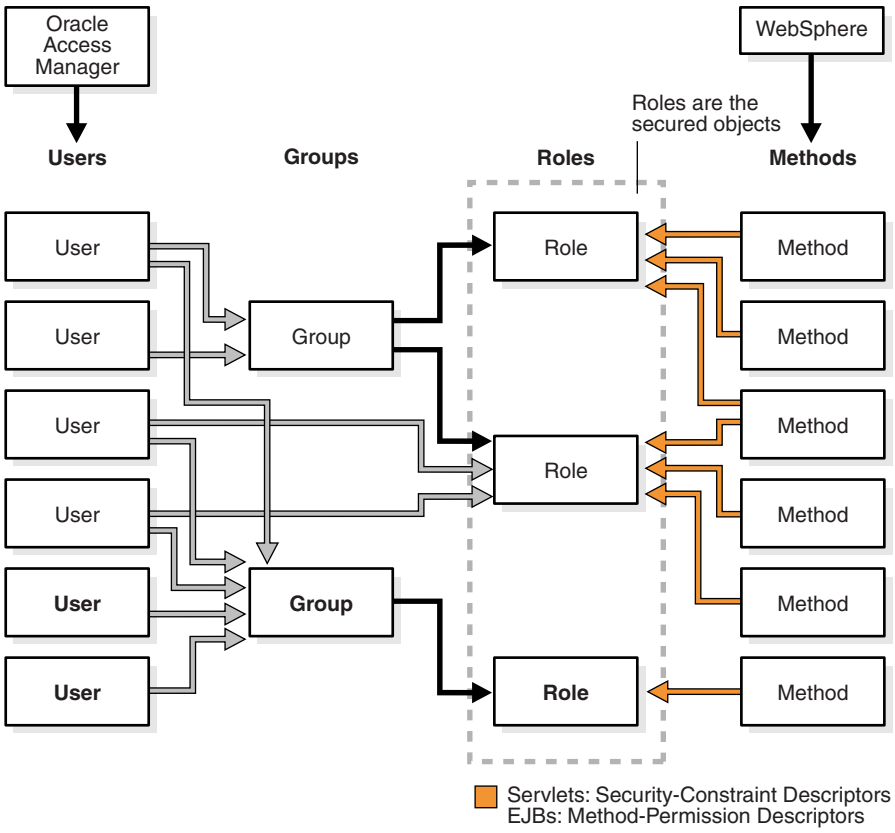
7. The Connector for WebSphere returns this information to the WAS.
8. If you are using the Oracle Access Manager CMR, the Portal Server is invoked, as described in "Integration Scenario with the Oracle Access ManagerCMR" on page 11-9.

   If you are not using the CMR, the WAS checks the deployment descriptor for a user-security or group-security role mapping. If the user or group belongs to a security role that is allowed to access the resource, the WAS allows the user to access the resource.

Mapping Users and Groups to Security Roles in WAS

WebSphere security is consistent with J2EE role-based security specifications. Roles are specified in the deployment descriptors for an application. When the application is installed, these roles are bound to Oracle Access Manager users or groups. You can change the binding information for roles in an application through the WebSphere Administrative Console.

Oracle Access Manager manages users and groups. WebSphere manages roles with the help of the AAT or via the Administration Console. The following illustration shows Oracle Access Manager mappings of Identity System users to Identity System groups and Identity System users and groups to WebSphere roles. It also shows WebSphere mappings of methods to roles.



In the WAS 5 Administrative Console, the role of "Admin" is special. Any user added to the role "Admin" in the Identity System must be in a group called "Admin" or

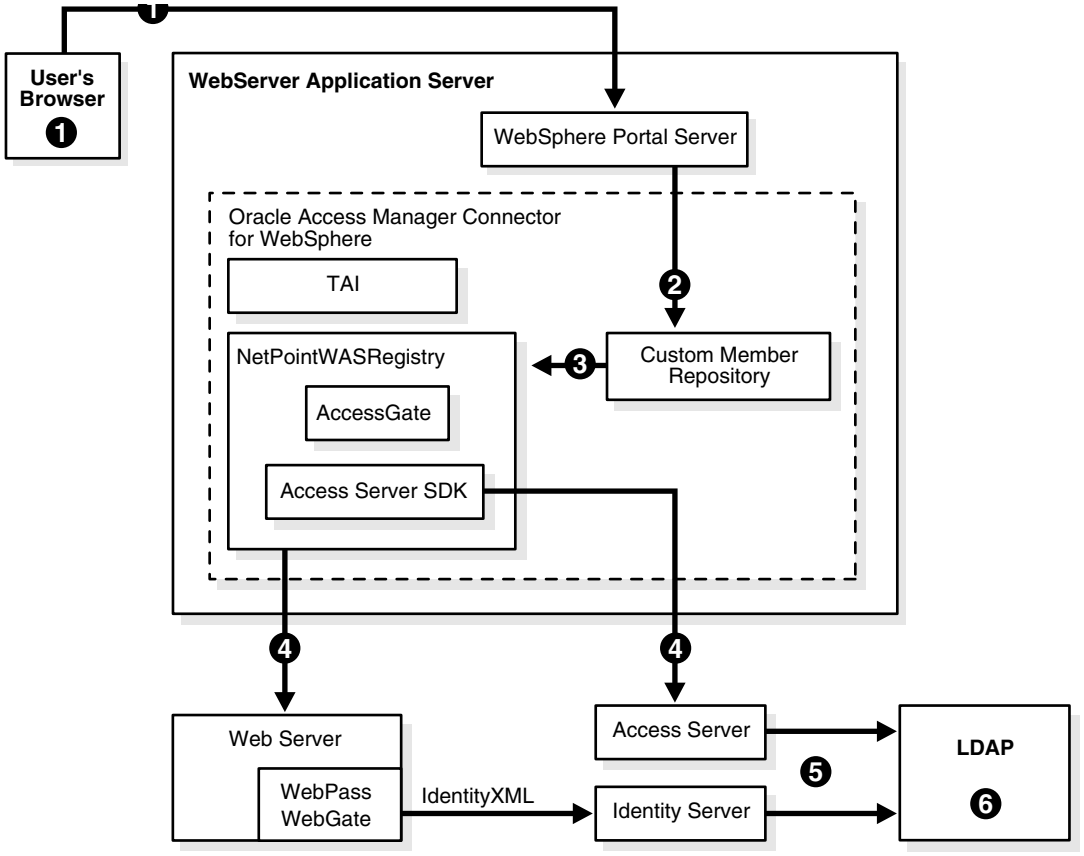
"Administrator". Otherwise, this user may not be able to login to the Administrative Console. Other roles, like "monitor" have no restrictions. Therefore, any Oracle Access Manager user added to these roles in the WAS 5 Administrative Console, can log in to the WAS 5 Administrative Console.

Integration Scenario with the Oracle Access ManagerCMR

During login, the user is authenticated as depicted in "Integration Architecture" on page 11-5. Without the CMR, the WebSphere Portal Server must communicate directly with the LDAP directory server to obtain user, group, and personalization information. With the CMR, communication between the WebSphere Portal Server and the directory server can be eliminated. The CMR performs read operations through the NetPointWASRegistry with the directory server.

Figure 11–3 shows the interaction between the WebSphere Portal Server, CMR, and LDAP directory server during the login authorization process. This follows processes described in "Integration Architecture" on page 11-5.

Figure 11–3 WebSphere Portal Server and the Custom Member Repository



Process overview: Authorization with the CMR

1. After authentication, a user requests access to a portlet through the WebSphere Portal Server.
2. The Portal Server forwards the request to the Oracle Access Manager CMR.
3. The CMR forwards the request to the NetPointWASRegistry.

4. The NetPointWASRegistry sends an IdentityXML call to the Identity Server or uses the Access Manager SDK to contact the Access Server through WebPass or WebGate, depending upon the required method.

   For instance, the Access Manager SDK uses the checkPassword method while IdentityXML uses all other methods:

   - findByAttribute to search users by attribute
   - get
   - getGroupMemberIdentifiers
   - getMemberAttributes
   - getMemberships
   - IsAttributeSupported
   - searchMembers

5. The Identity Server (or Access Server) communicates with the LDAP directory server.
6. The directory server returns information.

Supported Versions and Platforms

The Connector for WebSphere includes the Oracle Access Manager Custom Member Repository (CMR) for the WebSphere Portal Server. The Connector for WebSphere supports the WebSphere Application Server v5.0 and 5.1 with the CMR.

Any references to versions and platforms in this chapter are for demonstration purposes.

You can find support and certification information at the following URL:

`http://www.oracle.com/technology/documentation/`

You must register with OTN to view this information.

Also, you can see the supported versions and platforms for this integration on Metalink, as follows.

To view information on Metalink

1. In your browser, enter the following URL:

   `https://metalink.oracle.com`

2. Log in to MetaLink.
3. Click the **Certify** tab.
4. Click **View Certifications by Product**.
5. Select the **Application Server** option and click **Submit**.
6. Choose **Oracle Identity Management** and click **Submit**.
7. Click **Oracle Identity Management Certification Information 10g (10.1.4.0.1) (html)** to display the Oracle Identity Management page.
8. Click the link for **Section 6, Oracle Access Manager Certification** to display the certification matrix.

Preparing to Install the Connector

Before you can install and configure the Connector for WebSphere, you must complete the following tasks.

Task overview: Preparing to install the Connector for WebSphere

1. Install IBM and Oracle Access Manager components, as described in "Preparing Your Environment" on page 11-11.
2. Configure the Identity Server after installation, as described in "Configuring the Identity System for WAS Integration" on page 11-12.
3. Complete Access System configuration, as described in "Configuring the Access System for WAS Integration" on page 11-14.
4. Set up resource protection, as described in "Configuring Resource Protection in the Access System" on page 11-16.
5. Define a policy domain for the Websphere Administration Console, as described in "Defining a Policy Domain for the WebSphere v6.0 Administration Console" on page 11-19.

Preparing Your Environment

Preparing your environment includes installing the appropriate IBM applications and Oracle Access Manager.

To prepare your environment for integration

1. Ensure that your environment meets the requirements under "Supported Versions and Platforms" on page 11-10.
2. Install and configure the following IBM applications and components using the IBM documentation for these products:
   a. IBM WebSphere Application Server
   b. Application Assembly Tool (AAT)
   c. WebSphere Portal Server.

   The Oracle Access Manager CMR requires the WebSphere Portal Server. References to specific versions are for demonstration purposes. See "Supported Versions and Platforms" on page 11-10 for details.
3. **WebSphere Portal Server v5**: See "Supported Versions and Platforms" on page 11-10 then follow instructions from IBM as you complete one of the following steps to ensure you have an up-to-date version that includes Fix Pack PQ93461:

   Note: Fix Pack PQ93461 is already included incorporated in Portal 5.1.

   - Install a new WebSphere Portal 5.0.2 instance using the installation program provided for v5.0.2.
   - Upgrade an existing installation of WebSphere Portal 5.0 to v5 .0.2.
   - Install a new WebSphere 5.1 Portal instance using the installation program provided for v5.1.
   - Upgrade an existing installation of WebSphere Portal 5.0 or 5.0.2 to 5.1

4. Install and setup Oracle Access Manager components, as described in the *Oracle Access Manager Installation Guide*:
   a. Identity Server
   b. WebPass
   c. Policy Manager
   d. Access Server
   e. WebGate
   f. AccessGate
5. Configure the WAS integration, as follows:
   a. **Identity System**: See "Configuring the Identity System for WAS Integration" on page 11-12 for details.
   b. **Access System**: See "Configuring the Access System for WAS Integration" on page 11-14 for details.
6. Configure Access System resource protection for WAS, as described in "Configuring Resource Protection in the Access System" on page 11-16.

Configuring the Identity System for WAS Integration

The Identity Server and WebPass are the two main components of the Identity System. WebPass is an Access System plug-in for Web servers. When a user requests access to a Web server resource, the WebPass redirects the request to a Identity Server, which then checks the user's identity through the directory server.

After you install the Identity Server and a WebPass, you must set up the Identity System, as explained in the *Oracle Access Manager Installation Guide*. After setup, you configure the Identity System for integration as follows.

Task overview: Configuring the Identity System for WAS integration

1. Prepare for WebPass failover, if desired, as described in "Configuring WebPass Failover" on page 11-12.
2. Setup the Identity Server for WAS, as described in "Configuring the Identity Server" on page 11-12.

Configuring WebPass Failover

Oracle Access Manager uses failover to maximize performance and provide uninterrupted service to end users. Failover redirects requests in the event that a server fails. You must install a WebPass plug-in on each Web server. You may want to install multiple instances of WebPass. Later you may configure these for failover.

For more information, see "Configuring Multiple WebPass Instances for the Connector" on page 11-25.

Configuring the Identity Server

The Identity Server is the Oracle Access Manager component that provides user and group information to the WAS. You must configure the Identity Server to be compatible with search methods that may be used with the WAS.

You will configure the account for the NetPointWASRegistry Admin user. The administrator's login will be used to make IdentityXML calls to the Identity Server. The NetPointWASRegistry Admin user does not need to be the Master Administrator.

For example, the NetPointWASRegistry Admin user may be the Master Identity Administrator. However, if you prefer to limit the rights, the administrator you assign must possess at least the following rights:

- View access on the class attribute of the user class.
- View access on the class attribute of the Group class.
- The appropriate searchbase for the user and group class.
- GRANT+READ right on the class attribute of the user class.
- GRANT+READ right on the class attribute of the Group class.
- View access on those attributes listed in the call. For example: login ID, group member, group dynamic filter, and so on.

To configure the Identity Server after installation

1. Open the oblixappparams.xml file and set the searchstringMinimumLength to zero:

   Identity_install_dir \identity\oblix\apps\common\bin\oblixappparams.xml

   <NameValPair ParamName="searchstringMinimumLength" Value="0"/>

   where *Identity_install_dir* is the directory where you installed the Identity Server.

2. Open the groupservcenterparams.xml file and set the groupMemberSearchStringMinimumLength to zero:

   Identity_install_dir \identity\oblix\apps\groupservcenter\bin\groupservcenterparams.xml

   <NameValPair ParamName="groupMemberSearchStringMinimumLength" Value="0"/>

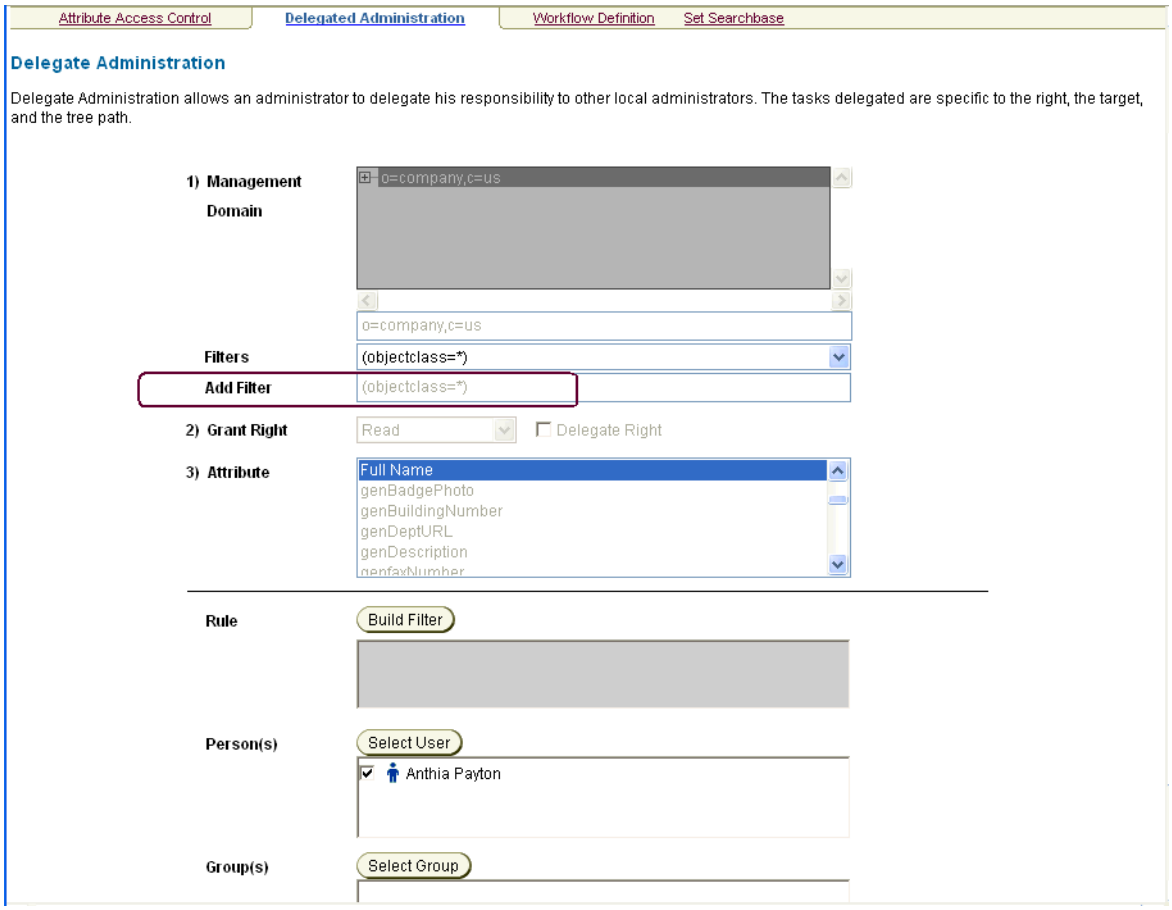
3. Restart the Identity Server.

   The next step must be completed after Identity System setup.

4. From the Identity System Console, click the User Manager tab, then click the Configuration sub-tab.

5. Click the Delegated Administration link and provide an administrator with the required View and Delegated Administration rights, as follows.

Caution: Do not check the Delegate Right box beside Grant Read Right.



6. Restart WAS and the WAS Admin Console to have the change take effect.

   For more information about configuring administrators, see the *Oracle Access Manager Identity and Common Administration Guide*.

Configuring the Access System for WAS Integration

The Access System has three main components: the Policy Manager, the Access Server, and the WebGate. Installation and configuration considerations for Access System components in a WAS integration are discussed here.

Policy Manager: The first component you install is the Policy Manager. You use the Policy Manager to create and manage policy domains to protect resources, and to test policy enforcement. The Access System Console is a part of the Policy Manager. The Access System Console is used for system configuration and system management tasks such as configuring administrators, managing logs, and configuring instances of AccessGates and Access Servers.

Access Server: You also must install at least one Access Server. However, it is recommended that you install at least two Access Servers on two different machines to ensure uninterrupted service to your users.

Each Access Server can be configured to communicate with one or more WebGate instances, and to communicate with a directory server. Access Servers record their activity in Greenwich Mean Time (GMT) because you could have servers operating in several time zones.

WebGate: The WebGate component is optional. You will need it if you want to support single sign-on and if you want to protect the WebPass.

AccessGate: An AccessGate is a an Access System component that processes Web and non-Web resource requests from users or applications. The Connector for WebSphere uses an AccessGate to communicate with the Access Server.

Note: For more information, see "Configuring the AccessGate for WAS Integration" on page 11-15 and the *Oracle Access Manager Access System Administration Guide*.

Configuring the AccessGate for WAS Integration

Before installing the Connector for WebSphere, you must install and configure an AccessGate. WebSphere intercepts user requests and passes them on to the Connector for WebSphere. The Connector uses the AccessGate to make calls to the Access Server for authentication and authorization of the requests.

To configure the AccessGate for the NetPointWASRegistry

1. Navigate to the AccessGate Configuration page:

   From the Access System Console, click Access System Configuration, then click AccessGate Configuration.

2. Click the Add button to display the Add a new AccessGate page.

3. Complete the following information:

   - **AccessGate Name**: Unique, descriptive name for this AccessGate. Use an alphanumeric string, and do not include spaces in the name.
   - **Hostname**: Name of the machine where the AccessGate will be installed.
   - **Port**: You do not need to specify a port. An AccessGate, unlike a WebGate does not require a port number. See the *Oracle Access Manager Access System Administration Guide* for details.
   - **AccessGate Password and Re-type AccessGate Password**: Unique password to verify and identify the component regardless of the transport security mode. This should differ for each AccessGate instance.
   - **Access Management Service**: This service only needs to be enabled if the Access Server that you are associating with this AccessGate has Access Management Service set to On.
   - **Transport Security**: Level of transport security to and from the Access Server. The default value is Open. The transport security mode of the AccessGate must match the transport security mode of the Access Server.

   See the *Oracle Access Manager Access System Administration Guide* for details on transport security.

4. Click Save.

   The Details for the AccessGate page appears.

5. Click the List Access Servers (or List Clusters) button to associate this AccessGate with the appropriate Access Server.

6. Click the Add button to add a new Access Server to associate with this AccessGate.

7. Select an Access Server from the list, and define the configuration for this Access Server.
8. Review the page that is returned.

For more information about AccessGates, see the *Oracle Access Manager Access System Administration Guide*.

Configuring Resource Protection in the Access System

The following procedures must be completed to configure the Access System to protect resources for WebSphere.

Task overview: Configuring resource protection in the Access System

1. Identify a resource type, as described in "Defining a Resource Type for WebSphere" on page 11-16.
2. Create an authentication scheme, as described in "Defining an Authentication Scheme for WebSphere" on page 11-17.
3. Create a policy domain in the Access System, as described in "Defining a Policy Domain for WebSphere" on page 11-17.
4. For WebSphere Application Server v6.0, create a policy domain in the Access System, as described in "Defining a Policy Domain for the WebSphere v6.0 Administration Console" on page 11-19.

Defining a Resource Type for WebSphere

Define a resource type for WebSphere, as described in the following procedure. See the *Oracle Access Manager Access System Administration Guide* for details on defining resource types.

In the following procedure you must provide the resource type values exactly as specified. If you need to specify a different resource name, you must change the resource name in the following locations:

- *CWS_install_dir* \oblix\config\NetPointWASRegistry.properties

  where *CWS_install_dir* is the directory where the Connector for WebSphere was installed.

- *WAS_install_dir* \properties\webgate.properties

  where *WAS_install_dir* is the directory where the WebSphere Application Server is installed.

By default, these configuration files assume you entered the values specified in the following procedure.

To define a resource type for WebSphere

1. From the Access System Console, click Access System Configuration, then click Common Information Configuration.
2. Click Resource Type Definitions.

   The Details for Resource Type page appears.
3. Define and save the resource type as follows:
   - **Resource Name**: Authen
   - **Display Name**: WebSphere Authentication Scheme

- **Resource Matching**: Case Insensitive
- **Resource Operation**: LOGIN.

4. Restart the Access Server to make this new resource available.

Defining an Authentication Scheme for WebSphere

You need to define an authentication scheme for WebSphere. The authentication scheme provides a method to be used when determining whether a user is allowed to access an Access System-protected WebSphere resource.

> **Note:** See the *Oracle Access Manager Access System Administration Guide* for more information on authentication schemes.

To define an authentication scheme for WebSphere

1. From the Access System Console, click Access System Configuration, then click Authentication Management.
2. Define and save the authentication scheme as follows.
   - **Name**: WebSphere Basic Over LDAP
   - **Description**: Used to protect WebSphere-related URLs
   - **Level**: Enter a number for security level that is lower than or equal to the level specified in the authentication scheme protecting the WebSphere and Portal Server URLs. For details, see "Configuring the TAI for WebSphere v5" on page 11-32 or "Configuring the TAI for WebSphere 6 and 6.1" on page 11-43.
   - **Challenge Method**: Basic
   - **Challenge Parameter**: Set the challenge parameter for the user credentials that you want to map.
   - **SSL required**: No
   - **Challenge Redirect**: Enter information in this field if you are going to redirect the end user's request to another server for the authentication process. The most common use of this field is to redirect from a non-SSL server to an SSL server. Redirection is transparent to the end user.
   - **Plug-ins**: Select Access System plug-ins to create a customized challenge scheme. For example, the Oracle Access and Identity challenge scheme requires the validate_password and credential_mapping plug-ins.

   Now you need to create a policy domain for the WebSphere Application Server, as described next.

> **Note:** See the *Oracle Access Manager Access System Administration Guide* for details on policy domains.

Defining a Policy Domain for WebSphere

You need to use the Access System to create a policy domain for WebSphere. This policy domain identifies the authentication scheme that WebSphere will use to protect the resource type that you configured in "Defining a Resource Type for WebSphere" on page 11-16. You also define an action in the Access System. This action creates a user attribute for the WebSphere Application Server. When a user is authenticated to

WebSphere, the user ID defined on the action is sent to the WebSphere Application Server.

To create a policy domain for WebSphere

1. From the Policy Manager, click Create Policy Domains.
2. Click the General tab and enter and save the information for your organization. For example:
   - **Name**: WebSphere
   - **Description**: Policy Domain Used for WebSphere
   - **Enabled**: Yes
   - **Update Cache**
3. Click the Resources tab, then enter and save the following information for your organization. For example:
   - **Resource Type**: WebSphere Authentication Scheme

     This is the resource you defined earlier. If you changed the name of the WebSphere resource type, ensure that you specify it here.
   - **URL Prefix**: /Authen/Basic
   - **Description**: NetPointWASRegistry uses this resource for authenticating users in the Access System. Do not delete this resource.
4. Click the Default Rules tab, Authentication Rules, Add to create and save a default authentication rule using the WebSphere Basic Authentication Scheme that you defined earlier.
   - **Name**: WebSphere
   - **Description**: Default authentication rule using the WebSphere Basic Authentication Scheme.
   - **Authentication Scheme**: Basic Over LDAP type

   You must create a Basic Over LDAP authentication scheme. The Access Manager SDK uses only this type of authentication scheme. You can create a new authentication scheme with a different name for WebSphere, but the authentication scheme type must be Basic Over LDAP.
5. Click the Authorization Rules tab, then create and save an authorization rule to allow access to WebSphere resources. For example:
   a. Click General, then enter and save:
      * **Name**: Allow Everyone.
      * **Description**: Allow access to WebSphere resources.
      * **Enabled**: Yes
      * **Allow takes Precedence**: Yes

      By default, nobody is allowed. Changing the default to Allow Everyone enables the Access Server to check a user's identification for authentication.
   b. Click Actions (Authorization Success), then enter and save:
      * **Return Type**: WAS_REGISTRY
      * **Name**: uid

This is a hard-coded value. Use this exact string.

* **Return Attribute**: login attribute.

  This attribute must be the same as the attribute configured for the login semantic type in the Identity Server or a unique attribute in the user's profile, such as uid.

c. Click Allow Access, then enter and save:

* **Role**: Anyone

6. Click OK to create the new policy domain.

You can now install the Connector for WebSphere.

Defining a Policy Domain for the WebSphere v6.0 Administration Console

For Websphere 6.0, you must use the Access System to create a policy domain for the WebSphere administration Console. This domain protects the console URLs in the ibm/console and /admin folders, and it is required when TAI is enabled for the WebSphere AppServer. The domain helps set the TAI cookie required by the TAI component, and it also redirects the access URL to the application server Administration Console (port 9060). Under these conditions, you must also set an Authorization Success URL

To create a policy domain for the WebSphere Administration Console

1. Navigate to Policy Manager, Create Policy Domains, General.

2. Enter and save the following information for your organization.

For example:

- **Name**: Protect WebSphere Admin Console
- **Description**: Policy Domain Used for the WebSphere Administration Console URL

3. Click the Resources tab, then enter and save the following information for your organization

For example:

- **Resource Type**: http
- **Host Identifier**: The host identifier defined for the machine hosting the Web server.
- **URL Prefix**: `/admin` and `ibm/console`
- **Description**: Used by NetPointWASRegistry TAI component.

4. Click the Authorization Rules tab, then create and save an authorization rule to allow access to WebSphere Admin Console resources

For example:

a. Click General, then enter and save:

* **Name**: Allow Administrator.
* **Description**: Allow access to WebSphere Admin Console resources.
* **Enabled**: Yes
* **Allow takes Precedence**: Yes

By default, nobody is allowed. Changing the default to only Administrator User enables the Access Server to check a user's identification for authentication.

b. Click Actions (Authorization Success), then enter and save the following:

* Redirect to: `http://hostname:9060/ibm/console`

  If the Client Cert Authentication Scheme is to be used, set the associated variables as:

* Redirect to: `https://hostname:9043/ibm/console`

  Where hostname is the fully qualified domain name of the machine hosting the WebSphere AppServer.

c. Click Allow Access

* Select the User who has Administrative rights, then click Save.

5. To create a default authentication rule using the desired authentication scheme, (Basic Over LDAP, Form-based, Client Cert), navigate to Default Rules, then click Authentication Rules, then click Add. Enter and save the following values:

- **Name**: WebSphere Default Scheme
- **Description**: Default authentication rule for Admin Console.
- **Authentication Scheme**:

Create this scheme as a Basic Over LDAP, Form-based, or Client Cert authentication scheme.

6. To create an authorization expression using the "Allow Administrator" authorization rule created previously, navigate to Default Rules, Authorization Expression, Add, then click Select Authorization Rule - "Allow Administrator". Next, click Add and Save

7. Install the Connector for WebSphere.

8. After you install and set up the connector and have also enabled TAI, enable the policy by completing the following steps:

a. Navigate to General, Modify.

b. Set Enable to YES, then click Save.

At this point, you can access the WebSphere Administration Console through either of the following URLs:

`http://`*`WebServerFQDN`*`:`*`port`*`/admin`

`http://`*`WebServerFQDN`*`:`*`port`*`/ibm/console`

Where *WebServerFQDN* is the fully qualified domain name of the machine hosting Web server and *port* is the port number used by the Web server.

Installing the Connector for WebSphere

After completing the prerequisites described in the previous sections, you are ready to install the Connector for WebSphere, as follows.

Caution: **If you plan to include the WebSphere Portal Server in your integration, you must install and configure the portal before you install the Connector for WebSphere. See "Preparing Your Environment" on page 11-11.**

Tip: For information on upgrading this integration, see the *Oracle Access Manager Upgrade Guide*.

Task overview: Installing the Connector

1. Run the installer, as described in "Launching the Installation" on page 11-21.
2. Create an installation directory, as described in "Defining the Installation Directory" on page 11-21.
3. Configure the Connector, as described in "Specifying Connector Details" on page 11-22.
4. Configure the WebGate, as described in "Completing Details for the WebGate" on page 11-23
5. Configure the AccessGate, as described in "Specifying AccessGate Details" on page 11-24.
6. Install a certificate, as described in "Installing a Certificate" on page 11-25
7. Configure WebPass, as described in "Configuring Multiple WebPass Instances for the Connector" on page 11-25.

Launching the Installation

The initial installation and setup procedure differs depending on the platform on which you are installing Oracle Access Manager.

To launch installation

1. Insert the installation CD.

   The DemoShield launches.

2. Launch the program according to your platform:

   Windows: Navigate to Install Oracle Access Manager, Access System, Connector for WebSphere.

   Unix: Complete these steps:

   - Navigate to /Software/Solaris/AccessSystem/Connector for WebSphere.
   - Execute /COREidx.x_EN_sparc-s2_Connector_for_WebSphere.

   The install wizard launches.

Defining the Installation Directory

In this sequence, you will accept the terms of the license agreement and identify the installation directory for the Connector for WebSphere.

You need to specify the installation directory for the Connector for WebSphere on the machine where you installed WAS.

To define the installation directory

1. Click Next to dismiss the Welcome Screen.
2. Read and accept the terms of the license agreement, then click Next to continue.
3. Respond to the next question based upon your platform. For example:
   - **Windows**: Click Next if you are logged in with administrator rights (otherwise click Cancel, log in as a user with administrator privileges, then restart the installation).
   - **Unix**: Specify the username and group, then click Next.

     Typically, the defaults are "nobody".

   You are asked to specify the installation directory for the Identity Server. When you do this and click Next, the installation will begin and you will not be able to return to restate the name.
4. Accept the default directory by clicking Next (or change the destination, then click Next).

   For example:

   \Netpoint

   You are informed that the connector is being installed, which may take several seconds.

   The Configuration for Oracle Access Manager Connector for WebSphere screen appears.

Specifying Connector Details

You need to configure the WebPass for the Connector for WebSphere. For failover purposes, you can configure multiple WebPass instances during installation or through the NetPointWASRegistry.properties file.

To configure multiple WebPass instances during installation, when you enter the WebPass hostname and port number be sure to use a comma as a separator. The hostname must be fully qualified with the domain name. For example:

Hostname: foo.domain.com, bar.domain.com

Port Number: 80, 81

Where the valid WebPass *host:port* combinations are:

- `foo.domain.com:80`
- `bar.domain.com:81`

For details about configuring multiple WebPass instances through the properties file, see "NetPointWASRegistry.properties" on page 11-71.

To specify Connector for WebSphere details

1. Enter the information requested:
   - **Hostname of WebPass**: The fully qualified name of the machine on which WebPass is installed.
   - **Port Number of WebPass**: The port number for WebPass.
   - **Is the Identity System (WebPass) protected by WebGate**: Specify whether the Identity System (WebPass) is protected by a WebGate.

2. Click Next.

   If the Identity System WebPass is protected by a WebGate, the WebGate configuration screen appears.

Completing Details for the WebGate

The following procedure describes how to provide WebGate configuration details.

> **Note:** If you have chosen to use WebGate to protect WebPass, the assumption is that you are protecting the Identity System applications with policy domains. It is also assumed that single sign-on between these components has been configured correctly.

To complete WebGate configuration details

1. Enter the cookie domain for the WebGate (for example, .domain.com).

   The ObSSOCookie is then recognized by all servers within this domain.

2. Enter the cookie path (/).

3. Click Next.

4. Specify whether WebPass requires an HTTPS connection.

   This is the SSL for secure connection when WebPass runs on HTTPS.

5. Specify the user attribute.

   This attribute must be the same as the attribute configured for the Login semantic type in the Identity Server or a unique attribute in the user's profile such as uid.

6. Specify the user search attribute.

   The user attribute and the user search attribute cannot be the same.

   This attribute must be the same as the attribute configured for the DN Prefix semantic type for the person object class in the Identity Server. The person object class type must a structural object class. The administrator of your directory server sets this search attribute.

7. Specify the group search attribute.

   This attribute must be the same as the attribute configured for the DN Prefix semantic type for the group object class in the Identity Server. The group object class a structural object class. The administrator of your directory server sets the group search attribute.

8. You can select Yes to specify that you want the Connector for WebSphere's jar files to be copied from the Connector for WebSphere installation directory to the following directory:

   WAS_install_dir /classes

   Where *WAS_install_dir* is the directory where you installed WebSphere.

   For WebSphere 6.x and later versions, the classes folder is not present by default. Create the classes folder under the *WAS_install_dir* and click OK to copy the JAR files.

   If you select No, copy the jobaccess.jar and NetPointWASRegistry.jar files manually to the *WAS_install_dir*/classes after installation. Or, add the location of these jar files to the WebSphere runtime classpath.

9. Be sure that .../classes folder path is mentioned in the WAS CLASSPATH in setupCMDline.sh script.

   This script is located in the folder *WAS_INSTALL_DIR*/bin.

   This enables the NetPointWASRegistry.jar files to be searched when WebSphere starts up.

10. Click Next.

    The WebSphere Classes Directory screen appears.

Specifying AccessGate Details

As described in the following procedure, you specify the transport security mode for the AccessGate and other AccessGate details. See the *Oracle Access Manager Access System Administration Guide* for details on AccessGates.

To specify AccessGate details

1. Select the same transport security mode that is specified for the Access Server

   The pages that you see next vary depending on the transport security mode that you selected. See the *Oracle Access Manager Installation Guide* for details.

2. Click Next.

3. Enter the following information for the AccessGate:

   - **AccessGate ID**: Enter the name you specified earlier when adding an AccessGate in the Access System Console.
   - **Access Server ID**: Enter the name of the Access Server you associated with this AccessGate.
   - **Password for AccessGate**: Enter the AccessGate password you specified earlier when adding an AccessGate in the Access System Console, if applicable.

     You can specify any Access Server that is associated with this AccessGate.
   - **Hostname where Access Server is installed**: Enter the fully qualified hostname for the Access Server you associated with this AccessGate; for example, stontium.oblix.com.
   - **Port Number Access Server Listens To**: Enter the port of the Access Server you associated with this AccessGate.
   - **Global NetPoint Access Protocol Pass Phrase**: Enter a pass phrase for all Oracle Access Manager components such as Access Server and WebGate.

     This field appears only if you specify Simple Mode or Cert Mode.
   - **Global NetPoint Access Protocol Pass Phrase Confirmation**: Reenter the pass phrase to confirm it.

4. Click Next.

   A new AccessGate Configuration screen appears only if you specify Simple Mode or Cert Mode.

   - If you need a certificate for transport security, select Request for Certificate.

     The Access System sends out a request for a certificate.
   - If you already have a certificate, select Install Certificate to install it.

5. Click Next.

   If you selected Install Certificate, the AccessGate Configuration screen reappears.

Installing a Certificate

You must provide the paths to the certificate, chain, and key files.

> **Note:** If the Installation fails unexpectedly or you need to change these settings later, you can run the configureAccessGate tool located in *CWS_install_dir*\oblix\tools\configureAccessGate, where *CWS_install_dir* is the directory where the Connector for WebSphere is installed.

To supply the paths to the certificate files

1. Enter the full paths to the certificate, chain, and key files.

   The certificate consists of these files. If necessary, click Browse to navigate to the location of these files.

2. Click Next to display the summary screen and then click Finish.

   The installation is complete.

   Next, if necessary, configure multiple WebPass instances for Connector for WebSphere.

Configuring Multiple WebPass Instances for the Connector

Oracle Access Manager uses failover to maximize performance and provide uninterrupted service to end users. Failover redirects requests when a server fails. You may want to configure multiple WebPass instances for failover purposes.

This section assumes that you have already installed more than one instance of WebPass. See the *Oracle Access Manager Deployment Guide* for more information on failover.

To configure multiple WebPass instances for the Connector for WebSphere

1. Open the NetPointWASRegistry.properties file:

   CWS_install_dir/oblix/config/NetPointWASRegistry.properties

   where *CWS_install_dir* is the directory where you installed the Connector for WebSphere.

2. Enter the fully qualified host name with the domain name and port number for the WebPass using a comma-separated list as follows:

   ```
   # IdentitySystem WebPass webserver host name and port number
   OB_WebPassHost=foo.domain.com,bar.doman.com
   OB_WebPassPort=81,80
   ```

   In this example, the valid WebPass *host*:*port* combinations are:

   - foo.domain.com:81
   - bar.domain.com:80

3. Complete Connector setup as described next.

Completing Connector Setup

After installing the Connector for WebSphere, you must provide information about the Oracle Access Manager components that it communicates with, including the WebPass and Access Server.

Task overview: Completing Connector Setup

1. Complete your setup of the Connector, as described in "Setting Up the Connector for WebSphere" on page 11-26.
2. Test the Connector environment, as described in "Testing Environment Setup" on page 11-28.

Setting Up the Connector for WebSphere

In the following procedure, you ensure that the jar files added during installation appear in the proper location, or you add their location to the WebSphere classpath. You also ensure the environment variable path is correct. This is required because at run time, NetPointWASRegistry looks for the obaccess.dll file (Windows) or the libobaccess.so file (UNIX) that is located in the Oracle Access Manager installation directory.

To set up the Connector for WebSphere

1. Ensure that the following jar files that you added during installation exist in the directory *WAS_install_dir*/classes or add the location of these jar files to the WebSphere classpath:
   - NetPointWASRegistry.jar
   - jobaccess.jar
2. Add *CWS_install_dir*\oblix\lib to the environment variable path, as follows:

   Solaris: In the setupCmdLine.sh file, add as follows:

   CWS_install_dir\oblix\lib to $LD_LIBRARY_PATH

   where *CWS_install_dir* is the directory where you installed the Connector for WebSphere.

   Note: The NetPointWASRegistry may fail to install if it can not find the Access Manager SDK. You may need to add the following environment variables to the setupCmdLine.sh: *OBACCESS_INSTALL_DIR=CWS_install_dir*.

   AIX: In setupCmdLine.sh file, add as follows:

   CWS_install_dir\oblix\lib to $LIBPATH

   Restart the WebSphere Administration Server.

   Windows 2000: The installer automatically adds the information. However, you can:

   a. Manually add *CWS_install_dir*\oblix\lib to the PATH System variable.
   b. Reboot the machine.
   c. Start the WebSphere Administration Server.

3. Check the configuration to ensure that WebGate is protecting the WebPass.

   From the Access System Console, click Access System Configuration, click AccessGate Configuration in the left navigation pane, click the link for the WebGate that protects the WebPass, and in the IPValidation field select the Off option.

   As of Oracle Access Manager 10.1.4, the WebGateStatic.lst file no longer exists. The options in this file have moved to the Access System Console. See *Oracle Access Manager Access System Administration Guide* for details.

4. Restart the Web server.

5. Verify the information in the NetPointWASRegistry.properties file located *CWS_install_dir* \oblix\config.

   where *CWS_install_dir* is the directory where your Connector for WebSphere is installed. The file is populated with information that was specified during installation. For more information, see "NetPointWASRegistry.properties" on page 11-71.

6. Determine if the machine hosting WebPass is running SSL, and if so, complete the following steps:

   a. Open the NetPointWASRegistry.properties file and set OB_WebPassSSLEnabled = True.

   b. Obtain the Web server and CA certificates of the Web server hosting WebPass or WebGate running in SSL mode and place them respectively in the server.cer file and the ca.cer file.

   c. Use keytool in JAVA_HOME\bin or JAVA_HOME\jre\bin to add the following ca and server certificates to jssecacert keystore:

      * keytool -import -alias ca -file ca.cer -keystore jssecacert

      * keytool -import -alias server -file server.cer -keystore jssecacert

   d. Depending on the Java version that you are using, copy this file to the security directory located in JAVA_HOME\lib\security, or in JAVA_HOME\jre\lib\security.

The Connector for WebSphere uses WebPass to make IdentityXML calls. You can specify only one WebPass at a time for the Connector for WebSphere. Typically, this will be on the same Web server that hosts the Policy Manager. If you have more than one WebPass, and want the Connector for WebSphere configured for a different WebPass, you can change the host machine and port in NetPointWASRegistry.properties file after installation.

If you edit the NetPointWASRegistry.properties file, you must restart the WebSphere Administration Server.

Note: If WebPass is protected by a WebGate, ensure that the security level in this authentication scheme is the same level or a lower level than the one specified in the WebSphere authentication scheme discussed in "Defining an Authentication Scheme for WebSphere" on page 11-17.

Testing Environment Setup

Before you enable the NetPointWASRegistry, you need to run the registryTester program to ensure that the NetPointWASRegistry is registered and can successfully connect to the Identity System.

The following procedure applies to all WAS versions.

Note: On Linux systems, ensure that the LD_ASSUME_KERNEL environment variable is set to 2.4.1.9, as follows:

```
LD_ASSUME_KERNEL=2.4.19
export LD_ASSUME_KERNEL
```

To run the registryTester program

1. Edit the file registerTester.bat (Windows) or registryTester.sh (UNIX) in the *CWS_install_dir*/unsupported directory.

   where *CWS_install_dir* is the directory where you installed the Connector for WebSphere.

2. Modify these two variables as follows:
   - ***INSTALL_DIR***: Specify the path to the Connector for WebSphere.
   - ***WAS_INSTALL_DIR***: Specify the path to the WAS.
3. Specify the correct classpath and comment out the unused classpath for your installation:
   - **WebSphere 5.0 with Connector 7.0**: Keep WebSphere classpath for v5.0; comment out irrelevant WebSphere classpaths.
   - **WebSphere 5.0.2**: Keep WebSphere classpath for v5.0.2; comment out irrelevant WebSphere classpaths.
   - **WebSphere 5.1 with Connector 7.0.2**: Keep the WebSphere classpath for v5.1; comment out irrelevant WebSphere classpaths.
   - **WebSphere 6.0 with Connector 7.0.4**: Keep the WebSphere classpath for v5.1; comment out irrelevant WebSphere classpaths.
4. If you do not have a JAVA_HOME environment variable defined, set the JAVA_HOME parameter value as follows in the registryTester.bat or registryTester.sh file:

   Windows: %JAVA_HOME% = *WAS_install_dir* \java

   UNIX: $JAVA_HOME$ = *WAS_install_dir* /java

   where *WAS_install_dir* is the directory where you installed WebSphere.

5. **WebSphere 5.x only**: Update the registryTester.sh (.bat on Windows) as follows:

```
set CLASSPATH=.:${CLASSPATH}:${INSTALL_DIR}/oblix/lib/NetPointWASRegistry.jar
:${INSTALL_DIR}/oblix/lib/jobaccess.jar
:${WAS_INSTALL_DIR}/lib/xerces.jar
:${WAS_INSTALL_DIR}/lib/j2ee.jar
:${WAS_INSTALL_DIR}/lib/wssec.jar
:${WAS_INSTALL_DIR}/java/jre/lib/ext/ibmjsse.jar
```

Note: You may check the registryTester.bat for details.

---

6. From the command line, run registryTester:
   - **NT/W2K**: registryTester.bat
   - **UNIX**: registryTester.sh
7. Supply a Oracle Access Manager user ID and password when prompted.
8. Verify the result:
   - If the program completes successfully, it connects to the Identity Server and returns a list of groups to which the user belongs.
   - If the registryTester program fails to connect with the Access Server or the Identity Server, check the parameter values in the NetPointWASRegistry.properties file and correct them as needed.

Configuring WebSphere Application Server v5

The following sections describe the integration of Oracle Access Manager with the WebSphere Application Server v5.0 or 5.1.

Before you begin, see "Supported Versions and Platforms" on page 11-10 for complete details about version support.

Task overview: Integrating with WAS v5

1. Enable the registry, as described in "Enabling the NetPointWASRegistry in WAS v5" on page 11-29.
2. Test the configuration, as described in "Testing the NetPointWASRegistry for WebSphere v5" on page 11-32.
3. Configure the TAI, as described in "Configuring the TAI for WebSphere v5" on page 11-32.

Enabling the NetPointWASRegistry in WAS v5

Once you have installed the products and tested them to be sure they are communicating, you can enable the NetPointWASRegistry. Enabling the registry causes Oracle Access Manager to be used as the authentication source for the WebSphere Application Server.

The NetPointWASRegistry works with the User Registry in WebSphere 5.

Note: In the following procedure, any Identity System user who is added to the role "Admin" must be in a group called "Admin" or "Administrator". Otherwise, this user may not able to login to WebSphere Administrative Console. Other roles, like "monitor" have no restrictions. Any Identity System user added to these roles in the WAS 5 Administrative Console can log in to the WAS 5 Administrative Console.

To enable the NetPointWASRegistry in WAS 5

1. On Linux systems, ensure that the LD_ASSUME_KERNEL environment variable is set to 2.4.19 in the following location:

   WAS_install_dir/bin/setupCmdLine.sh

   Use the following command to set the environment variable:

```
LD_ASSUME_KERNEL=2.4.19
export LD_ASSUME_KERNEL
```

2. Ensure that WebGate protecting the WebPass has IPValidation set to Off.

   From the Access System Console, click Access System Configuration, click AccessGate Configuration in the left navigation pane, click the link for the WebGate that protects the WebPass, and in the IPValidation field select the Off option.

3. Create a backup copy of security.xml.

   In the following step, you will modify the configuration. You need to create a backup copy of security.xml before making a change to the configuration. If there are errors in the new configuration, you can always restore the previous version of the security.xml file.

4. Copy the security.xml file located in the following directory:

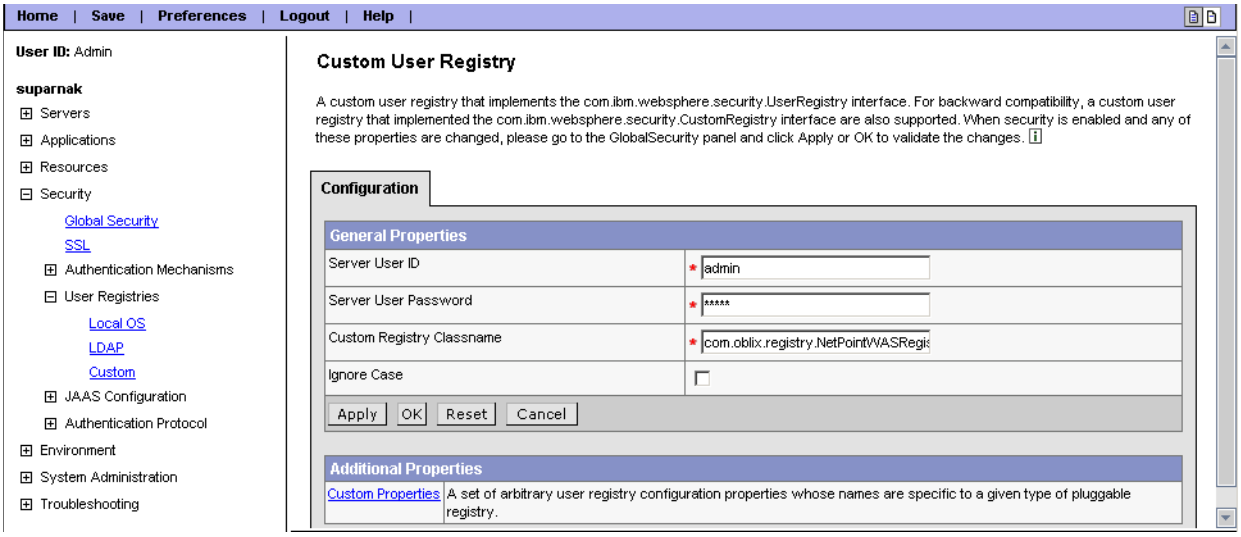
   WAS_install_dir/config/cells/serverName

5. Start the WebSphere Admin Service.

6. Log in as the Identity Administrator into the WebSphere Administrative Console.

7. Navigate to Security, User Registries, Custom Properties and enter the following:

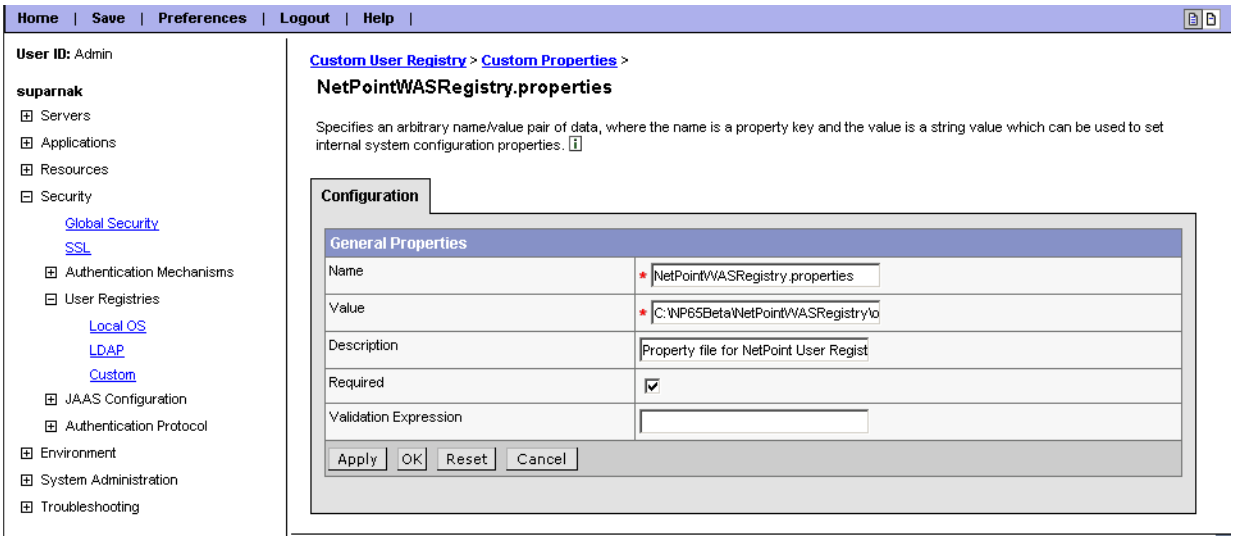
   - Identity Administrator user ID
   - Server User Password
   - CustomRegistry Classname
   - Identity System Admin ID
   - User's password
   - Oracle Access Manager Classname: com.oblix.registry.NetPointWASRegistry



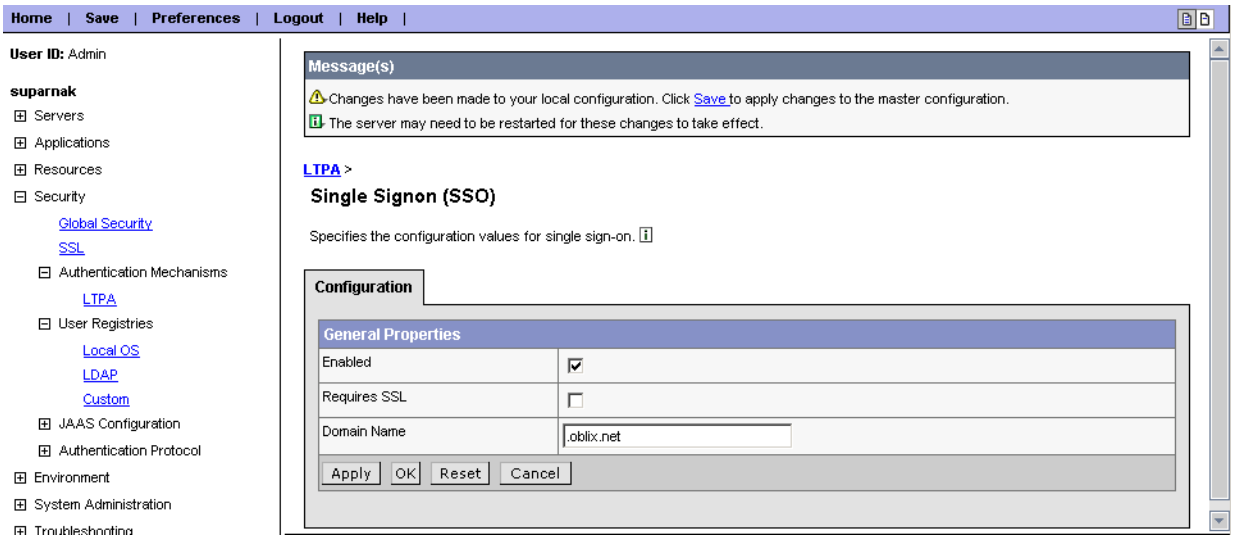
8. Under Additional Properties in the Configuration tab, click Custom Properties, click New, and enter the following:

   - **Name:** NetPointWASRegistry.properties
   - **Value:** C:*CWS_install_dir*\oblix\config\NetPointWASRegistry.properties.

- **Description:** Property file for Oracle Access Manager User Registry.
- Select the Required checkbox, click OK, and save your changes.



9. In the navigation pane on the left, click Authentication Mechanism, LTPA.
10. In the Configuration tab, specify a password.
11. Click OK.
12. Navigate back to LTPA (in the navigation pane on the left, click Authentication Mechanism, LTPA) and do the following:
    - Click Single Signon (SSO).
    - Click the Enabled box.
    - Enter the domain name. For example, .oracle.com.
    - Click OK.



13. In the navigation pane on the left, click Security, Global Security and change the following:

a. Set the Active Authentication Mechanism to LTPA.

b. Set the Active User Registry to Custom.

c. Click OK.

d. Click the Enabled box.

e. Click Apply to test the configuration.

   If the information is correct, a message confirming the changes is displayed at the top. Correct any errors that are displayed.

f. Click Save.

g. Click Logout and close the browser window.

h. Stop the WebSphere Application Service.

   If you get a message stating that the Service could not be stopped, switch to the Task Manager and ensure that there are no java processes running.

14. Start the WebSphere Admin Service.

Testing the NetPointWASRegistry for WebSphere v5

Next, verify the NetPointWASRegistry is configured correctly.

To test the NetPointWASRegistry configuration

1. Access the Snoop Servlet sample running on the default server at

   `http://hostname:port/snoop`

   or

   `http://hostname:web_server_plug-in_port/snoop`

2. When challenged by WebSphere, enter a username and password that is valid in Oracle Access Manager.

   By default, any authenticated user should be allowed access.

3. Launch the WebSphere Administrative Console and login as the user specified in Security, User Registries, Custom Properties.

   If the configuration is correct, you will be able to login successfully.

4. Set access control for the WebSphere Administrative Console by specifying the users and groups and the roles to which they belong.

   See WebSphere 5 documentation for more information.

5. Note the .xml files that are modified to support Admin Console access and click Save.

After you have installed Connector for WebSphere, you must configure the NetPointWASRegistry and TAI. The NetPointWASRegistry is used for authentication and the TAI enables single sign-on using Oracle Access Manager.

Configuring the TAI for WebSphere v5

You configure the TAI to enable single sign-on between Oracle Access Manager and WAS, as well as between Oracle Access Manager and the WebSphere Portal Server. For WebSphere 5.0 or 5.1, you must install webgate.properties, add the TAI, and then add custom properties.

Note: On Linux systems, ensure that the LD_ASSUME_KERNEL environment variable is set to 2.4.19 in the corresponding WebServer startup script, as follows:

```
LD_ASSUME_KERNEL=2.4.19
export LD_ASSUME_KERNEL
```

Tip: For optional details, see "Implementation Notes for the TAI" on page 11-77.

To install and configure TAI for WAS 5

1. Copy the configuration file named webgate.properties (see Table 11–2 for parameters):

   From: *CWS_install_dir* /oblix/config

   To: *WAS_install_dir* /properties folder in the WebSphere installation properties directory.

   where *CWS_install_dir* is the directory where the Connector for WebSphere is installed, and *WAS_install_dir* is the directory where the WebSphere Application Server is installed.

   This file contains configuration information that WebSphere will use to connect to the AccessGate.

2. In the WebSphere installation properties directory, modify the parameter values of the webgate.properties file (See Table 11–2) as follows:

   OB_InstallDir = *CWS_install_dir*

   where *CWS_install_dir* is the directory where the Connector for WebSphere is installed. For example:

   ```
   C:\COREid\NetPointWASRegistry
   ```

   If WebGate is installed on a proxy server that is used as a front end server to direct all user requests to Web servers that interface with WebSphere Application Servers, then specify the following parameter values:

   - OB_IsProxyEnabled=true
   - OB_hostnames = *serverName*

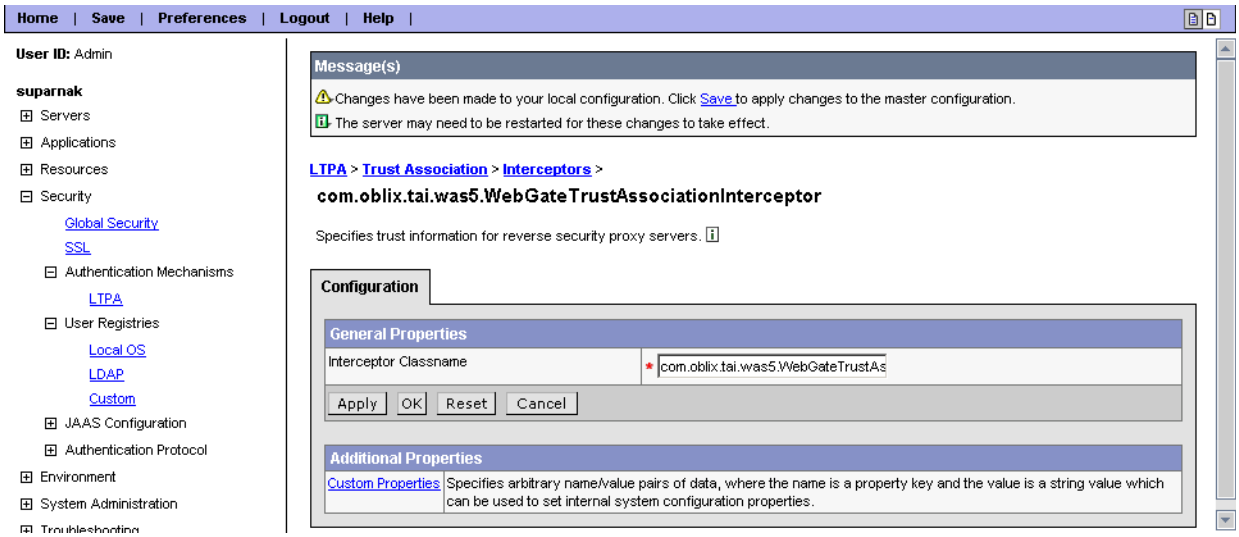
     where *serverName* is the name of the proxy server.
   - OB_ports = *portNumber*

     where *portNumber* is the port number of the proxy server.

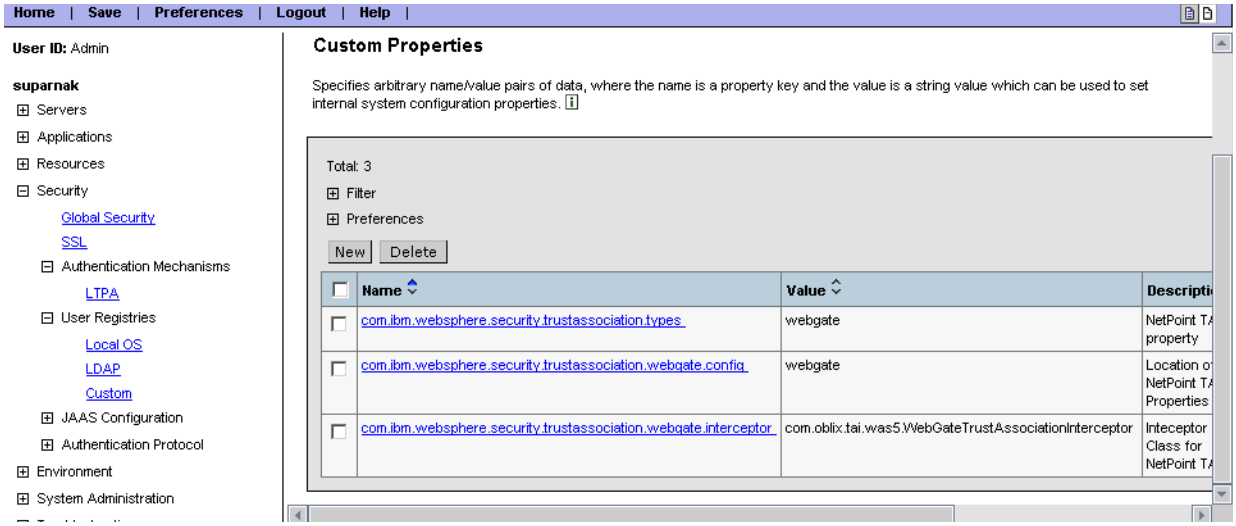
   Note: If you used a resource other than Authen, you must specify the resource name in the webgate.properties file.

3. Launch the WebSphere Administrative Console.
4. In the navigation pane on the left, click Authentication Mechanisms, LTPA.
5. Under Additional Properties, click Trust Association, Interceptors, New.
6. In the Name field, enter Oracle TAI Interceptor.

7. In the Interceptor classname field, enter com.oblix.tai.was5.WebGateTrustAssociationInterceptor.



8. Click OK.
9. Add the following three properties:
   a. **Name**: com.ibm.websphere.security.trustassociation.types
      * **Value**: webgate
      * **Description**: TAI property
      * Select the Required checkbox.
      * Click OK.
   b. **Name**: com.ibm.websphere.security.trustassociation.webgate.config
      * **Value**: webgate
      * **Description**: Name of the TAI properties file located in *WAS_install_dir*/properties directory.
      * Select the Required checkbox.
      * Click OK.
   c. **Name**: com.ibm.websphere.security.trustassociation.webgate.interceptor
      * **Value**: com.oblix.tai.was5.WebGateTrustAssociationInterceptor
      * **Description**: TAI class for WebSphere 5
      * Select the Required checkbox.
      * Click OK.



10. Click Interceptors at the top of the page and then click Save.
11. Navigate to:

    WAS_install_dir /config/cells/serverName_dir

    where *serverName_dir* is the directory where the server is installed.
12. Make a backup copy of the security.xml file.
13. In the WebSphere Administrative Console, navigate to LTPA, Trust Association, Interceptors.
14. Select the WebSeal Interceptor and click Delete.
15. Click Trust Association and click the Enabled check box.
16. Click Apply and then click Save.
17. Logout of the WebSphere Administrative Console and close the browser.
18. Shut down the Websphere Admin Server.

    If you get an error message, go to Task Manager and ensure that the java process is not running.
19. Restart the WebSphere Admin Service.

    If the Service does not start, verify that the properties are set correctly in the security.xml file and that the webgate.properties file is in the correct location.
20. Create an Access System policy to protect WebSphere resources

    In the Policy Manager, define a policy for the resource that you want to protect.

    Other authorization rules can also be added at this point. The policy that you use to protect the URL can use basic, form, cert, or other authentication schemes that the Access System supports.

    Ensure that the security level in this authentication scheme is *equal* or *greater* than the one specified in the WebSphere authentication scheme discussed in "Preparing Your Environment" on page 11-11.

    See the *Oracle Access Manager Access System Administration Guide* for more information on defining policies.
21. Verify that the TAI is working as detailed in the following section.

Testing the TAI for WAS v5

After you have configured TAI, restart WAS and test for successful authentication and single sign-on between WebSphere and Oracle Access Manager.

To conduct these tests, you use the Snoop servlet that WebSphere provides. The Snoop servlet has security constraints that only allow access to authenticated users. When WebSphere security is not enabled, access to the Snoop is unrestricted. When WebSphere security *and* TAI are enabled, users attempting to access Snoop will be challenged by the Access System. If TAI is *not* enabled, users attempting to access Snoop will be challenged by WebSphere.

To test Access System single sign-on for WAS, you must build and configure a new WebSphere secure application. Then, test Access System authentication and single sign-on for the secured application.

During installation, a secure application is built that you can use for testing and stored in the following location:

CWS_install_dir/examples/securityapp/SimpleSessionSecure.ear

where *CWS_install_dir* is the directory where you installed the Connector.

Note: If you wish, you can build your own secure application. The following procedure describes how to build an application for WAS 5.x.

To build a WebSphere secure application

1. Build a secure application named SimpleSessionSecure according to the instructions available at the following URL:

   `http://www-3.ibm.com/software/webservers/appserv/doc/v40/ae/infocenter/was/060704_security.html`

2. Save the SimpleSessionSecure.ear file in the appropriate location; for example, in c:\temp.

3. Verify that the WAS Administration Server is running.

To install the SimpleSessionSecure application

1. Launch the WebSphere Administrative Console located in *WAS_install_dir* \bin\adminclient.

   where *WAS_install_dir* is the directory where the WebSphere Application Server is installed.

2. In the WebSphere Console tree view, right-click WebSphere Administrative Domain, Enterprise Applications.

3. From the resulting menu, click Install Enterprise Application to launch the Install Enterprise Application wizard.

   The Specifying the Application or Module panel appears.

4. Verify the following settings:

   - The Browse for file on node field is set to your current node.
   - The Install Application option is selected.

5. Click Browse to locate and select the SimpleSessionSecure.ear file.

6. Verify that its name is now displayed in the Path field and specify SimpleSessionSecure as the Application name.
7. Click Next, then click Yes when prompted whether to deny access to unprotected methods.
8. In the Mapping Users to Roles panel, verify that the Goodguys role is mapped to valid Identity System Users.
9. Click Select; verify that Identity System Users are listed in the Selected Users/Groups area of the resulting Select Users/Groups dialog, then click OK to close the dialog after verification.
10. Click Next.
11. On the Mapping EJB RunAs Roles to Users panel, click Next.
12. In the Binding Enterprise Beans to JNDI Names panel, verify that the JNDI Name is set to gs/hello, and then click Next.
13. In the Mapping EJB References to Enterprise Beans panel, verify that the JNDI Name is set to gs/hello, and then click Next.
14. Click Next in the next three panels, until the Selecting Virtual Hosts for Web Modules panel appears.
15. In the Selecting Virtual Hosts for Web Modules panel, ensure that the Virtual Host is set to default_host, then click Next.
16. In the Selecting Application Server panel, ensure that the EJB11 and SimpleSessionWar modules reside on Application Server named Default Server and then click Next.
17. In the Completing the Application Installation Wizard panel, click Finish.
18. When prompted to regenerate code, click No.
19. Look for the message confirming successful installation of the application

    It may be a minute before it is displayed. You can now view the SimpleSessionSecure application in the WebSphere Administrative Console tree view.
20. After you build the SimpleSessionSecure application, regenerate the plug-in configuration to enable the Web server to locate the WebSphere application.

To regenerate the plug-in configuration

1. In the console tree view, right-click WebSphere Administrative Domain, Nodes, *hostname*.

   Where *hostname* is the name of the machine where WebSphere is installed.
2. From the resulting menu, select Regen Webserver Plugin.
3. In the Event Message panel, a message appears stating that the plug-in regeneration has been completed.
4. Stop the WebSphere Administrative Server and start it again:

   To stop the administrative server, under Nodes in the WebSphere Administrative Console, right-click *hostname* and select Restart from the resulting menu. The console will close.
5. Open the WebSphere Administrative Console again after the administrative server starts.

This time, you will be asked to log in, because security is enabled.

6. In the WebSphere Administrative Console tree view, click WebSphere Administrative Domain, Nodes, *hostname,* Application Servers, Default Server.

   Where *hostname* is the fully qualified name of the machine where WebSphere is installed; for example, xyz.domain.com.

7. Ensure that the Module Visibility setting of the Default Server is set to Compatibility.

8. If you want to change the visibility setting, click Apply.

To test Access System authentication and single sign-on

1. Access the SimpleSessionSecure application at the following URL:

   `http://hostname/gettingstarted3/SimpleSession?msg=Hi`

   where *hostname* is the fully qualified name of the machine where WebSphere is installed; for example, xyz.domain.com.

2. Log in as a Oracle Access Manager user.

   - **TAI Not Enabled**: If you have not enabled the TAI, you will be challenged by WebSphere and your credentials are passed on to the Access System. After the Access System authenticates you, you will be allowed to access SimpleSessionSecure.
   - **TAI Enabled**: If you *have* enabled TAI, you will be challenged and authenticated by the Access System. Because single sign-on between the Access System and WAS is enabled, you are allowed to access SimpleSessionSecure and other Access System-protected resources (URLs) without being challenged by WebSphere.

To test single sign-on for Access System-protected WebSphere resources

1. On the Web server you use to access the WAS, navigate to the document root and create a directory named test.

2. In the test directory, create a file named index.html.

3. In the Access System, create and enable policies to protect /servlet and /test directories.

4. Access the Snoop servlet at the following URL:

   `http://hostname.domain.com:9080/servlet/snoop`

   where *hostname* is the fully qualified name of the machine where WebSphere is installed; for example, xyz.domain.com.

   You will be challenged for authentication. After you are authenticated, you will be allowed to access the Snoop servlet.

5. Access the /test URL.

   You must be allowed to access the URL and view the index.html file without being challenged.

Enabling Logging for TAI for WAS v5

You can enable logging for the TAI from the WebSphere Administrative Console. The procedure to enable logging varies depending on the WAS version that you use.

To enable logging for TAI for WAS 5

1. Launch the WebSphere Administrative Console.
2. Navigate to Troubleshooting, Logs and Trace
3. Select your Server.
4. Select Diagnostic Trace.
5. Modify the Trace specification.
6. Select the Components tab.
7. Enable debug logging for com.oblix.tai.was5.WebGateTrustAssociationInterceptor.
8. Integrate Oracle Access Manager with the WAS Portal v5, if desired.

Configuring the WebSphere Application Server v6

The following sections describe how you integrate Oracle Access Manager with the WebSphere Application Server v6:

- Enabling the NetPointWASRegistry for WAS 6 and 6.1
- Testing the NetPointWASRegistry for WebSphere v6
- Configuring the TAI for WebSphere 6 and 6.1
- Testing the TAI for WAS 6 and 6.1
- Enabling Logging for TAI for WAS 6 and 6.1

Supported Versions and Platforms

You can find support and certification information at the following URL:

`http://www.oracle.com/technology/documentation/`

You must register with OTN to view this information.

Also, you can see the supported versions and platforms for this integration on Metalink, as follows.

To view information on Metalink

1. In your browser, enter the following URL:

   `https://metalink.oracle.com`
2. Log in to MetaLink.
3. Click the **Certify** tab.
4. Click **View Certifications by Product**.
5. Select the **Application Server** option and click **Submit**.
6. Choose **Oracle Identity Manager** and click **Submit**.
7. Click **Oracle Identity Management Certification Information 10g (10.1.4.0.1) (html)** to display the Oracle Identity Management page.
8. Click the link for **Section 6, Oracle Access Manager Certification** to display the certification matrix.

Enabling the NetPointWASRegistry for WAS 6 and 6.1

After you have installed WAS 6 or 6.1 and the Connector for Websphere, then tested them to be sure they are communicating, you can enable the NetPointWASRegistry, which interoperates with the User Registry in WebSphere 6 and 6.1. This enables Oracle Access Manager to act as the authentication source for the WebSphere Application Server.

Notes: The procedures for WAS 6 and 6.1 are slightly different.

In the following procedures, any Oracle Access Manager user who is added to the role "Admin" must belong to the group named "Admin" or "Administrator." Otherwise, this user may not able to log in to WebSphere Administrative Console. Other roles, such as "Monitor" have no restrictions. Any Oracle Access Manager user added to these roles through the WAS 6 or 6.1 Administrative Console can log in to the WAS 6 or 6.1 Administrative Console.

To enable the NetPointWASRegistry in WAS 6

1. Ensure that WebGate protecting the WebPass has IPValidation set to Off.

   From the Access System Console, click Access System Configuration, click AccessGate Configuration in the left navigation pane, click the link for the WebGate that protects the WebPass, and in the IPValidation field select the Off option.

2. Make a backup copy of the *security.xml* file located in the following directory:

   WPS_install_dir/profiles/default/config/cells/*serverNodeName*

   Create a backup copy of security.xml whenever you make a change to the configuration. If you make errors in the new configuration, you can restore from backup.

3. Start the WebSphere Admin Service.

4. Log in to the WebSphere Administration Console as the WebSphere administrator.

   The Administrative Console for WAS 6 is accessible through the following URL:

   http://*hostname.domain*.com:9060/ibm/console

   where *hostname* is the fully qualified name of the machine where WebSphere is installed; for example, xyz.domain.com.

5. In the left navigation pane, click Security, then click Global Security, and do the following:

   - Uncheck the Enable Global Security checkbox, click Apply, click Save, then log out an close the browser window.
   - Stop the WebSphere Application Service.

     If a message announces that Service could not be stopped, switch to the Task Manager and ensure that no Java processes are currently running.
   - Restart WebSphere and log in to the WebSphere Administration Console as the Identity administrator.

6. Navigate to Security, Global Security, Custom and enter values for the following variables:

   - **Server user ID**

- **Server User Password**
- **CustomRegistry Classname**
- **Oracle Access Manager Admin ID**
- **User password**
- **Oracle Access Manager Classname**: com.oblix.registry.NetPointWASRegistry

7. Navigate to Configuration, Additional Properties, Custom Properties, New.

   Enter values for the following parameters:

   - **Name**: `NetPointWASRegistry.properties`
   - **Value**: /opt/netpoint/oblix/config/NetPointWASRegistry.properties
   - **Description**: Property file for User Registry

8. Click Apply.
9. Navigate to Global Security, Authentication Mechanisms, LTPA.
10. In the Configuration tab, specify a password, then click Apply.
11. Click Single Signon (SSO).
    - Click the Enabled box.
    - Enter the domain name (for example, oracle.com)., then click Apply.
12. In the left navigation pane, click Security, click Global Security, then complete the following steps:
    a. Set the Active Authentication Mechanism to LTPA.
    b. Set the Active User Registry to Custom User Registry, then click OK.
    c. Click the Enable Global Security box.
    d. Click Apply to test the configuration.

       If the information is valid, a message confirming the changes displays at the top. Otherwise, correct any errors that are reported.

    e. Click Save.
    f. Click Logout and close the browser window.
    g. Stop the WebSphere Application Service. If a message announces that Service could not be stopped, switch to the Task Manager and ensure that no Java processes are currently running.
13. Start the WebSphere Administration Service.

To enable the NetPointWASRegistry in WAS 6.1

1. Ensure that WebGate protecting the WebPass has IPValidation set to Off.

   From the Access System Console, click Access System Configuration, click AccessGate Configuration in the left navigation pane, click the link for the WebGate that protects the WebPass, and in the IPValidation field select the Off option.

2. Make a backup copy of the *security.xml* file located in the following directory:

   WPS_install_dir/profiles/default/config/cells/*serverNodeName*

Create a backup copy of security.xml whenever you make a change to the configuration. If you make errors in the new configuration, you can restore from backup.

3. Start the WebSphere Admin Service.
4. Log in to the WebSphere Administration Console as the Identity Administrator.

   The Administrative Console for WAS 6 is accessible through the following URL:

   http://*hostname.domain*.com:9060/ibm/console

   where *hostname* is the fully qualified name of the machine where WebSphere is installed; for example, xyz.domain.com.
5. Navigate to Security, then to Secure administration, applications, and infrastructure, and in the Available realm definitions section select Standalone Custom Registry, click Configure, and enter the following under General Properties.
   - **Primary administrative user name**: Enter *oam_administrator_ID*, where the value is an Oracle Access Manager administrator ID.
   - **Server user identity**: Select the Automatically generated user identity option.
   - **Custom registry class name**: `com.oblix.registry.NetPointWASRegistry`
   - Check the Ignore Case for Authorization checkbox.
6. Click Apply.
7. Navigate to Additional Properties, Custom Properties, New, and enter values for the following parameters:
   - **Name**: NetPointWASRegistry.properties
   - **Value**: /opt/netpoint/oblix/config/NetPointWASRegistry.properties

     This is the full path to the NetPointWASRegistry.properties file.
   - **Description**: Property file for User Registry
8. Click Apply.
9. Navigate to Security, then to Secure Administration, applications, and infrastructure, then to Web Security.

   Click Single sign-on (SSO), and enter the following under General Properties:
   - Click the Enabled box.
   - Enter the domain name (for example, mycompany.com).

   Click Apply.
10. Go to Security, then to Secure Administration, applications, and infrastructure, and do the following:
    a. Under Administrative Security, click the Enabled administrative security box.
    b. Under Application security, click the Enable application security box.
    c. In the Available Realm Definition list, select Standalone Custom Registry and click the Set As Current button.
    d. Click Apply to test the configuration.

If the information is valid, a message confirming the changes displays at the top. Otherwise, correct any errors that are reported.

e. Click Save.

f. Click Logout and close the browser window.

g. Stop the WebSphere Application Service.

If a message announces that Service could not be stopped, switch to the Task Manager and ensure that no Java processes are currently running.

11. Start the WebSphere Administration Service.

Testing the NetPointWASRegistry for WebSphere v6

After you have enabled the NetPointWASRegistry, you need to verify that it is configured correctly.

To test the NetPointWASRegistry configuration

1. Access the Snoop Servlet sample running on the default server at the following URL:

   `http://`*`hostname`*`:9080/snoop`

   Where *hostname* is the fully qualified name of the machine where WebSphere is installed.

   Alternatively, you can use the following URL:

   `http://`*`hostname`*`:`*`web_server_plug-in_port`*`/snoop`

   Where *hostname* is the fully qualified domain name of the machine on which WebServer is installed.

2. When challenged by WebSphere, enter a username and password that is valid in Oracle Access Manager.

   By default, any authenticated user should be allowed access.

3. Launch the WebSphere Administrative Console and login as the user specified in Security, Global security, Custom.

   If the configuration is valid, the login will succeed.

4. Set access control for the WebSphere Administrative Console by specifying the users and groups and the roles to which they belong.

   See WebSphere 6 documentation for details.

5. Write down the names of the .xml files that you have modified to support Admin Console access, then click Save.

After you have installed Connector for WebSphere, you must configure the NetPointWASRegistry and TAI. The NetPointWASRegistry handles authentication, and the TAI facilitates Access System single sign-on.

Configuring the TAI for WebSphere 6 and 6.1

You configure the TAI to enable single sign-on between Oracle Access Manager and WAS. For WebSphere 6.0 or 6.1 you must install *webgate.properties*, add the TAI, and then add *custom.properties*.

Note: The procedures for configuring the TAI are slightly different for WAS 6 and 6.1.

To install and configure TAI for WebSphere v6

1. Copy the configuration file webgate.properties from the following directory:

   CWS_install_dir/oblix/config

   where *CWS_install_dir* is the directory where the Connector for WebSphere is installed, to the WebSphere installation properties directory, which resides in the following location:

   WAS_install_dir/properties

   where *WAS_install_dir* is the directory where the WebSphere Application Server is installed.

   Webgate.properties contains configuration information that WebSphere will use to connect to the AccessGate.

2. In the WebSphere installation properties directory, modify the parameter values in the webgate.properties file as follows:

   OB_InstallDir = *CWS_install_dir*

   where *CWS_install_dir* is the directory where the Connector for WebSphere is installed.

   If the associated WebGate is installed on a proxy server used as a front end server to direct all user requests to Web servers that interface with WebSphere Application Servers, then specify the following parameter values:

   - `OB_IsProxyEnabled=true`
   - `OB_hostnames = serverName`

     where *serverName* is the name of the proxy server.

   - `OB_ports = portNumber`

     where *portNumber* is the port number of the proxy server.

   Note: If you used a resource other than Authen, you must specify the resourcename in the `webgate.properties` file.

3. Launch the WebSphere Administrative Console.
4. In the navigation pane on the left, navigate to Global security, Select Authentication Mechanisms, LTPA, Trust Association, Additional Properties, Interceptors, New.
5. In the Interceptor classname field, enter `com.oblix.tai.was5.WebGateTrustAssociationInterceptor`, then click Apply.
6. Navigate to Additional Properties, then to Custom Properties, then to New.
7. Make the following changes:
   a. **Name**: com.ibm.websphere.security.trustassociation.types
      * **Value**: `webgate`

* **Description**: TAI property

Click OK.

b. **Name**: `com.ibm.websphere.security.trustassociation.webgate.config`

* **Value**: `webgate`

* **Description**: Name of the TAI properties file located in *WAS_install_dir*/properties directory.

Click OK.

c. **Name**: `com.ibm.websphere.security.trustassociation.webgate.interceptor`

* **Value**: `com.oblix.tai.was5.WebGateTrustAssociationInterceptor`

* **Description**: TAI class for WebSphere 6

Click OK.

8. Click Interceptors at the top of the page and then click Save.
9. Navigate to the following:

   WAS_install_dir/profiles/default/config/cells/serverNodeName
10. Make a backup copy of the `security.xml` file.
11. In the WebSphere Administrative Console, navigate to LTPA, Trust Association, Interceptors.
12. Select the WebSeal Interceptor, then click Delete.
13. Click Trust Association, then click the Enable Trust Association check box.
14. Click Apply, then click Save.
15. Logout of the WebSphere Administrative Console, then close the browser.
16. Shut down the Websphere Admin Server.

    If you get an error message, go to Task Manager and ensure that no Java process is running.
17. Restart the WebSphere Admin Service.

    If the Service does not start, verify that the properties are set correctly in the `security.xml` file and that the `webgate.properties` file is in the correct location.
18. Create an Access System policy to protect WebSphere resources as described in "To install and configure TAI for WAS 5" on page 11-33.

    To facilitate access the Administration Console after you have enabled TAI, you need to enable the Policy created in "Defining a Policy Domain for WebSphere" on page 11-17. You also need to enable the policy created in "Defining a Policy Domain for the WebSphere v6.0 Administration Console" on page 11-19.
19. Verify that the TAI is working, as detailed in "Testing the TAI for WAS 6 and 6.1" on page 11-48.

To install and configure TAI for WAS 6.1

1. Copy the configuration file webgate.properties from the following directory:

CWS_install_dir/oblix/config

where *CWS_install_dir* is the directory where the Connector for WebSphere is installed, to the WebSphere installation properties directory, which resides in the following location:

WAS_install_dir/properties

where *WAS_install_dir* is the directory where the WebSphere Application Server is installed.

Webgate.properties contains configuration information that WebSphere will use to connect to the AccessGate.

2. In the WebSphere installation properties directory, modify the parameter values in the webgate.properties file as follows:

   OB_InstallDir = *CWS_install_dir*

   where *CWS_install_dir* is the directory where the Connector for WebSphere is installed.

   If the associated WebGate is installed on a proxy server used as a front end server to direct all user requests to Web servers that interface with WebSphere Application Servers, then specify the following parameter values:

   - `OB_IsProxyEnabled=true`
   - `OB_hostnames = serverName`

     where *serverName* is the name of the proxy server.
   - `OB_ports = portNumber`

     where *portNumber* is the port number of the proxy server.

   Note: If you used a resource other than Authen, you must specify the resourcename in the `webgate.properties` file.

3. Launch the WebSphere Administrative Console.
4. Go to Security, then to Secure Administration, applications, and infrastructure, then to Web Security.

   Click Trust Association, then click Additional Properties, then click Interceptors, then click New.
5. In the Interceptor classname field, enter `com.oblix.tai.was5.WebGateTrustAssociationInterceptor`, then click OK.
6. Click `com.oblix.tai.was5.WebGateTrustAssociationInterceptor`, then click Additional Properties – Custom Properties, then click New.
7. Add the following custom properties:
   a. **Name**: com.ibm.websphere.security.trustassociation.types
      * **Value**: `webgate`
      * **Description**: TAI property

      Click OK.

b. **Name**: `com.ibm.websphere.security.trustassociation.webgate.config`

   * **Value**: `webgate`
   * **Description**: Name of the TAI properties file located in *WAS_install_dir*/properties directory.

   Click OK.

c. **Name**: `com.ibm.websphere.security.trustassociation.webgate.interceptor`

   * **Value**: `com.oblix.tai.was5.WebGateTrustAssociationInterceptor`
   * **Description**: TAI class for WebSphere 6

   Click OK.

8. Click Interceptors at the top of the page and then click Save.
9. Navigate to the following:

   WAS_install_dir/profiles/default/config/cells/serverNodeName
10. Make a backup copy of the `security.xml` file.
11. Go to Security, then to Secure Administration, applications, and infrastructure, then to Web security.

    Click Trust Association, then click Additional Properties, then click Interceptors.
12. Delete the following Interceptors:
    - `com.ibm.ws.security.spnego.TrustAssociationInterceptorImpl`
    - `com.ibm.ws.security.web.TAMTrustAssociationInterceptorPlus`
    - `com.ibm.ws.security.web.WebSealTrustAssociationInterceptor`
    - `com.ibm.ws.sip.security.digest.DigestTAI`
13. Click Trust Association, then click the Enable Trust Association check box.
14. Click Apply, then click Save.
15. Logout of the WebSphere Administrative Console, then close the browser.
16. Shut down the Websphere Admin Server.

    If you get an error message, go to Task Manager and ensure that no Java process is running.
17. Restart the WebSphere Admin Service.

    If the Service does not start, verify that the properties are set correctly in the `security.xml` file and that the `webgate.properties` file is in the correct location.
18. Create an Access System policy to protect WebSphere resources as described in "To install and configure TAI for WAS 5" on page 11-33.

    To facilitate access the Administration Console after you have enabled TAI, you need to enable the Policy created in "Defining a Policy Domain for WebSphere" on page 11-17. You also need to enable the policy created in "Defining a Policy Domain for the WebSphere v6.0 Administration Console" on page 11-19.

19. Verify that the TAI is working, as detailed in "Testing the TAI for WAS 6 and 6.1" on page 11-48.

Testing the TAI for WAS 6 and 6.1

The following procedure describes testing the TAI for WAS 6 and 6.1.

To test the TAI

After you have configured TAI, test for successful authentication and single sign-on between WebSphere and Oracle Access Manager.

To conduct these tests, use the Snoop servlet that WebSphere provides. The Snoop servlet has security constraints that only allow access to authenticated users.

When WebSphere security is not enabled, access to the Snoop is unrestricted.

When WebSphere security and TAI are enabled, users attempting to access Snoop will be challenged by the Access System. If TAI is not enabled, users attempting to access Snoop will be challenged by WebSphere as well.

To test single sign-on for Access System-protected WebSphere resources

1. On the Web server you use to access the WAS, navigate to the document root and create a directory named test.
2. In the test directory, create a file named index.html.
3. In the Access System, create and enable policies to protect the `/snoop` and `/test` directories.
4. Access the Snoop servlet through the following URL:

   `http://`*`hostname`*`.`*`domain`*`.com:80/snoop`

   where *hostname* is the fully qualified name of the machine where the Web server is installed.

   You will be challenged for authentication. After you are authenticated, you will be allowed to access the Snoop servlet.
5. Access the */test* URL.

   Verify that you can access the URL and view the index.html file without being challenged.

Enabling Logging for TAI for WAS 6 and 6.1

You can enable logging for TAI from the WebSphere Administrative Console.

To enable logging for TAI for WAS 6 and 6.1

1. Launch the WebSphere Administrative Console.
2. Navigate to Troubleshooting, Logs and Trace
3. Select your Server.
4. Select Change Log Level Details.
5. Select Components.
6. Enable debug logging for `com.oblix.tai.was5.WebGateTrustAssociationInterceptor`

Integrating with WebSphere Portal

A portal provides a single point of access to enterprise data and applications, presenting a unified and personalized view of that information to employees, customers, and business partners.

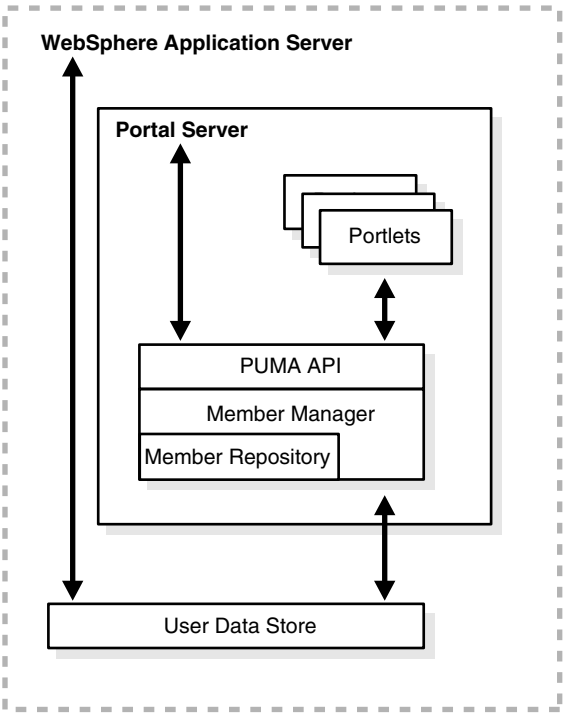
The WebSphere Portal Server runs on top of the WAS and uses the WAS security infrastructure to enforce access control. Integrating Oracle Access Manager with the WebSphere Portal provides the following Oracle Access Manager functionality for the portal:

- User and group management
- Password management
- SSO to the portal
- Unified logout between Oracle Access Manager, WAS, and the WebSphere Portal

The WebSphere Portal V5 uses the following component to manage user and group information.

Member Manager: Member Manager presents a Java object view of Users and Groups to WebSphere Portal, including all portlets installed on WebSphere Portal. Member Manager (as accessed through PUMA) is the abstraction interface that WebSphere Portal V5.0 uses to access user and group information. This includes the user accounts, which tell Portal that the user exists, any user groups within which the user might be a member, and attributes about the users.

The following figure shows the architecture of the Member Manager.



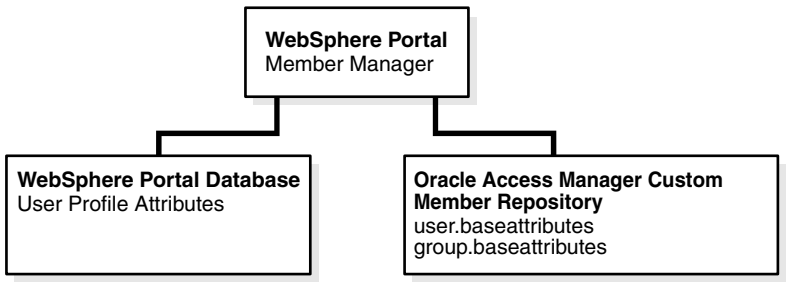
See the WebSphere Portal v5 documentation for more information on the portal and related components.

Oracle Access Manager Custom Member Repository: The Custom Member Repository (CMR) is available with the Connector for WebSphere, as described in "Supported Versions and Platforms" on page 11-10. The CMR is an instance of a

Member Manager component that connects the WebSphere Portal Server to the Identity System users and groups.

The CMR is a custom user data store that implements the IBM WebSphere MemberRepository interface. As shown in Figure 11–4, the CMR stores user and group base attributes in user data store. The CMR is used by the WebSphere Portal Server to make queries like getAttributes for a user for personalization, getGroupMemberships, search users by attribute, and similar functions. User profile information is in the Portal database and authentication information is available through the Administration Console and LDAP.

Figure 11–4 Member Manager, WebSphere Portal Database, and the CMR



Note: Group MemberShip functionality is not currently supported. As a result, with Nested Groups, if you check for inner group membership it won't display its parent group details.

The CMR supports only read operations, not create, modify, or delete operations. The CMR is an extension of the custom user registry (CUR) and requires the Portal Server.

The configuration files used to control WebSphere Portal Member Manager come into play with the CMR are explained in the following paragraphs. These files are usually created during the portal installation:

wmm.xml: Top level Member Manager configuration. Lists and configures the various MemberRepository implementations used by Member Manager. Most other Member Manager configuration files are pointed to from this file. The CMR details should be configured in this file.

PumaService.properties: This is the configuration file for the PUMA API, which in WebSphere Portal V5.0.x is a mapping layer between WebSphere Portal and Member Manager. This is not a Member Manager configuration file, but because PUMA is part of the "user stack" in Portal, this configuration file is important.

This file includes a comma-separated list of attribute names that will be passed to Member Manager requests and several multi-valued properties. This file may need to be configured for the user attributes for personalization. All user.base.attributes and group.base.attributes values will be searchable in the CMR. For example:

```
user.base.attributes=cn,uid,cn,logonId,logonPasswordVerify,logonPassword ...
 group.base.attributes=cn,uniqueOwnerIdentifier,membergroups,
 groupmembers,memberGroupName,memberGroupType,distinguishedName
```

All other attributes go to the Portal database.

During startup, only the attributes identified in the user.minimum.attributes parameter are retrieved. For example:

```
user.minimum.attributes=cn,genUserid,cn,givenName,sn,mail
```

Note: All attributes in the user.minimum.attributes list must have correct attribute access control set in the User Manager and Group Manager for the Administrator, and all need to be in one of the User Manager configuration panels. For example, if the Portal Server needs the givenName attribute, one of the Identity System panels needs First Name. In the Identity System, the givenName attribute is mapped to Display Name, First Name.

To see the LDAP to Oracle Access Manager mapping, you can select the desired attribute and view the corresponding Display Name in the Identity System Console, User Manager Configuration, Configure Tab, Link, Modify Attributes page.

See the *Oracle Access Manager Identity and Common Administration Guide* for details.

wpconfig.properties: This is WebSphere Portal configuration file. The portal user/group administrator details needs to be set in this file. This file is present in *WPS_install_dir*/config folder, where *WPS_install_dir* is the Portal Server home directory.

VaultService.properties: This file is located in WPS_install_dir /shared/app/config/services folder. This configuration file is used to specify Vault Adapter Implementations. You have to set correct system admin credential DN in this file.

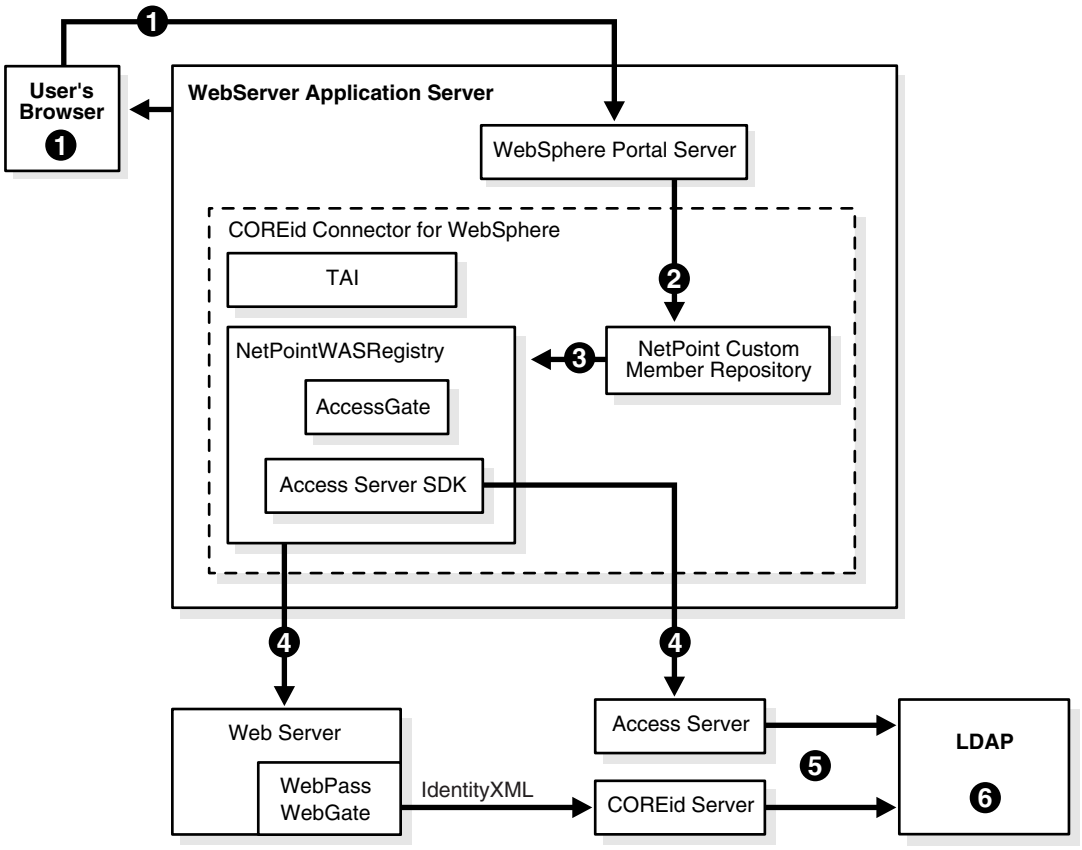
About Integration with the CMR

During login, the user is authenticated as depicted in "Scenario 1: Use of NetPointWASRegistry" on page 11-5.

Without the Oracle Access Manager CMR, the WebSphere Portal Server must communicate directly with the LDAP directory server to obtain user, group, and personalization information. With the CMR, communication between the WebSphere Portal Server and the directory server can be eliminated. The CMR performs read operations through the NetPointWASRegistry with the directory server.

Figure 11–5 shows the interaction between the WebSphere Portal Server, CMR, and LDAP directory server during the login authorization process. This follows processes described in "Scenario 1: Use of NetPointWASRegistry" on page 11-5.

Figure 11–5 WebSphere Portal Server and the Custom Member Repository



Process overview: Authorization with the CMR

For instance, the Access Manager SDK uses the checkPassword method while IdentityXML uses all other methods:

1. After authentication, a user requests access to a portlet through the WebSphere Portal Server.
2. The Portal Server forwards the request to the CMR.
3. The CMR forwards the request to the NetPointWASRegistry.
4. The NetPointWASRegistry sends an IdentityXML call to the Identity Server or uses the Access Manager SDK to contact the Access Server through WebPass or WebGate, depending upon the required method.
   - findByAttribute to search users by attribute
   - getMember
   - getGroupMemberIdentifiers
   - getMembers
   - search etc.
5. The Identity Server (or Access Server) communicates with the LDAP directory server.
6. The directory server returns information.

Setting up the WebSphere Portal v5.0.2 with Oracle Access Manager

Integrating the WebSphere Portal v5 with Oracle Access Manager involves a series of installation and configuration tasks.

To integrate the WebSphere Portal with Oracle Access Manager

1. Complete tasks in "Preparing to Install the Connector" on page 11-11:
   a. Install the WebSphere Application Server with appropriate Fix Pack as described in the WebSphere documentation to correct the following issues.

      For example:

      Issue: Access permissions to portlets do not work if the user dn contains intermediate spaces after a comma, for example:

      Cn=PortalUser, o=company, c=us

      IBM has released a Fix PQ93461 for this Portal User access permission problem.

      Action: This fix pack needs to be applied. Verify the Portal Server History Log to ensure that all required fixes have been installed. Refer to the Portal Infocenter document for details.

   b. Install the WebSphere Portal with WebSphere Application Server security disabled.

      Note: Both the Global and Java 2 security should be disabled while installing the Portal Server.

      See the IBM WebSphere Portal Infocenter document for details.

   c. Install and configure Oracle Access Manager, as discussed in "Preparing to Install the Connector" on page 11-11.
2. Install the Connector for WebSphere and configure the NetpointWASRegistry and TAI components, as described in "Installing the Connector for WebSphere" on page 11-20.
3. Complete Connector setup and testing, as discussed in "Completing Connector Setup" on page 11-26.
4. In the WebSphere Administration Console, change security to custom registry.

   This specifies the NetPointWASRegistry, which establishes a connection between the WAS and Oracle Access Manager. The WAS uses the NetPointWASRegistry to authenticate and authorize portal users with the Access System's security policies.
5. Ensure that the following Admin credentials are set in clear text in the NetPointWASRegistry.properties file:

   ```
   OB_AdminUserName=admin OB_AdminUserCreds=password
   ```

   Where the *OB_AdminUserName* value is the user ID of the Portal Server administrator who is a Master Administrator or a Master Identity Administrator.

   This is required for the CMR. The Admin credentials should be set in clear text. The NetPointWASRegistry reads the password, encrypts it, and rewrites the properties file with the encrypted password. The encryptor can be executed by running the registryTester program, as well as from WebSphere. To assist you with adding these parameters, see the NetPointWASRegistryProperties.sample file,

which includes comments. See also, "NetPointWASRegistry.properties" on page 11-71.

> **Note:** The formatting of NetPointWASRegistry.properties is lost when the Connector for WebSphere rewrites the file with the encrypted password. You may want to save a copy of the NetPointWASRegistry.properties.

6. Restart the WAS Server.
7. Ensure that the WebSphere Application server and Portal Server is stopped.
8. Make a back-up copy of the following file:

   WPS_install_dir/config/wpconfig.properties file

   Where *WPS_install_dir* is the directory where WebSphere Portal Server is installed.
9. Edit the wpconfig.properties file and add the following:

```
PortalAdminId (Example: PortalAdminId=DN of wpsadministrator)
PortalAdminIdShort (Example: PortalAdminIdShort=wpsadministrator)
PortalAdminPwd (Example: PortalAdminPwd=wpsadminpassword)
WasUserid (Example: WasUserid=wasadministrator)
WasPassword (Example: WasPassword=wasadminpassword)
PortalAdminGroupId (Example: PortalAdminGroupId=DN of PortalAdminGroupId)
PortalAdminGroupIdShort (Example: PortalAdminGroupIdShort=PortalAdminGroupId)
```

10. Restart the WebSphere Application server.
11. **Optional**: Turn PUMA traces on in Portal, which can be done by entering the following in the log.properties file.

    For example:

```
WPS_install_dir\shared\app\config\log.properties
```

```
traceString=com.ibm.wps.services.puma.*=all=enabled:com.ibm.wps.
puma.*=all=enabled:com.ibm.wps.command.puma.*=all=enable
```

12. Backup the following file:

    WPS_install_dir/shared/app/wmm/wmm.xml file

13. Edit the file to make the changes for CMR configuration as follows (and the lookaside flag should be set to "false").

    Note that the file named customRepositoryAttributes.xml is a dummy file and is not part of the WebSphere or Oracle Access Manager configuration.

```
<supportedMemberTypes>

<supportedMemberType name="Person"
 rdnAttrTypes="uid"
 defaultParentMember="o=company,c=us"
 defaultProfileRepository="CNR"/>
 <!-- o=company,c=us is Root DN. please enter the DN in your environment. -->

 <supportedMemberType name="Group"
 rdnAttrTypes="cn"
 defaultParentMember="o=company,c=us"
 defaultProfileRepository="CNR"/>
 <!-- Name of the CMR information tag -->
```

```
 </supportedMemberTypes>

 <profileRepository name="NetpointCustomRepository" UUID="CNR"
 description="This is Oracle WMM custom MemberRepository implementation."
 vendor="Oracle"
 adapterClassName="com.oblix.registry.NetPointMemberRepositoryImpl_v5"
 specVersion="1.0" adapterVersion="2.0"
 configurationFile="customRepositoryAttributes.xml"
 wmmGenerateExtId="false" supportGetPersonByAccountName="false"
 supportDynamicAttributes="false" profileRepositoryForGroups="CNR"
 enableTrace="true "
PumaService.properties="WPSInstallDir/shared/app/config/services/PumaService.pr
operties"
NetPointWASRegistry.properties="NetpointWASConnInstallDir\oblix\config\NetPoint
WASRegistry.properties">

<!-- 1. com.oblix.registry.NetPointMemberRepositoryImpl_v5 is name of the CMR
implementtion class
2. Mention path of the PUMA service against PumaService.properties paramter.
3. NetPointWASRegistry.properties - Path of the Oracle Access Manager WAS
registry configuration file.
For Unix this path should be mentioned as
"NetpointWASConnInstallDir/oblix/config/NetPointWASRegistry.properties
4. CustomRepositoryAttributes.xml is a dummy file.
 -->

 <readMemberType>

 <memberType name="Person" /> <memberType name="Group" />
 <!-- Only read access to portal is provided by COREid Connector CMR-->
 </readMemberType>
 <createMemberType>
 <!-- <memberType name="Person" /> <memberType name="Group" /> -->
 <!-- Commented out - can't create in the sample -->
 </createMemberType>
<updateMemberType>
 <!-- <memberType name="Person" /> <memberType name="Group" / > -->
 <!-- Commented out - can't update in the sample -->
 </updateMemberType>
 <deleteMemberType>
 <!-- <memberType name="Person" /> <memberType name="Group" /> -->
 <!-- Commented out - can't delete in the sample -->
 </deleteMemberType>
 <renameMemberType>
 <!-- <memberType name="Person" /> <memberType name="Group" /> -->
 <!-- Commented out - can't rename in the sample -->
 </renameMemberType>
 <moveMemberType>
 <!-- <memberType name="Person" /> <memberType name="Group" /> -->
 <!-- Commented out - can't move in the sample -->
 </moveMemberType>
 <nodeMaps>
 <nodeMap node="o=company,c=us" pluginNode="o=company,c=us" />
 <!-- Root DN configured in your environment -->
 </nodeMaps>
 </profileRepository>
```

14. In the PumaService.properties file, ensure that the following filters are correct for your environment and edit the values if needed.

For example:

```
WPS_install_dir\shared\app\config\services\PumaService.properties
user.base.attributes=cn,uid,cn,logonId,logonPasswordVerify,logonPassword ...
group.base.attributes=cn,uniqueOwnerIdentifier,membergroups,groupmembers,member
GroupName,memberGroupType,distinguishedName...
```

Ensure that the user.minimum.attributes values include all attributes for the user that the CMR will retrieve from the Oracle Access Manager back-end application and send to the WAS portal server. For example:

```
user.minimum.attributes=cn,genUserid,cn,givenName,sn,mail...
```

Ensure that you specify the uniquemember attribute to retrieve group members:

```
group.minimum.attributes=cn,uniquemember...
```

15. In the PumaService.properties file, ensure that all user.minimum.attributes have the correct attribute access control for the Administrator and all are in one of the Identity System panels, as described in the *Oracle Access Manager Identity and Common Administration Guide*.

    From the Identity System Console menu click User Manager Configuration, click the Configure Tab, click the tab link, click View Object Profile, then click Configure Panels.

16. Configure WebSphere Portal security by executing the following command.

    Note that some WPSConfig steps may overwrite all of the files that you modified prior to this step, so it is necessary to back up all of these files.

    ```
    WPS_install_dir/config/WPSConfig.bat action-secure-portal-ldap
    ```

17. Stop WebSphere Portal Server.
18. Make sure all the configuration file changes done before running WPSConfig.bat are in place.
19. Restart WebSphere Portal server.
20. Make sure the systemcred.dn is valid in the following file:

    ```
    WPS_install_dir\shared\app\config\services\VaultService.properties file
    ```

    For example:

    ```
    systemcred.dn=cn=PortalAdmin,o=company,c=us
    ```

    Note: This is always the fully qualified unquiet of wpsadmin.

21. Configure WebSphere Portal credentials by executing the following command
22. Access `http://host:port/wps/portal`.

    ```
    WPS_install_dir/config/WPSConfig.bat
    action-create-deployment-credentials
    ```

    where *host* is the fully qualified server name and *port* is the port number configured for the Portal Server.

23. Log in to the Portal as the Oracle Access Manager admin user.

Login should be successful and the Admin user should able to search for users and groups in the Oracle Access Manager repository.

Setting Up WebSphere Portal v5.1 With Oracle Access Manager

Integrating the WebSphere Portal v5.1 for Oracle Access Manager involves a series of installation and configuration tasks.

To integrate the WebSphere Portal v5.1 with Oracle Access Manager

1. Complete the tasks in "Preparing to Install the Connector" on page 11-11:
   a. Install the WebSphere Application Server with appropriate Fix Pack as described in the WebSphere documentation to correct the following issues.

      For example:

      Issue: Access permission is not working for any user dn containing intermediate spaces after the comma separators, as in the following: cn=PortalUser, o=company, c=us.

      Consult IBM Fix PQ93461 for details about this Portal User access permission problem.

      Action: This IBM fix pack needs to be applied. Please verify the Portal Server History Log to ensure that all required fixes have been installed. Refer to the Portal Infocenter documentation for additional details.

   b. Install the WebSphere Portal with WebSphere Application Server security disabled.

      Note: Both the Global and Java 2 security should be disabled while installing the Portal Server.

      See the IBM WebSphere Portal Infocenter documentation for details on installation.

   c. Apply Fixes PQ99439 and PK02868_510. This is required for custom user registry configuration of WebSphere Portal 5.1. (If these fixes are not applied, task enable-security-wmmur-custom will fail.)

   d. Apply the latest available "Member Manager cumulative fix for WebSphere Portal version 5.1." This includes a fix for the group membership feature. This cumulative fix is located at the following Web site:

      `http://www-1.ibm.com/support/docview.wss?rs=0&uid=swg24009153`

   e. Install and configure Oracle Access Manager, as covered in "Preparing to Install the Connector" on page 11-11.
2. Install the Connector for WebSphere and configure the NetpointWASRegistry and TAI components, as described in "Installing the Connector for WebSphere" on page 11-20.
3. Complete Connector setup and testing, as covered in "Completing Connector Setup" on page 11-26.
4. In the WebSphere Administration Console, change security to custom registry. This specifies the NetPointWASRegistry, which establishes a connection between

the WAS and Oracle Access Manager. The WAS uses the NetPointWASRegistry to authenticate and authorize portal users through Access System security policies.

5. Ensure that the following Admin credentials are set in clear text in the NetPointWASRegistry.properties file:

```
OB_AdminUserName=admin
OB_AdminUserCreds=password
```

Where the *OB_AdminUserName* value is the user ID of the Portal Server administrator who is a Master Identity Administrator or Oracle Access Manager Administrator. This is required for the CMR. The Admin credentials should be set in clear text. The NetPointWASRegistry reads the password, encrypts it, and rewrites the properties file with the encrypted password. The encryptor can be executed by running the registryTester program, as well as from WebSphere. To assist you with adding these parameters, see the NetPointWASRegistryProperties.sample file, which includes comments. For additional details, consult, "NetPointWASRegistry.properties" on page 11-71.

> **Note:** NetPointWASRegistry.properties file formatting is lost when the Connector for WebSphere rewrites the file with the encrypted password. Therefore, you may want to save a copy of the NetPointWASRegistry.properties.

6. Restart the WAS Server.
7. Ensure that both the WebSphere Application server and Portal Server are stopped.
8. Make a back-up copy of the following file:

   WPS_install_dir/config/wpconfig.properties

   Where *WPS_install_dir* is the directory where WebSphere Portal Server is installed.
9. Edit the wpconfig.properties file and add the following:

```
PortalAdminId (Example: PortalAdminId= DN of wpsadministrator )
PortalAdminIdShort (Example: PortalAdminId= wpsadministrator )
PortalAdminPwd (Example: PortalAdminPwd=wpsadminpassword )
WasUserid (Example: WasUserid=wasadministrator )
WasPassword (Example: WasPassword=wasadminpassword )
PortalAdminGroupId (Example: PortalAdminGroupId= DN of PortalAdminGroupId )
PortalAdminGroupIdShort (Example: PortalAdminGroupIdShort= PortalAdminGroupId )
LTPAPassword (Example: LTPAPassword= Password configured for LTPA in the
AppServer Configuration)
WmmSystemId (Example: WmmSystemId= Set this value to same as that of
PortalAdminIdShort)
WmmSystemIdPassword (Example: WmmSystemIdPassword= Set this value to same as
that of PortalAdminPwd)
LDAPSuffix (Example: LDAPSuffix=o=company,c=us LDAP suffix of Oracle Access
Manager installation)
LDAPUserSuffix (Example: LDAPUserSuffix=ou=users Keep this blank if user-nodes
are directly under LDAPSuffix)
LDAPGroupSuffix (Example: LDAPGroupSuffix=ou=groups Keep this blank if
group-nodes are directly under LDAPSuffix)
LdapUserPrefix (Example: LdapUserPrefix=cn )
LdapGroupPrefix (Example: LdapGroupPrefix=cn )
PortalAdminId (Example: PortalAdminId= DN of wpsadministrator )
PortalAdminIdShort (Example: PortalAdminId= wpsadministrator )
PortalAdminPwd (Example: PortalAdminPwd=wpsadminpassword )
```

```
WasUserid (Example: WasUserid=wasadministrator )
WasPassword (Example: WasPassword=wasadminpassword )
PortalAdminGroupId (Example: PortalAdminGroupId= DN of PortalAdminGroupId )
PortalAdminGroupIdShort (Example: PortalAdminGroupIdShort= PortalAdminGroupId )
LTPAPassword (Example: LTPAPassword= Password configured for LTPA in the
AppServer Configuration)
WmmSystemId (Example: WmmSystemId= Set this value to same as that of
PortalAdminIdShort)
WmmSystemIdPassword (Example: WmmSystemIdPassword= Set this value to same as
that of PortalAdminPwd)
LDAPSuffix (Example: LDAPSuffix=o=company,c=us LDAP suffix of Oracle Access
Manager installation)
LDAPUserSuffix (Example: LDAPUserSuffix=ou=users Keep this blank if user-nodes
are directly under LDAPSuffix)
LDAPGroupSuffix (Example: LDAPGroupSuffix=ou=groups Keep this blank if
group-nodes are directly under LDAPSuffix)
LdapUserPrefix (Example: LdapUserPrefix=cn )
LdapGroupPrefix (Example: LdapGroupPrefix=cn )
```

10. Restart the WebSphere Application server.
11. Optional: Turn on PUMA traces in Portal, which can be done by entering the following in the log.properties file, which typical resides at

    WPS_install_dir\shared\app\config\log.properties

    traceString=com.ibm.wps.services.puma.*=all=enabled:com.ibm.wps.puma.*=all=enabled:com.ibm.wps.command.puma.*=all=enable
12. Back up the following files:

    WPS_install_dir/wmm/wmm.xml

    WPS_install_dir/wmm/wmm_DB.xml
13. Make a copy of wmm.xml and save it under the name wmm_custom.xml in the same folder as wmm.xml (*WPS_install_dir*/wmm/wmm_custom.xml).

    Edit wmm_custom.xml, changing your CMR configuration as follows, and verify that the lookaside flag is set to false.

```
<supportedMemberTypes>

<supportedMemberType name="Person"
 rdnAttrTypes="uid"
 defaultParentMember="o=company,c=us"
 defaultProfileRepository="CNR"/>
 <!-- o=company,c=us is Root DN. please enter the DN in your environment. -->

 <supportedMemberType name="Group"
 rdnAttrTypes="cn"
 defaultParentMember="o=company,c=us"
 defaultProfileRepository="CNR"/>
 <!-- Name of the CMR information tag -->
 </supportedMemberTypes>

 <profileRepository name="NetpointCustomRepository" UUID="CNR"
 description="This is Oracle WMM custom MemberRepository implementation."
 vendor="Oracle"
 adapterClassName="com.oblix.registry.NetPointMemberRepositoryImpl_v51"
 specVersion="1.0" adapterVersion="2.0"
 configurationFile="customRepositoryAttributes.xml"
 wmmGenerateExtId="false" supportGetPersonByAccountName="false"
```

```
 supportDynamicAttributes="false" profileRepositoryForGroups="CNR"
 enableTrace="true "
PumaService.properties="WPSInstallDir/shared/app/config/services/PumaService.pr
operties"
NetPointWASRegistry.properties="NetpointWASConnInstallDir\oblix\config\NetPoint
WASRegistry.properties">

<!-- 1. com.oblix.registry.NetPointMemberRepositoryImpl_v51 is name of the CMR
implementation class
2. Mention path of the PUMA service against PumaService.properties paramter.
3. NetPointWASRegistry.properties - Path of the Oracle Access Manager WAS
registry configuration file.
For Unix this path should be mentioned as
"NetpointWASConnInstallDir/oblix/config/NetPointWASRegistry.properties
4. CustomRepositoryAttributes.xml is a dummy file.
 -->

 <readMemberType>

 <memberType name="Person" /> <memberType name="Group" />
 <!-- Only read access to portal is provided by COREid Connector CMR-->
 </readMemberType>
 <createMemberType>
 <!-- <memberType name="Person" /> <memberType name="Group" /> -->
 <!-- Commented out - can't create in the sample -->
 </createMemberType>
<updateMemberType>
 <!-- <memberType name="Person" /> <memberType name="Group" / > -->
 <!-- Commented out - can't update in the sample -->
 </updateMemberType>
 <deleteMemberType>
 <!-- <memberType name="Person" /> <memberType name="Group" /> -->
 <!-- Commented out - can't delete in the sample -->
 </deleteMemberType>
 <renameMemberType>
 <!-- <memberType name="Person" /> <memberType name="Group" /> -->
 <!-- Commented out - can't rename in the sample -->
 </renameMemberType>
 <moveMemberType>
 <!-- <memberType name="Person" /> <memberType name="Group" /> -->
 <!-- Commented out - can't move in the sample -->
 </moveMemberType>
 <nodeMaps>
 <nodeMap node="o=company,c=us" pluginNode="o=company,c=us" />
 <!-- Root DN configured in your environment -->
 </nodeMaps>
 </profileRepository>
```

14. Overwrite the following files with new contents of wmm_custom.xml:

    WPS_install_dir/wmm/wmm.xml

    WPS_install_dir/wmm/wmm_DB.xml

15. Ensure that the following filters are correct for your environment in the following file:

    WPS_install_dir\shared\app\config\services\PumaService.properties

    In this file, ensure that the user.minimum.attributes include all attributes for the user that the CMR will retrieve from Oracle Access Manager and send to the WAS portal server.

In the required attributes list in user.base.attributes, enter all attributes that are required for searching in the WAS portal server for users with these attribute values.

Make sure that group.minimum.attributes has assigned uniqueMember and class attribute configured for Group object class. (In the following sample file it is "cn").

The user.password.attribute should be set to name of the password attribute.

Ensure that directory is set to CUSTOM.

Edit the values, as needed.

```
#SAMPLE PumaService.properties file. Please make changes according to your
environment

#In the following sample file. User object classAttribute is uid and Group
object class attribute is cn.

#Please set these values according to your environment.

user.fbadefault.filter=uid
user.template.attribute=uid
user.password.attribute=userPassword
user.minimum.attributes=uid,cn
user.base.attributes=uid,cn,givenName,sn,preferredLanguage
user.sync.remove.attributes=passwordCreation,RDN,lastSession,selfAddress,packag
eSuppression,createdTimestamp,policyAccountId,registrationUpdate,uniqueUserIden
tifier,memberId,ancestors,addressId,truncatedUniqueIdentifier,profileType,regis
tration,primary,approvalState,lastOrder,publishPhone2,publishPhone1,salt,addres
sBookId,uniqueParentIdentifier,parentMemberId,registrationCancel,passwordExpire
d,previousLastSession,displayName,addressType,shortName,preferredLanguageId,use
rId,timeout,registerType,uniqueIdentifier,passwordInvalid,nickName,type,memberg
roups,income,age,organizationUnitId,demographicField6,children,household,status
,passwordRetries,uniqueNumericIdentifier,cn
group.fbadefault.filter=cn
group.template.attribute=cn
group.minimum.attributes=cn,uniqueMember
group.base.attributes=cn,uniqueMember
group.sync.remove.attributes=uniqueOwnerIdentifier,membergroups,groupmembers,me
mberGroupName,memberGroupType,distinguishedName,cn,uniqueNumericIdentifier,uniq
ueMemberIdentifier
directory=CUSTOM
ejbName=ejb/MemberServiceHome
```

16. Ensure that all user.minimum.attributes have the correct attribute access controls for the Administrator.

    See *Oracle Access Manager Identity and Common Administration Guide* for details.

    Also ensure that all user.minimum.attributes are in one of the Identity System panels, as described in the *Oracle Access Manager Identity and Common Administration Guide*.

    To access these panels, from the Identity System Console, click User Manager Configuration, click Tabs, click the link for the tab, click View Object Profile, then click Configure Panels.

17. Modify the file *WPS_install_dir*/wmm/wmmWASAdmin.xml as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<wmmWASAdmins>
  <admin logonId="PortalAdmin" logonPassword="oblixoblix"
```

```
uniqueUserId="cn=Portal Admin,o=company,c=us"/>
</wmmWASAdmins>
```

logonId, logonPassword and uniqueUserId correspond to the new values of PortalAdminIdShort, PortalAdminId, and PortalAdminPwd respectively in wpconfig.properties.

18. Modify the *WPS_install_dir*/wmm/wmmur.xml file.

    Set the value of `wmmnode` to your `LDAPRoot` as follows:

    ```
    <node wmmnode="o=company,c=us"/>
    ```

19. Modify *WPS_install_dir*/wmm/wmmAttributes.xml so that it contains only supported attributes.

    The `wmmAttributes.xml` file contains definitions of all the supported attributes. Each attribute definition might have two properties named `applicableMemberTypes` and `requiredMemberTypes`. Values of these properties should be supported member types. For example, if `wmm_custom.xml` has only two `supportedMemberTypes` (`Person` and `Group`), then other types such as `organizationalUnit` or `organization` should not be `applicableMemberTypes` or `requiredMemberTypes` for any of the attributes in `wmmAttributes.xml`.

20. Modify *WPS_install_dir*/shared/app/config/services/VaultService.properties.

    Set the value of `systemcred.dn` to the DN of `wpsadministrator`.

    Example:

    ```
    systemcred.dn=cn=Portal Admin,o=company,c=us
    ```

21. Backup all configuration files, stop the portal server, then configure WebSphere Portal security by executing the following command:

    WPS_install_dir/config/WPSConfig.bat enable-security-wmmur-custom

22. Verify that all of the configuration files are in place.

    You can compare the values with the backup copies that you created in the previous step.

    The contents of `wmm.xml` should match the contents of `wmm_custom.xml`.

23. Restart the portal server.

24. Access the following Web page:

    `http://`*`host`*`:`*`port`*`/wps/portal`

    where `host` is the fully qualified server name and *port* is the port number configured for the Portal Server.

25. Log in to the Portal as the Oracle Access Manager admin user. The Login should succeed, and Admin user should able to search for other Oracle Access Manager Repository users and groups.

Setting Up WebSphere Portal v6.0 With Oracle Access Manager

Integrating the WebSphere Portal v5.1 for Oracle Access Manager involves a series of installation and configuration tasks.

To integrate the WebSphere Portal v6.0 with Oracle Access Manager

1. Complete the tasks in "Preparing to Install the Connector" on page 11-11:
   a. Install the WebSphere Application Server.
   b. Install the WebSphere Portal with WebSphere Application Server security disabled.

      Note: Both the Global and Java 2 security should be disabled while installing the Portal Server.

      See the IBM WebSphere Portal Infocenter documentation for details on installation.
   c. Install and configure Oracle Access Manager, as covered in "Preparing to Install the Connector" on page 11-11.
2. Install the Connector for WebSphere and configure the NetpointWASRegistry and TAI components, as described in "Installing the Connector for WebSphere" on page 11-20.
3. Complete Connector setup and testing, as covered in "Completing Connector Setup" on page 11-26.
4. In the WebSphere Administration Console, change security to custom registry. This specifies the NetPointWASRegistry, which establishes a connection between the WAS and Oracle Access Manager. The WAS uses the NetPointWASRegistry to authenticate and authorize portal users through Access System security policies.
5. Ensure that the following Admin credentials are set in clear text in the NetPointWASRegistry.properties file:

   ```
   OB_AdminUserName=admin
   OB_AdminUserCreds=password
   ```

   Where the *OB_AdminUserName* value is the user ID of the Portal Server administrator who is a Master Identity Administrator or Oracle Access Manager Administrator. This is required for the CMR. The Admin credentials should be set in clear text. The NetPointWASRegistry reads the password, encrypts it, and rewrites the properties file with the encrypted password. The encryptor can be executed by running the registryTester program, as well as from WebSphere. To assist you with adding these parameters, see the NetPointWASRegistryProperties.sample file, which includes comments. For additional details, consult, "NetPointWASRegistry.properties" on page 11-71.

   Note: NetPointWASRegistry.properties file formatting is lost when the Connector for WebSphere rewrites the file with the encrypted password. Therefore, you may want to save a copy of the NetPointWASRegistry.properties.
6. Restart the WAS Server.
7. Ensure that both the WebSphere Application server and Portal Server are stopped.
8. Make a back-up copy of the following file:

   WPS_install_dir/config/wpconfig.properties

   Where *WPS_install_dir* is the directory where WebSphere Portal Server is installed.

9. Edit the wpconfig.properties file and add the following:

```
PortalAdminId (Example: PortalAdminId= DN of wpsadministrator)
PortalAdminIdShort (Example: PortalAdminId= wpsadministrator)
PortalAdminPwd (Example: PortalAdminPwd=wpsadminpassword)
WasUserid (Example: WasUserid=wasadministrator)
WasPassword (Example: WasPassword=wasadminpassword)
PortalAdminGroupId (Example: PortalAdminGroupId= DN of PortalAdminGroupId)
PortalAdminGroupIdShort (Example: PortalAdminGroupIdShort= PortalAdminGroupId)
LTPAPassword (Example: LTPAPassword= Password configured for LTPA in the
AppServer Configuration)
WmmSystemId (Example: WmmSystemId= Set this value to same as that of
PortalAdminIdShort)
WmmSystemIdPassword (Example: WmmSystemIdPassword= Set this value to same as
that of PortalAdminPwd)
LDAPSuffix (Example: LDAPSuffix=o=company,c=us LDAP suffix of Oracle Access
Manager installation)
LDAPUserSuffix (Example: LDAPUserSuffix=ou=users Keep this blank if user-nodes
are directly under LDAPSuffix)
LDAPGroupSuffix (Example: LDAPGroupSuffix=ou=groups Keep this blank if
group-nodes are directly under LDAPSuffix)
LdapUserPrefix (Example: LdapUserPrefix=cn)
LdapGroupPrefix (Example: LdapGroupPrefix=cn)

PortalAdminId (Example: PortalAdminId= DN of wpsadministrator )
PortalAdminIdShort (Example: PortalAdminId= wpsadministrator )
PortalAdminPwd (Example: PortalAdminPwd=wpsadminpassword )
WasUserid (Example: WasUserid=wasadministrator )
WasPassword (Example: WasPassword=wasadminpassword )
PortalAdminGroupId (Example: PortalAdminGroupId= DN of PortalAdminGroupId )
PortalAdminGroupIdShort (Example: PortalAdminGroupIdShort= PortalAdminGroupId )
LTPAPassword (Example: LTPAPassword= Password configured for LTPA in the
AppServer Configuration)
WmmSystemId (Example: WmmSystemId= Set this value to same as that of
PortalAdminIdShort)
WmmSystemIdPassword (Example: WmmSystemIdPassword= Set this value to same as
that of PortalAdminPwd)
LDAPSuffix (Example: LDAPSuffix=o=company,c=us LDAP suffix of Oracle Access
Manager installation)
LDAPUserSuffix (Example: LDAPUserSuffix=ou=users Keep this blank if user-nodes
are directly under LDAPSuffix)
LDAPGroupSuffix (Example: LDAPGroupSuffix=ou=groups Keep this blank if
group-nodes are directly under LDAPSuffix)
LdapUserPrefix (Example: LdapUserPrefix=cn )
LdapGroupPrefix (Example: LdapGroupPrefix=cn )
PortalAdminId (Example: PortalAdminId= DN of wpsadministrator )
PortalAdminIdShort (Example: PortalAdminId= wpsadministrator )
PortalAdminPwd (Example: PortalAdminPwd=wpsadminpassword )
WasUserid (Example: WasUserid=wasadministrator )
WasPassword (Example: WasPassword=wasadminpassword )
PortalAdminGroupId (Example: PortalAdminGroupId= DN of PortalAdminGroupId )
PortalAdminGroupIdShort (Example: PortalAdminGroupIdShort= PortalAdminGroupId )
LTPAPassword (Example: LTPAPassword= Password configured for LTPA in the
AppServer Configuration)
WmmSystemId (Example: WmmSystemId= Set this value to same as that of
PortalAdminIdShort)
WmmSystemIdPassword (Example: WmmSystemIdPassword= Set this value to same as
that of PortalAdminPwd)
LDAPSuffix (Example: LDAPSuffix=o=company,c=us LDAP suffix of Oracle Access
Manager installation)
LDAPUserSuffix (Example: LDAPUserSuffix=ou=users Keep this blank if user-nodes
```

```
are directly under LDAPSuffix)
LDAPGroupSuffix (Example: LDAPGroupSuffix=ou=groups Keep this blank if
group-nodes are directly under LDAPSuffix)
LdapUserPrefix (Example: LdapUserPrefix=cn )
LdapGroupPrefix (Example: LdapGroupPrefix=cn )
```

Note that the script may fail unless you set the following parameters:

```
WpsContentAdministrators (Example: WpsContentAdministrators = DN of Content
Administrator group)
WpsDocReviewer (Example: WpsDocReviewer = DN of Document Reviewer group)
```

10. Restart the WebSphere Application server.
11. Optional: Turn on PUMA traces in Portal, which can be done by entering the following in the log.properties file, which typical resides at

    WPS_install_dir\shared\app\config\log.properties

    ```
    traceString=com.ibm.wps.services.puma.*=all=enabled:com.ibm.w
    ps.puma.*=all=enabled:com.ibm.wps.command.puma.*=all=enable
    ```

12. Back up the following files:

    WPS_install_dir/wmm/wmm.xml

    WPS_install_dir/wmm/wmm_DB.xml

13. Make a copy of wmm.xml and save it under the name wmm_custom.xml in the same folder as wmm.xml (*WPS_install_dir*/wmm/wmm_custom.xml).

    Edit wmm_custom.xml, changing your CMR configuration as follows, and verify that the lookaside flag is set to false.

```
<supportedMemberTypes>

<supportedMemberType name="Person"
 rdnAttrTypes="uid"
 defaultParentMember="o=company,c=us"
 defaultProfileRepository="CNR"/>
 <!-- o=company,c=us is Root DN. please enter the DN in your environment. -->

 <supportedMemberType name="Group"
 rdnAttrTypes="cn"
 defaultParentMember="o=company,c=us"
 defaultProfileRepository="CNR"/>
 <!-- Name of the CMR information tag -->
 </supportedMemberTypes>

 <profileRepository name="NetpointCustomRepository" UUID="CNR"
 description="This is Oracle WMM custom MemberRepository implementation."
 vendor="Oracle"
 adapterClassName="com.oblix.registry.NetPointMemberRepositoryImpl_v51"
 specVersion="1.0" adapterVersion="2.0"
 configurationFile="customRepositoryAttributes.xml"
 wmmGenerateExtId="false" supportGetPersonByAccountName="false"
 supportDynamicAttributes="false" profileRepositoryForGroups="CNR"
 enableTrace="true "

PumaService.properties="WPSInstallDir/config/properties/PumaService.properties"
NetPointWASRegistry.properties="NetpointWASConnInstallDir\oblix\config\NetPoint
WASRegistry.properties">

<!-- 1. com.oblix.registry.NetPointMemberRepositoryImpl_v51 is name of the CMR
```

```
implementation class
2. Mention path of the PUMA service against PumaService.properties paramter.
3. NetPointWASRegistry.properties - Path of the Oracle Access Manager WAS
registry configuration file.
For Unix this path should be mentioned as
"NetpointWASConnInstallDir/oblix/config/NetPointWASRegistry.properties
4. CustomRepositoryAttributes.xml is a dummy file.
 -->

 <readMemberType>

 <memberType name="Person" /> <memberType name="Group" />
 <!-- Only read access to portal is provided by COREid Connector CMR-->
 </readMemberType>
 <createMemberType>
 <!-- <memberType name="Person" /> <memberType name="Group" /> -->
 <!-- Commented out - can't create in the sample -->
 </createMemberType>
<updateMemberType>
 <!-- <memberType name="Person" /> <memberType name="Group" / > -->
 <!-- Commented out - can't update in the sample -->
 </updateMemberType>
 <deleteMemberType>
 <!-- <memberType name="Person" /> <memberType name="Group" /> -->
 <!-- Commented out - can't delete in the sample -->
 </deleteMemberType>
 <renameMemberType>
 <!-- <memberType name="Person" /> <memberType name="Group" /> -->
 <!-- Commented out - can't rename in the sample -->
 </renameMemberType>
 <moveMemberType>
 <!-- <memberType name="Person" /> <memberType name="Group" /> -->
 <!-- Commented out - can't move in the sample -->
 </moveMemberType>
 <nodeMaps>
 <nodeMap node="o=company,c=us" pluginNode="o=company,c=us" />
 <!-- Root DN configured in your environment -->
 </nodeMaps>
 </profileRepository>
```

14. Overwrite the following files with new contents of wmm_custom.xml:

    WPS_install_dir/wmm/wmm.xml

    WPS_install_dir/wmm/wmm_DB.xml

15. Ensure that the following filters are correct for your environment in the following file:

    WPS_install_dir\config\properties\PumaService.properties

    In this file, ensure that the user.minimum.attributes include all attributes for the user that the CMR will retrieve from Oracle Access Manager and send to the WAS portal server.

    In the required attributes list in user.base.attributes, enter all attributes that are required for searching in the WAS portal server for users with these attribute values.

    Make sure that group.minimum.attributes has assigned uniqueMember and class attribute configured for Group object class. (In the following sample file it is "cn").

    The user.password.attribute should be set to name of the password attribute.

Ensure that directory is set to CUSTOM.

Edit the values, as needed.

```
#SAMPLE PumaService.properties file. Please make changes according to your
environment

#In the following sample file. User object classAttribute is uid and Group
object class attribute is cn.

#Please set these values according to your environment.

user.fbadefault.filter=uid
user.template.attribute=uid
user.password.attribute=userPassword
user.minimum.attributes=uid,cn
user.base.attributes=uid,cn,givenName,sn,preferredLanguage
user.sync.remove.attributes=passwordCreation,RDN,lastSession,selfAddress,packag
eSuppression,createdTimestamp,policyAccountId,registrationUpdate,uniqueUserIden
tifier,memberId,ancestors,addressId,truncatedUniqueIdentifier,profileType,regis
tration,primary,approvalState,lastOrder,publishPhone2,publishPhone1,salt,addres
sBookId,uniqueParentIdentifier,parentMemberId,registrationCancel,passwordExpire
d,previousLastSession,displayName,addressType,shortName,preferredLanguageId,use
rId,timeout,registerType,uniqueIdentifier,passwordInvalid,nickName,type,memberg
roups,income,age,organizationUnitId,demographicField6,children,household,status
,passwordRetries,uniqueNumericIdentifier,cn
group.fbadefault.filter=cn
group.template.attribute=cn
group.minimum.attributes=cn,uniqueMember
group.base.attributes=cn,uniqueMember
group.sync.remove.attributes=uniqueOwnerIdentifier,membergroups,groupmembers,me
mberGroupName,memberGroupType,distinguishedName,cn,uniqueNumericIdentifier,uniq
ueMemberIdentifier
directory=CUSTOM
ejbName=ejb/MemberServiceHome
```

16. Ensure that all user.minimum.attributes have the correct attribute access controls for the Administrator.

    See *Oracle Access Manager Identity and Common Administration Guide* for details.

    Also ensure that all user.minimum.attributes are in one of the Identity System panels, as described in the *Oracle Access Manager Identity and Common Administration Guide*.

    To access these panels, from the Identity System Console, click User Manager Configuration, click Tabs, click the link for the tab, click View Object Profile, then click Configure Panels.

17. Modify the file *WPS_install_dir*/wmm/wmmWASAdmin.xml as follows:

    ```
    <?xml version="1.0" encoding="UTF-8"?>
    <wmmWASAdmins>
      <admin logonId="PortalAdmin" logonPassword="oblixoblix"
    uniqueUserId="cn=Portal Admin,o=company,c=us"/>
    </wmmWASAdmins>
    ```

    Where logonId, logonPassword and uniqueUserId correspond to the new values of PortalAdminIdShort, PortalAdminPwd, and PortalAdminId respectively in wpconfig.properties.

18. Modify the *WPS_install_dir*/wmm/wmmur.xml file.

Set the value of wmmnode to your LDAPRoot as follows:

```
<node wmmnode="o=company,c=us"/>
```

19. Modify *WPS_install_dir*/wmm/wmmAttributes.xml so that it contains only supported attributes.

    The wmmAttributes.xml file contains definitions of all the supported attributes. Each attribute definition might have two properties named applicableMemberTypes and requiredMemberTypes. Values of these properties should be supported member types. For example, if wmm_custom.xml has only two supportedMemberTypes (Person and Group), then other types such as organizationalUnit or organization should not be applicableMemberTypes or requiredMemberTypes for any of the attributes in wmmAttributes.xml.

20. Modify *WPS_install_dir*/config/properties/VaultService.properties.

    Set the value of systemcred.dn to the DN of wpsadministrator.

    Example:

    ```
    vault.default-release.class=com.ibm.wps.services.credentialvault.DefaultVault
    vault.default-release.config=defaultvault
    vault.default-release.domain=rel
    vault.default-release.manageresources=true
    vault.default-release.readonly=false
    systemcred.dn=cn=Portal Admin,o=company,c=us
    ```

21. Back up all configuration files, stop the portal server, then configure WebSphere Portal security by executing the following command:

    WPS_install_dir/config/WPSConfig.bat enable-security-wmmur-custom

22. Verify that all of the configuration files are in place.

    You can compare the values with the backup copies that you created in the previous step.

    The contents of wmm.xml should match the contents of wmm_custom.xml.

23. Restart the portal server.

24. Access the following Web page:

    http://*host*:*port*/wps/portal

    where host is the fully qualified server name and *port* is the port number configured for the Portal Server.

25. Log in to the Portal as the Oracle Access Manager admin user. The Login should succeed, and Admin user should able to search for other Oracle Access Manager Repository users and groups.

Managing Users and Groups with Portal v5 and v6

Portal Administrators can use the Identity System to perform user and group management tasks such as adding or deleting users and groups, modifying user profiles and attributes.

To use the Identity System user and group management functionality, ensure that you do not create users and groups in the WebSphere Portal. Instead, create and modify users and groups in the Identity System.

You can add and delete static groups and user membership in groups through the Identity System. The information that you update in the Identity System is immediately reflected in the WebSphere Portal.

Note: To recognize group membership, the Identity System requires the dynamic group to be expanded.

After you create users and groups in the Identity System, you can search for them in the WebSphere Portal.

See the *Oracle Access Manager Identity and Common Administration Guide* for more information on managing users and groups.

Modifying User Profiles and Attributes with Portal v5 and v6

When users modify their profile through the Identity System, the modifications are immediately visible in the WebSphere Portal. This ensures that the most current user information is used when portal developers personalize user pages.

You can map additional attributes to a use's profile if necessary. See the WebSphere Portal documentation for information on mapping attributes.

Password Management with Portal v5 and v6

Because the portal uses Access System SSO, users are subject to the Oracle Access Manager password policies during authentication.

Important: To implement the password management feature, turn off the portal's password management functionality.

The Oracle Access Manager password management functionality includes defining password policies, resetting passwords, expiration notification, and challenge phrases for lost passwords.

See the *Oracle Access Manager Identity and Common Administration Guide* for more information on password policies.

Access Control for the WebSphere Portal v5 and v6

Portal administrators use the portal's access control functionality to grant access to portlets. From the WebSphere Portal, administrators can search for Oracle Access Manager-managed users and groups to whom they want to grant portal administration privileges as well as portlet access control.

Configuring Single Sign-on Functions for the Portal v5 and v6

Configuring SSO between the Access System and the WebSphere portal enables the WebSphere portal to utilize the ObSSOCookie and enable Connector for WebSphere to authenticate Oracle Access Manager users.

Configuring SSO logout for the WebSphere Portal Server ensures that when a user logs out of a Access System-protected WebSphere resource, both the LTPA token and the ObSSOCookie are killed. The user will not be able to access any other WebSphere resource or other Access System-protected resources without authenticating again.

- If you have configured the TAI for single sign-on between Oracle Access Manager and WebSphere, you must configure single sign-on logout for the WebSphere Portal Server.
- If you have not configured the TAI for single sign-on, users can use the portal's logout button to log out of all Access System-protected resources.

To configure single sign-on for the WebSphere Portal v5 and v6

1. Install the WebGate plug-in for the Web server that you selected when you installed the WebSphere Portal.
2. In the Policy Manager, define the URL that you want to protect.

A WebGate prompts for authentication when users attempt to log in to this URL. Be sure to protect / if you want the WebGate to prompt for authentication when the user gets to the root of the WebSphere Portal. You can also add other authorization rules, if needed.

Note: To protect resources, Oracle recommends that you use a form-based authentication scheme. However, if you use the basic authentication scheme, set the Challenge Redirect field to another WebGate to ensure that the ObSSOCookie gets set. See the *Oracle Access Manager Access System Administration Guide* for more information on authentication schemes.

To configure single sign-on logout for WebSphere Portal v5 and v6

1. Create an Access System policy with a Form Over LDAP type of authentication scheme to protect the portal URL, as described in the *Oracle Access Manager Access System Administration Guide*.
2. Create a custom logout page using HTML, JSP, or CGI protocol.

   The default logout page for Oracle Access Manager, logout.html, is located in:

   WebGate_install_dir\access\oblix\apps\common\bin

   Where *WebGate_install_dir* is the directory where the WebGate is installed.
3. Save the logout page in the document root of the Web server on which the WebGate that protects WebSphere is installed.

   For example:

   ```
   http://foobar/myportal/logout.html
   ```

   Note: Ensure that the name of the logout page contains the string "logout".

   ---

4. Protect the logout page with an Access System policy, as described in the *Oracle Access Manager Access System Administration Guide*.
5. Locate and open the following file:

   For WebSphere Portal version 5:

   WPS_install_dir\shared\app\config\services\ConfigServices.properties

   For WebSphere Portal version 6:

   WPS_install_dir\config\properties\ConfigServices.properties

Where *WPS_install_dir* is the directory where WebSphere Portal Server is installed.

6. Add the following two parameters in ConfigServices.properties file:

   ```
   redirect.logout = true
   ```

   ```
   redirect.logout.url = The path to the logout page
   ```

   For example:

   - `http://foobar/myportal/logout.html`

7. For WebSphere Portal v5, back up the properties files in *WPS_Install_Dir*/config/properties/
8. For Websphere Portal v6, back up the properties files in the following directory:

   WPS_Install_Dir/config/properties/

   Ensure that the Application Server and Portal Server are not running.

   - On Windows, run the following command:

     ```
     WPS_Install_Dir\config\WPSconfig.bat update-properties
     ```

   - On UNIX, run the following command:

     ```
     WPS_Install_Dir/config/WPSconfig.sh update-properties
     ```

   After successfully completing the commands, restore the properties files from backup.

9. Restart the WebSphere Portal Server and Application Server.

Configuration Files

The following configuration files are used when integrating Connector for WebSphere with WAS:

- NetPointWASRegistry.properties
- WebGate.properties
- TrustedServers.properties

NetPointWASRegistry.properties

Table 11–1 describes the parameters of NetPointWASRegistry.properties file located in *CWS_install_dir* /oblix/config. This file contains data that was specified during NetPointWASRegistry installation, as well as some default parameter values for logging. For example:

```
# Logging level (none, info or debug);
OB_LogLevel=debug
OB_LogFileName=C:/CWS_install_dir/log
```

The Oracle Access Manager Connector for WebSphere caches data that it receives from the Access or Identity Server. For subsequent requests from WebSphere servers, user and group data is retrieved from the connector cache. The caches are updated on a scheduled basis. Each user, group, or user profile attribute entry in the connector cache is associated with a Time To Live (TTL) parameter that is defined in the NetPointWASRegistry.properties file. When a request is issued after the timeout limit is reached, the cache miss handler is invoked and fresh data is retrieved from the Identity and Access servers. The default timeout for all cache parameters in the

NetpointWASRegistry.properties file is 3600 seconds. This value can be set as needed. There is no dynamic cache updating between Identity and Access Servers and the connector. To reflect user, group, or user profile attribute changes immediately, set the cache timeout value to 0. A value of 0 disables the connector cache.

Note: Webpass to GroupSrvCenter performance has been improved with the addition of configuration options to improve IdentityXML calls in the Identity System. For example, when no nested groups are used and are turned off, you may use a new option so that getGroups will not generate a request for nested groups.

Tip: See also Table 11–1.

Table 11–1 Parameters in NetPointWASRegistry.properties

Parameter Name	Description	Optional/ Mandatory
Installation	.	.
OB_InstallDir	The directory where NetPointWASRegistry is installed.	Mandatory
Logging	.	.
OB_LogLevel	The logging level that is recorded in the log file. Values are none, info, and debug.	Optional
OB_LogFileName	The file name for Custom User Registry (NetPointWASRegistry) log messages. Default = CWS_install_dir/log **Note**: Log messages for the CMR are directed to the *WPS_install_dir*/log/appserver-out.log file.	Optional
OB_LogMilliSeconds=true	The data/time format of log messages in the file specified with OB_LogFileName. When true, log messages are time formatted in milliseconds. Default =true	Optional
WebPass	.	.
OB_WebPassHost	The WebPass server host machine name. The host name must be fully qualified; for example, OB_WebPassHost=hostname.acme.com. To configure multiple WebPass instances for failover purposes, separate the names with a comma. For example: `OB_WebPassHost=foo.domain.com, bar.domain.com` Note that the host name corresponds to the port number in the specified order. See the example in the Ob_WebPassPort description in this table.	Mandatory

Table 11–1 (Cont.) Parameters in NetPointWASRegistry.properties

Parameter Name	Description	Optional/ Mandatory
OB_WebPassPort	The port number of the host machine. To configure multiple WebPass instances, separate the port numbers with a comma. For example: OB_WebPassPort=80, 81 Note that the host name corresponds to the port number in the specified order. In this example, the *hostname:port* number pairing is as follows: foo.domain.com:80 bar.domain.com:81 For failover to work, all other variables such as user name, credentials and webgate protection must be the same.	Mandatory
OB_WebPassIsProtected	Values are true and false. If WebPass is protected, set value=true.	Mandatory
OB_AdminUserName	The Identity System requires the Admin username and password to make IdentityXML calls to the WebPass. For details about administrator rights, see "Configuring the Identity Server" on page 11-12.	Mandatory
OB_AdminUserCreds	The Identity System requires the Admin username and password to make IdentityXML calls to the WebPass. Without the password the connector will not work. **Note**: You need to enter a clear-text password, which the program will encrypt and rewrite to the properties file after the first run.	Mandatory
Cookie	.	.
OB_CookieDomain	The cookie domain specified in the WebGate installer configuration. Needed if WebPass is protected. For example, .xyz.com	Mandatory
OB_CookiePath	The cookie path specified in the WebGate configuration. Needed if WebPass is protected. Default = /	Mandatory
WebPass SSL	.	.
OB_WebPassSSLEnabled	Specifies whether WebPass needs HTTPS connection. Values are true and false. Default = false	Mandatory
Login and Search Attributes	.	.
OB_UserAttr	The unique user identification (for example, uid).	Mandatory
OB_UserSearchAttr	The DN prefix for users from LDAP (for example, cn).	Mandatory

Table 11–1 (Cont.) Parameters in NetPointWASRegistry.properties

Parameter Name	Description	Optional/ Mandatory
OB_GroupSearchAttr	The DN prefix for groups from LDAP (for example, cn).	Mandatory
Active Directory Forest	.	.
OB_WebPassADDomain	Optional. The domain of the Admin user. To be used in case of Active Directory Forest with multiple domains. For example, OB_ WebPassADDomain=ou=company,dc=qalab, dc=acme,dc=com The ADDomain must be the same as the default defined in the Identity Server.	Optional
Records Returned	.	.
OB_WebPassXPIRecordsReturned	Optional. The number of records to return for getUsers or getGroups. This is used only in the WebSphere Portal. Default = return all	Optional
Authentication	.	.
OB_ AuthnSchemeResourceTypeName	Authen	Mandatory
OB_AuthnSchemeOperation	LOGIN	Mandatory
OB_AuthnSchemeResourceName	/Authen/Basic	Mandatory
OB_AuthzActionType	WAS_Registry	Mandatory
OB_AuthzActionName	uid	Mandatory
Cache	.	.
OB_AllUserCache_enabled	Enables caching of all users. Values are true and false.	Optional
OB_AllUserCache_timeout	Timeout for cache of list of all users.	Optional
OB_UserAttributesCache_enabled	Enables Caching of user attributes. Values are true and false.	Optional
OB_UserAttributesCache_timeout	The timeout for the cache of user attributes. Timeout is for the whole cache.	Optional
OB_UserAttributesCacheElement_ timeout	The timeout for the cached user attributes. The Timeout is per user.	Optional
OB_GroupAttributesCache_ enabled	Enables Caching of group attributes. Values are true and false.	Optional
OB_GroupAttributesCache_ timeout	The timeout for the cache of group attributes. Timeout is for the whole cache.	Optional
OB_ GroupAttributesCacheElement_ timeout	The timeout for the cached group attributes. The Timeout is per group.	Optional

Table 11–1 (Cont.) Parameters in NetPointWASRegistry.properties

Parameter Name	Description	Optional/ Mandatory
OB_AllGroupCache_enabled	Enables caching of list of all groups. Values are true and false. Used only for all groups, and mostly used by the Admin Console.	Optional
OB_AllGroupCache_timeout	The timeout for cache of the list of all groups.	Optional
OB_UserGroupsCache_enabled	Enables caching of list groups of which the user is a member. Values are true and false. Maintains a cache of all the groups a logged in user belongs to.	Optional
OB_UserGroupsCache_timeout	The timeout for cache of the list of groups for a user. The timeout is per user. This value should not be very high--if the user's group membership changes the new membership will only take affect at cache timeout. For example, a value of 3600 equates to 1 hour.	Optional
OB_GroupMembersCache_ enabled	Enables caching of list of groups and list of members in each group. Values are true and false. Stores members for each groups (not a frequently used cache).	Optional
OB_GroupMembersCache_ timeout	Specifies the timeout for cache of list of groups and the list of members in each group.	Optional
Keystore	.	.
OB_Keystore	Specifies the keystore file used by the registry when it makes SSL connections to HTTPS WebPass. The keystore contains the requestor's public and private key pairs, X.509 certificate, and certificates for Certificate Authorities trusted to certify responder servers. The keystore is managed using the JDK keytool. For example: *CWS_install_dir*/oblix/config/jssecacerts	Optional
OB_KeystorePassword	Optional. The password for the keystore.	Optional
Users and Groups	.	.
OB_UserTabId	For future use. Do not change the default. Default = Employees	Mandatory
OB_GroupTabId	For future use. Do not change the default. Default = Groups	Mandatory
Performance	.	.

Table 11–1 (Cont.) Parameters in NetPointWASRegistry.properties

Parameter Name	Description	Optional/ Mandatory
OB_NestedGroupsEnabled	Values are true and false. The default is true. To improve GroupSrvCenter performance when nested groups are not used, set the value to false. ▪ Nested groups will not be included in the search; the uniquemember attribute will not be requested in a group search when OB_NestedGroupsEnabled=false. ▪ A value of true retrieves the uniquemember attribute in the group search, uses this for nested group computation, then removes it before the group is recorded.	Optional
OB_DynamicGroupsEnabled	Values are true and false. To improve GroupSrvCenter performance when you are not using dynamic groups, set the value to false. Dynamic groups will not be included in the search.	Optional
Non-Unique Login ID in Different Domains	.	.
OB_DnIsUniqueIdentifier= See also OB_ DnIsUniqueIdentifier= in "WebGate.properties" on page 11-76.	Values are true and false. The default is false. Set the value to true to enable the Connector for WebSphere to work in a multi-domain directory server with a non-unique login ID in the different domains.	Optional

WebGate.properties

Table 11–2 describes the parameters of the webgate.properties file. This file is located in *CWS_install_dir* /oblix/config with a copy in *WAS_install_dir* \properties.

Table 11–2 Parameters in webgate.properties

Parameter	Description
OB_InstallDir	The directory where the NetPointWASRegistry is installed. Default =*CWS_install_dir*
OB_ISPROXYENABLED	Not required unless you use a proxy server. The default value is false. If you use a proxy server the value must be changed to true.
OB_hostnames	Not required unless you use a proxy server. The name of the host machine. This is only used for proxy servers.
Ob_loginID	Not required.
OB_AuthnSchemeResourceTypeName	Authen
OB_AuthnSchemeOperation	LOGIN
OB_AuthnSchemeResourceName	/Authen/Basic
OB_AuthzActionType	uid

Table 11–2 (Cont.) Parameters in webgate.properties

Parameter	Description
OB_DnIsUniqueIdentifier	Not required unless you want to enable the Connector for WebSphere to work in a multi-domain directory server with a non-unique login ID in the different domains. See also, Non-Unique Login ID in Different Domains in "NetPointWASRegistry.properties" on page 11-71.

TrustedServers.properties

Table 11–3 describes the parameters of the trustedservers.properties configuration file.

Table 11–3 Parameters in TrustedServers.properties

Parameter	Description
com.ibm.websphere.security.trustassociation.enabled	true
com.ibm.websphere.security.trustassociation.types	webgate
com.ibm.websphere.security.trustassociation.webgate.interceptor	com.oblix.tai.WebGateTrustAssociationInterceptor
com.ibm.websphere.security.trustassociation.webgate.config	webgate

Implementation Notes for the TAI

The following implementation is *optional* to enable the Connector for WebSphere to work in a multi-domain directory server with a non-unique login ID in the different domains.

To accomplish this optional implementation, you use two parameters in the NetPointWASRegistry.properties file:

OB_DnIsUniqueIdentifier=

The default is false. Be sure to set OB_DnIsUniqueIdentifier to true if the DN is being used.

To enable the Connector for WebSphere to work in a multi-domain directory server with a non-unique login ID in the different domains, you also need the following parameter in the webgate.properties file:

OB_DnIsUniqueIdentifier=false

This optional parameter is used when the TAI module is configured to pass on the users DN instead of the userAttr or LoginID. The default is false. If the OB_DnIsUniqueIdentifier parameter is set to true, the DN is used to communicate between the TAI and Registry. Be sure to set the OB_DnIsUniqueIdentifier to true if the DN is being used.

Note: The NetPointRegistry.properties and webgate.properties files must be synchronized.

The optional implementation described in the previous paragraphs works with the following caveats:

- Using TAI or WebGate as the only means of authentication should not be an issue since it is likely to be a requirement for multi-domain authentication. No users can go directly to WAS applications.

  Note: An exception is when logging into the Identity System Administration Console.

- A unique ID is used for the WAS_ADMIN Account across all domains.
- This solution causes a loss of functionality regarding mapping individual users to roles.

On WAS 5, role mapping can be done only through Identity System groups.

Implementation Notes for Active Directory

The following sections discuss issues to consider when implementing the Connector for WebSphere on Active Directory:

- Configuring the Connector for WebSphere for an Active Directory Forest
- Set Active Directory Domain in NetPointWASRegistry.properties

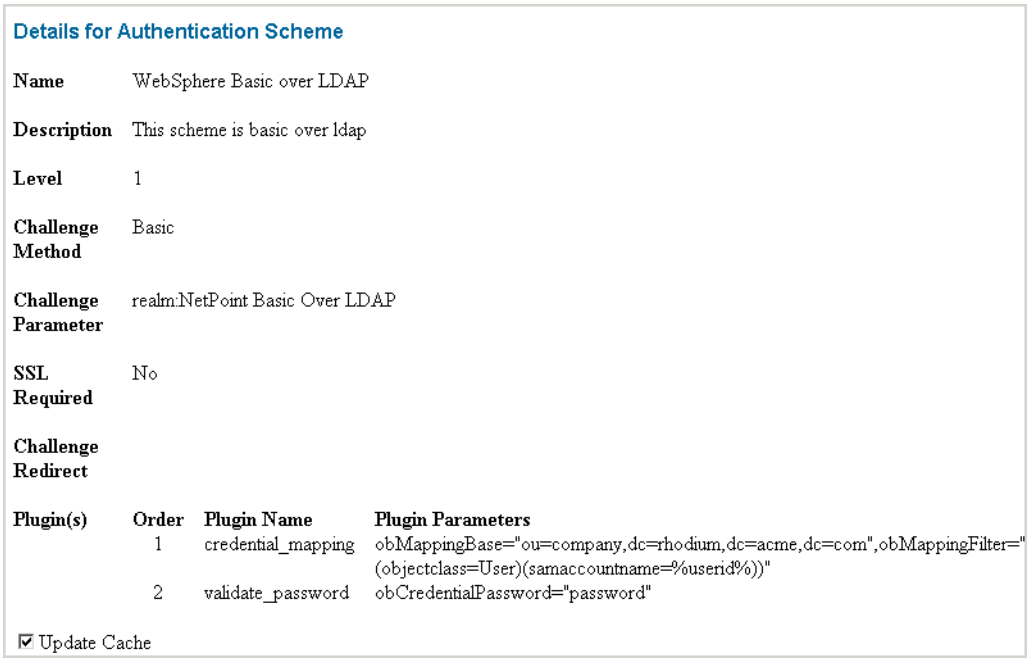
Configuring the Connector for WebSphere for an Active Directory Forest

The steps to configure the Connector for WebSphere for an Active Directory Forest follow.

To configure the Connector for an Active Directory forest

1. In the Access System Console, create a new Basic Over LDAP authentication scheme for a domain in the Active Directory Forest.

   The base credentials that you specify in the Plugin(s) field must be the same as the search base that you specified in the directory server profile.



Details for Authentication Scheme

Name	WebSphere Basic over LDAP
Description	This scheme is basic over ldap
Level	1
Challenge Method	Basic
Challenge Parameter	realm:NetPoint Basic Over LDAP
SSL Required	No
Challenge Redirect	

Plugin(s)	Order	Plugin Name	Plugin Parameters
	1	credential_mapping	obMappingBase="ou=company,dc=rhodium,dc=acme,dc=com",obMappingFilter="(objectclass=User)(samaccountname=%userid%))"
	2	validate_password	obCredentialPassword="password"

Update Cache

If you already created an administrator during pre-installation setup, you do not need to complete step 2. See "Preparing to Install the Connector" on page 11-11 for more information.

2. Create a WebSphere administrator in Oracle Access Manager with View and Delegated Administration rights.

   Ensure that the administrator's login identification is unique.

3. Specify the WebSphere administrator as the administrator for the Active Directory forest domain.

   This domain must be the same as the one for which you created the authentication scheme in Step 1. To do this, specify values for the OB_WebPassADDomain parameter in the NetPointWASRegistry.properties file as described in Table 11–1.

   You can search for users in the parent domain but you cannot search for users in sibling or children domains.

Note: You do not need to create an administrator for every domain in an Active Directory Forest.

Set Active Directory Domain in NetPointWASRegistry.properties

If you are running Active Directory using multiple domains, you must manually edit the NetPointWASRegistry.properties file to include a value for the OBWebPassADDomain parameter.

For example, OBWebPassADDomain=dc=xyz, dc=acme, dc=com

The domain must be the same as the domain defined for the default directory server in Oracle Access Manager.

See the *Oracle Access Manager Identity and Common Administration Guide* for more information.

#OB_UserAttr should be the Login Attribute example LoginID which is uid or genuserid.

Troubleshooting the Connector for WebSphere

The following is a list of the most frequently asked questions on the Connector for WebSphere. See also, "Troubleshooting the Connector for Portal Server v5" on page 11-90.

Problem

When I try to enable administrative or application security, I receive the following error:

```
You must supply the primary administrative user name on the active registry or
realm panels to enable security.

The server user ID or password is not specified. Enter a server user ID and
password for the active user registry, or select to use the automatically
generated server identity on the realm panel.
```

Solution

This message appears if you have not selected Standalone Custom Registry as the default realm when configuring the Secure Administration, Applications, and Infrastructure settings. See "To enable the NetPointWASRegistry in WAS 6.1" on page 11-41 for details.

Problem

I am trying to conduct a search of users or groups using WAS role mapping. I am supplying a search string but I am getting an error message.

Solution

WAS role mapping requires wildcards in all search strings. Put an asterisk (*) at the beginning and the end of your search string, for example, a search of "*user123*" returns all user IDs that have "user123" as a substring.

Problem

I am locked out of WAS 5 Admin Console or the WebSphere 5 Server does not start after making a configuration change. What should I do?

Solution

Restore the previous WAS 5.0 security.xml file located in WAS_install_dir/config/cells/serverName directory. This assumes you have made a backup of an older, working copy.

Problem

On Solaris, when setting up the SSL connector for the Connector for WebSphere, why does the keytool command give a "Signature not available" error?

Solution

This is a jdk 1.2.x problem. Use the NT version or any other jdk1.3.x version to create the cert db (jssecacerts) and then use it with WebSphere on Solaris.

Problem

Why do I get "All the jars are not in classpath: NoClassDefException"?

Solution

Make sure that the NetPointWASRegistry.jar and jobaccess.jar are in the classpath.

Problem

Why do I get an SSLPeerUnverifiedException - peer not authenticated exception?

Solution

The jvm being used is different from the jvm that has imported the certificates of ca and server. The jvm and keytools used must be from the same installation. If one keytool is used to add certificates and java is invoked from the other installation directory, then the jvm will not be able to use the certificates and will produce this exception.

Problem

Why do I get "ObConfig.NO_CONFIG_FILE"?

Solution

his error means that the Access Manager SDK client configuration file is not found. Check the Install_Dir parameter in the NetPointWASRegistry.properties file and ensure that the following points to the NetPointWASRegistry installation directory. For example:

```
# Installation directory of NetPointWASRegistry OB_InstallDir=/CWS_install_
dir/oblix/config/
```

Problem

Why do I get an UnsatisfiedLinkError?

Solution

You probably do not have the Access Manager SDK lib in the PATH or LD_LIBRARY_PATH depending on the platform.

For example:

- **On NT**: set the PATH as follows:

  ```
  set PATH=%PATH%;c:\CWS_install_dir\oblix\lib
  ```

- **On Solaris**: set LD_LIBRARY_PATH as follows:

  ```
  setenv LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/CWS_install_
  dir/oblix/lib export LD_LIBRARY_PATH
  ```

  This can be either done at system level or at start-up.

- **For AIX**: set LIBPATH as follows:

  ```
  setenv LD_LIBRARY_PATH=$LIBPATH:/CWS_install_dir/oblix/lib
  export LIBPATH
  ```

  You may also get the following error if you do not have the Access Manager SDK lib:

  "com.ibm.ejs.exception.InvalidUserRegistryConfigException: Custom [OSName]: com.oblix.registry.NetPointWASRegistry"

Problem

Why do I get NoClassDefFound error for com.oblix.access.ObAccessException?

Solution

Make sure that the jobaccess.jar file is in WebSphere's classpath.

Problem

Why do I not see the ObSSOCookie being set?

Solution

Make sure that you are using fully qualified domain names to access the WebSphere Server and the Web server that is running WebGate.

For example, use:

```
http://foobar.oblix.com:9080
```

Not:

```
http://foobar:9080
```

Check the Cookie domain for the following:

- Ensure that the Primary Cookie Domain set in the WebGate Configuration in Policy Manager.
- In the NetPointWASRegistry.properties file, ensure the values for these parameters are set as follows: OB_CookieDomain=.oblix.com, OB_CookiePath=/

Problem

I see the ObSSOCookie, but why is WebPass rejecting it?

Solution

Take the following steps:

- Ensure that the time on the machine on which WebGate and Webpass are installed is synchronized with the time on the WebSphere Server.
- Make sure that in the authentication schemes for WebGate, Webpass and the WebSphere Server resources have the same security level.

Problem

Why is WebGate rejecting the ObSSOCookie?

Solution

Ensure that the IPValidation parameter is set to Off.

From the Access System Console, click Access System Configuration, click AccessGate Configuration in the left navigation pane, click the link for the WebGate that protects the WebPass, and in the IPValidation field select the Off option.

Problem

Why do IdentityXML calls fail with an unauthorized exception?

Solution

Check the NetPointWASRegistry.properties file to ensure that the WebPass host name is fully qualified.

For example:

```
OB_WebPassHost=foobar.oblix.com

OB_WebPassPort=80
```

Problem

Why do I get a "server specific error 10" while restarting the WebSphere Administrative Server.

Solution

Ensure all Java process are killed before re-starting the WebSphere Administration Server.

Problem

I see the following exception:

```
Exception in thread "main" com.ibm.websphere.security.
CustomRegistryException: admin at
com.oblix.registry.NetPointWASRegistry.getUserDisplayName(NetPoi
ntWASRegistry.java:622) at
com.oblix.tools.registryTester.main(registryTester.java:69)
```

Solution

There can be many reasons for this exception. In the NetPointWASRegistry file, turn on the debug flag and check the debug log path as follows.

```
OB_LogLevel=debug

OB_LogFileName=/oblix/NetPointWASRegistry/log
```

Problem

I get the following error in the Oracle Access Manager log file:

```
Mon Jan 06 14:57:21 PST 2003: Error making SOAP request
java.io.FileNotFoundException:
http://cobalt.oblix.net:80/identity/oblix/apps/userservcenter/bin/
userservcenter.cgi at sun.net.www.protocol.http.
HttpURLConnection.getInputStream(HttpURLConnection.java:529)
at com.oblix.soapclient.OblixSoapClient.doRequest(OblixSoapClient.java,Compiled
Code) com.oblix.registry.NetPointWASRegistry.realGetUserDisplayName
(NetPointWASRegistry.java:650)
at com.oblix.registry.NetPointWASRegistry.getUserDisplayName(
NetPointWASRegistry.java:607)
at com.oblix.tools.registryTester.main(registryTester.java,Compiled Code).
```

Solution

Ensure that the IPValidation parameter is set to Off.

From the Access System Console, click Access System Configuration, click AccessGate Configuration in the left navigation pane, click the link for the WebGate that protects the WebPass, and in the IPValidation field select the Off option.

Problem

I get the following error in the Oracle Access Manager log file:

```
Mon Jan 20 15:37:24 GMT-06:00 2003: Error making SOAP request java.io.IOException:
Server returned HTTP response code: 401 for URL:
http://foobar.oblix.com:80/identity/oblix/apps/userservcenter/bin/
userservcenter.cgi
at sun.net.www.protocol.http.HttpURLConnection.getInputStream
(HttpURLConnection.java:604)
at com.oblix.soapclient.OblixSoapClient.doRequest(OblixSoapClient.java:285)
```

Solution

Be sure that the IPValidation parameter for the WebGate is set to Off.

From the Access System Console, click Access System Configuration, click AccessGate Configuration in the left navigation pane, click the link for the WebGate that protects the WebPass, and in the IPValidation field select the Off option.

Problem

I get the following exception:
`"com.ibm.ejs.exception.InvalidUserRegistryConfigException: User [username] not authenticated"?`

Solution

Make sure the OB_UserAttr, OB_UserSearchAttr and OB_GroupSearchAttr are set correctly in NetPointWASRegistry.properties.

OB_UserAttr=samaccountname

OB_UserSearchAttr=cn

OB_GroupSearchAttr=cn

Problem

I get the following error in the Oracle Access Manager log file:

```
java.lang.StringIndexOutOfBoundsException: String index out of range: -10 at
java.lang.String.substring(String.java(Compiled Code)) at
com.oblix.soapclient.OblixSoapClient.handleSoapResponse(OblixSoapClient.java:345)
at com.oblix.soapclient.OblixSoapClient.doRequest(OblixSoapClient.java:297) at
com.oblix.registry.NetPointWASRegistry.realGetUserDisplayName
(NetPointWASRegistry.java:650)
at com.oblix.registry.NetPointWASRegistry.getUserDisplayName
(NetPointWASRegistry.java:607)
at com.oblix.registry.NetPointWASRegistry.getUniqueUserId
(NetPointWASRegistry.java:680)
at com.ibm.ejs.security.registry.CustomRegistryImpl.createCredential
(CustomRegistryImpl.java:698)
at com.ibm.ejs.security.registry.CustomRegistryImpl.authenticate
(CustomRegistryImpl.java:166)
at com.ibm.ejs.security.registry.RegistryBean.authenticate (RegistryBean.java:109)
at
com.ibm.ejs.security.registry.EJSRemoteStatelessRegistry.authenticate(EJSRemoteSta
telessRegistry.java:25) at com.ibm.ejs.security.registry._Registry_
Stub.authenticate(_Registry_Stub.java:275) at
com.ibm.ejs.security.ltpa.LTPAServerObject.authenticate(LTPAServerObject.java:97)
at
com.ibm.ejs.security.util.LTPAAuthenticationCache.update(LTPAAuthenticationCache.j
ava:167) at com.ibm.ejs.security.util.Cache.get(Cache.java:114) at
```

```
com.ibm.ejs.security.util.LTPAAuthenticationCache.getCredential(LTPAAuthentication
Cache.java:82) at
com.ibm.ejs.security.SecurityServerBean.authenticateBasicAuthData(SecurityServerBe
an.java:145) at
com.ibm.ejs.security.EJSRemoteStatelessSecurityServer.authenticateBasicAuthData(EJ
SRemoteStatelessSecurityServer.j ava:49) at com.ibm.ejs.security._SecurityServer_
Stub.authenticateBasicAuthData(_SecurityServer_Stub.java:281) at
com.ibm.WebSphereSecurityImpl.SecurityServerImpl.authenticateBasicAuthData(Securit
yServerImpl.java:69) at
com.ibm.ISecurityLocalObjectLTPAImpl.PrincipalAuthenticatorImpl.authenticate(Princ
ipalAuthenticatorImpl.java:437) at
com.ibm.ISecurityLocalObjectBaseL13Impl.LoginHelperImpl.request_login_
controlled(LoginHelperImpl.java:1092) at
com.ibm.ISecurityLocalObjectBaseL13Impl.LoginHelperImpl.request_login_
controlled(LoginHelperImpl.java:827) at
com.ibm.ISecurityLocalObjectBaseL13Impl.CredentialsImpl.get_mapped_
credentials(CredentialsImpl.java:1206) at
com.ibm.ISecurityLocalObjectBasicAuthImpl.CredentialsImpl.get_mapped_
credentials(CredentialsImpl.java:188) at
com.ibm.ISecurityLocalObjectBaseL13Impl.VaultImpl.setServerCred(VaultImpl.java:386
2) at
com.ibm.ISecurityLocalObjectBaseL13Impl.PrincipalAuthenticatorImpl.setServerCred(P
rincipalAuthenticatorImpl.java:9 79) at
com.ibm.ISecurityLocalObjectLTPAImpl.PrincipalAuthenticatorImpl.authenticate(Princ
ipalAuthenticatorImpl.java:302) at
com.ibm.ISecurityLocalObjectBaseL13Impl.LoginHelperImpl.request_login_
controlled(LoginHelperImpl.java:1092) at
com.ibm.ISecurityLocalObjectBaseL13Impl.LoginHelperImpl.request_login_
controlled(LoginHelperImpl.java:827) at
com.ibm.ISecurityLocalObjectBaseL13Impl.CredentialsImpl.get_mapped_
credentials(CredentialsImpl.java:1206) at
com.ibm.ISecurityLocalObjectBasicAuthImpl.CredentialsImpl.get_mapped_
credentials(CredentialsImpl.java:188) at
com.ibm.ejs.security.SecurityCollaborator.getActualCredential(SecurityCollaborator
.java:999) at
com.ibm.ejs.security.SecurityContext.getActualCreds(SecurityContext.java:75) at
com.ibm.ejs.security.Initializer.bindServerIdToAdminApp(Initializer.java:458) at
com.ibm.ejs.security.Initializer.initialize(Initializer.java:217) at
com.ibm.ejs.security.Initializer.serverStarted(Initializer.java:133) at
com.ibm.ws.runtime.Server.fireServerStarted(Server.java:2001) at
com.ibm.ws.runtime.Server.fireServerStarted(Server.java:1994) at
com.ibm.ejs.sm.server.AdminServer.initializeRuntime0(AdminServer.java:1144) at
com.ibm.ws.runtime.Server.initializeRuntime(Server.java:884) at
com.ibm.ejs.sm.server.AdminServer.main(AdminServer.java:392) at
java.lang.reflect.Method.invoke(Native Method) at
com.ibm.ws.bootstrap.WSLauncher.main(WSLauncher.java:158)
```

Solution

Make sure the Identity Server is up and running.

Problem

Why does the Portal Server allow logins with old passwords even though it honors passwords updated through Oracle Access Manager?

Solution

Because the Portal Server installation sets the Security Cache Timeout to 600 seconds, old passwords will be stored in cache for that amount of time. This parameter is

present in the WebSphere Application Server Administrative Console - Security Center and under the General tab.

Problem

A deactivated Oracle Access Manager user can still access WebSphere resources.

Solution

To avoid this, ensure that all authentication schemes that WebSphere uses have the following lines added:

```
(|(!(obuseraccountcontrol=*))(obuseraccountcontrol=ACTIVATED))
```

Problem

Why does the WebSphere Portal Server not come up, resulting in security exceptions?

Solution

Ensure that the LPTA Token domain is set up correctly.

Problem

Single sign-on is not working when a URL resource is protected with a Basic Over LDAP authentication scheme, even though TAI is enabled.

Solution

Verify the you followed the steps described in "Configuring the TAI for WebSphere v5" on page 11-32, and that you have set the challenge re-direct field in the Basic Over LDAP authentication scheme.

Problem

The following error appears in the Event Viewer of the WebSphere Administrative Console:

```
CNTR0020E: Non-application exception occurred while processing method
isAllLevelNone on bean BeanId(admin#repository.jar#PmiService,
null):javax.transaction.TransactionRolledbackException: CORBA TRANSACTION_
ROLLEDBACK 0 No; nested exception is:
org.omg.CORBA.TRANSACTION_ROLLEDBACK: minor code: 0 completed: No
org.omg.CORBA.TRANSACTION_ROLLEDBACK: minor code: 0 completed: No
at com.ibm.ejs.jts.jts.JBrokerSupport$RI.client_unmarshalled_reque st
(JBrokerSupport.java:405)
at com.ibm.CORBA.iiop.RIs.iterateClientRequestPreRIs(RIs.java (Compiled Code))
at com.ibm.CORBA.iiop.ClientRequestImpl.reInvoke (ClientRequestImpl.java:851)
at com.ibm.CORBA.iiop.ClientDelegate.invoke (ClientDelegate.java:894)
at com.ibm.CORBA.iiop.ClientDelegate.invoke (ClientDelegate.java:409)
at org.omg.CORBA.portable.ObjectImpl._invoke (ObjectImpl.java:258)
at com.ibm.ejs.sm.agent._AdminAgent_Stub.invokeActiveObject (_AdminAgent_
Stub.java:39)
```

Solution

This may occur if the WebSphere Application Server startup time is long. There can be multiple reasons for this problem, including a startup servlet (load-on-startup = true) which requires long time performing the init() method for the servlet.

If the Ping Initial Timeout is set to a value lower than the amount of time needed for the App Server to start, the Ping Initial Timeout alarm expires before the App Server could come up fully and send the "serverIsAlive" message. As a result, the

administration server tries to kill and restart the Application Server process. In this situation, the state of the clone is recorded as Running.

The PMI client indicates that the clone is up and running and tries to invoke the "isAllLevelNone" method on the clone. Because the clone does not exist, it fails with an error message.

To correct this, set the Ping Initial Timeout to a larger number to allow the Application Server to start completely.

Problem

After enabling security and configuring LDAP as the Authentication Mechanism, the administration server restarts, the following errors show in the trace file:

```
[02.03.21 08:37:59:957 CST] 4be2cc Initializer W SECJ0007E: Error during security
initializationjava.lang.NullPointerException at
com.ibm.ejs.security.ltpa.LTPAPrivateKey.decode(LTPAPrivateKey.java:50) at
com.ibm.ejs.security.ltpa.LTPAPrivateKey.<init>(LTPAPrivateKey.java:40) at
com.ibm.ejs.security.ltpa.LTPAServerBean.updateAll(LTPAServerBean.java:106) at
com.ibm.ejs.security.Initializer.updateActiveLtpaConfig(Initializer.java:392) at
com.ibm.ejs.security.Initializer.propagateSecurityConfig(Initializer.java:296) at
com.ibm.ejs.security.Initializer.initialize(Initializer.java:173) at
com.ibm.ejs.security.Initializer.serverStarted(Initializer.java:129) at
com.ibm.ws.runtime.Server.fireServerStarted(Server.java:1977) at
com.ibm.ws.runtime.Server.fireServerStarted(Server.java:1970) at
com.ibm.ejs.sm.server.AdminServer.initializeRuntime0(AdminServer.java:1123) at
com.ibm.ws.runtime.Server.initializeRuntime(Server.java:882) at
com.ibm.ejs.sm.server.AdminServer.main(AdminServer.java:391) at
java.lang.reflect.Method.invoke(Native Method) at
com.ibm.ws.bootstrap.WSLauncher.main(WSLauncher.java:158)[02.03.21 08:38:00:001
CST] 4be2cc Server A WSVR0023I: Server __adminServer open for e-business
```

Solution

Sometimes the key and password get out of sync. Create a new key for WebSphere's use.

To create a new key, do as follows:

1. From the Security Center (4.0x) or the Global Security Wizard (3.5x) under the Authentication Tab, click Generate Keys.
2. At the prompt, enter a password for the new key.

Problem

A user's attempt to log in to the WebSphere Administrative Console results in the following error:

```
ADGU2009E: Security Error: Either username/password is wrong or
this user is not authorized to connect to admin server.
```

Solution

Only the following users can log in to the WebSphere Administrative Console when security is enabled:

- User defined in the security ID of the custom registry/LDAP.
- Users defined in Administrative roles.

Problem

I get a Not Authorized error when I try to access a WAS resource or a WebSphere Portal resource.

Solution

You get this error because the ObSSOCookie is not being sent. Refresh the page to send the ObSSOCookie.

It is recommended that you use a form-based authentication scheme to avoid this problem. If you use the basic authentication scheme, set the Challenge Redirect field to another WebGate to ensure that the ObSSOCookie gets sent.

Problem

If TAI is failing with a stack trace:

```
... 1493ff35 JaasLoginHelp E SECJ4001E: Login failed for
testeisintuser/Default Realm ....
```

You get this error because the ObSSOCookie is not being sent.

Solution

Do the following:

1. Refresh the page to send the ObSSOCookie.
2. Enable further security debugging for the following classes:

   ```
   com.ibm.ws.security.*

      com.ibm.websphere.security.*

         com.ibm.WebSphereSecurityImpl.*

         SASRas
   ```

3. To view detailed information on the runtime behavior of security, enable trace on the following components and review the output:

   `com.ibm.ws.security.*=all=enabled:com.ibm:WebSphereSecurityImpl.*=all=enabled:com.ibm.websphere.security.*=all=enabled`.

   This trace statement collects the trace for the security runtime:

   `com.ibm.ws.console.security.*=all=enabled`.

   This trace statement collects the trace for the security center GUI.

   `SASRas=all=enabled`.

   This trace statement collects the trace for SAS (low-level authentication logic):

   The logs should give better debugging messages, and ideas on what exactly is failing.

Problem

Error in the TAI logs:

Error no action found

Solution

Ensure that the Authentication Scheme level for the Authentication scheme protecting the WebSphere Policy (Authen/Basic) is less than or equal to the Authentication scheme level protecting the WebSphere resource.

Ensure that the WAS_REGISTRY action is set properly (see "Defining a Policy Domain for WebSphere" on page 11-17). Ensure that there is an Authorization Expression set and that upon Authorization success the WAS_REGISTRY action is set.

Problem

WAS Registrytester.bat may fail:

System variables may be picking up an older version of obaccess.dll based on the path.

Solution

Check your system variables to ensure these are correctly set. For example, $PATH and $CLASSPATH must be correctly set.

Problem

Unable to search users in the WebSphere Portal Admin page.

Solution

Do the following:

1. Log in as wpsadmin and go to Portal Administration, Security, Access Control List.
2. Click Get groups and users. Search for users using the wildcard character "*" and select wpsadmin and add it to the list, then click OK.
3. Select user groups in the Select the objects for the permissions for the user wpsadmin, then click Go.
4. Give Manage permissions for the group All authenticated users for user wpsadmin, then click Save.
5. Go to Users and Groups, Manage Users, then search for users using the wildcard character "*." to display all the users.
6. To view groups, repeat steps a, b, and c, then give Manage permissions for each group that needs to be viewed for wpsadmin and click Save

Problem

IBM WebSphere Portal Server gives an error "No Portlets to display"

Solution

Do the following:

1. Log in to WebSphere Administrative Console. Navigate to WebSphere Portal application.
2. Click the JVM settings tab and add the following to the System Properties"

   "HttpSession.RecurseThroughProxy", value = "true"
3. Restart the WebSphere Portal application.

Troubleshooting the Connector for Portal Server v5

The following Portal Server v5 issues are covered:

Portal Server v5 Installation-Related Issues

There are several Portal Server installation-related issues that you may encounter. See the following problem and solution descriptions.

Problem

Both the WebSphere AppServer and Portal Server installed successfully but not able to access Portal Server page i.e. wps/portal.

Solution

Check for the fix packs needs to be applied on Application Server for the respective Portal Server version integration. The order to apply fix packs is important.

For Example, In case of Application Server 5.0 and Portal Server 5.0, the Fix Pack 1 needs to be applied on WebSphere Application server. The order of applying patch would be,

- Wasfp1
- Pmefp1
- Fixes
- Manualfixes

You can check $WAS_install_dir/properties/version/history folder to confirm the list of fixes applied on Application server.

For more information on this, see the WebSphere Portal infocenter document available on the IBM WebSphere site.

Problem

On login to the Portal Server (wps/portal), the following error message displayed:

There has been an application error! Please contact to your system administrator to report this error.

Solution

Ensure the required fix pack has been applied on the Application Server. See the first problem in this section. Also, ensure the Java 2 security is disabled in the WebSphere Application server.

To disable Java 2 security

1. Login into WebSphere Application Server Console, Security, Global Security.
2. Disable the check box for the Enforce Java 2 Security option.
3. Save the changes, then restart both the WebSphere Application Server and Portal Server.

Note: It is recommended that you install Portal Server with disabling Global Security in WebSphere Application server.

Problem

On successful login to the Portal Server, the Portal Admin is not able to view the Administration link to manage the Portal Server. This link is displayed on the top-right side of the page.

The Administration portlet is not deployed on the system.

Solution

Do the following:

1. Run WPSconfig.{bat | sh} portlets script to deploy the same.

   This script is available in $WPS_install_dir/config folder. For more information check the wpsconfg.xml file.

2. Restart the Portal Server and login using admin credentials.

   The Portal Admin user will able to see the "Administration" link and he can setup the Portal Server Application.

> **Note:** The same script will also deploy some sample portlet applications come along with Portal Server installation. Also, WPS_ install_dir is the directory where the Portal Server is installed.

Problem

On successful login to the Portal Server, Portal Admin is not able to view a portlet page. The error message appears as:

There is no content available. Please check if there is content defined for the markup of your client device.

This message comes because portal page is not created in the Portal Server.

Solution

A page has to be created in the Portal Server so you can add portlets on this page.

To add a page, you need page addition/modification privileges and a portlet needs to be installed that adds a new page or edits a page.

See issue 3 to check whether the required portlet has been deployed on the system.

Problem

When searching users from the Portal Server "user search" interface you may get empty records as a search result. This problem occurs when user names stored in the directory server are in Latin ISO-8859-1 style.

Solution

Ensure that you unset the following environment variables.

```
unset LC_COLLATE LC_CTYPE LC_MESSAGES LC_MONETARY LC_NUMERIC LC_
TIME
```

Custom Security Integration Related Issues

The following issues are related to custom security integration.

Problem

Portal Server configuration has been done for a custom member repository (WAS Connector) but the user is still not able to log in into the Portal Server using Custom Data Store user id, that is, the user is present in Oracle Access Manager.

Solution

Check $WPS_install_dir\shared\app\wmm\ wmm.xml file. Confirm that ProfileRepository tag has been configured correctly for Custom Member Repository implementation.

Sometimes WPSconfig script execution modifies wmm.xml file and LDAP settings replace custom member repository settings.

Note: Oracle recommends that you always keep backup of wmm.xml file configured for Custom Member Repository.

Problem

Portal Admin is able to search for the user present in Custom Data Store but on assigning him a portlet access permission, that user is not able to view the portlet after successful login.

This problem occurs when user DN present in the Custom Data Store contains intermediate spaces.

For Example, cn=Portal User, o=company, c=us.

WebSphere Portal does normalization of DN before matching it with the allowed user DN present in its internal cloudscape database. As string entries do not match, access permissions to the user fails.

Solution

To overcome this problem, apply Fix Pack PQ93461 provided by IBM.

Problem

How can I retrieve only the first 10 entries (or the number of entries I want to retrieve) on a search for an Identity system user or group through the Portal Server.

Solution

he parameter of number of records to retrieve is configurable in NetpointWASRegistry.properties file.

The name of parameter is OB_WebPassXPIRecordsReturned.

If this parameter is not defined or set to zero then it will retrieve all the users or groups present in the Oracle Access Manager Repository.

Problem

Portal Admin user is not able to install a portlet.

This problem is related to invalid deployment credentials in Portal Server.

Solution

Verify that WPSConfig.{bat/sh} action-create-deployment-credentials script has been executed or not.

This script execution creates required Credential Vaults in portal server.

Before execution of this script, verify wpsconfig.properties file. Check values of all the configuration parameters mentioned in the *Oracle Access Manager Installation Guide*.

Also verify that *$WPS_install_dir*\ shared\app\config\services\VaultService.properties contains correct systemcred.dn value. This should be Portal Admin User DN.

After successful execution verify that the credentials vaults has been created for the Portal Server.

You can check the same from Portal Server Login, Administration, Access, Credential Vault, Manage System Vault Slots.

Problem

How can I get the debug logs of the Oracle Access Manager WAS Registry and CMR?

Solution

For WAS Registry logs, set the following parameters of NetpointWasRegistry.properties file:

OB_LogLevel=debug

OB_LogFileName=log file complete path

For enabling CMR logs and Portal Server logs set the following parameters in $WPS_ install_dir\ shared\app\config\log.properties file.

traceString=*=all=enabled

com.ibm.wps.services.puma.*=all=enabled:

com.ibm.wps.puma.*=all=enabled:

com.ibm.wps.command.puma.*=all=enabled:

com.ibm.wps.engine.commands.*=all=enabled:

com.ibm.wps.services.authentication.*=all=enabled:

com.ibm.ws.security.*=all=enabled:

com.ibm.websphere.security.*=all=enabled

All the puma services logs will be generated in the "Trace.log" file.

Refer Portal Server infocenter document for more details on turning on logs.

Problem

How can I stop/start WebSphere Application Server and Portal Server?

Solution

To start and stop Application Server use

- Run the startServer command, as follows:

  $WAS_install_dir/bin/startServer *name_of_app_server*

  (Default name is server1)

- Run the stopServer command, as follows:

  $WAS_install_dir/bin/stopServer *name_of_app_server* -username *WAS_Admin_userid* -password *WAS_Admin_Pwd*

To start and stop Portal Server use

- Run the startServer command, as follows:

  $WAS_install_dir/bin/startServer *name_of_portal_server*

  (Default name is WebSphere_Portal)

- Run the stopServer command, as follows.

  $WAS_install_dir/bin/stopServer *name_of_portal_server* -username *PortalAdmin* -password *PortalAdminPwd*

Problem

Portal Admin is not able to search the users present in the Oracle Access Manager Repository but these users are able to log in into the Portal Server.

Solution

Solution: Check the user.fbadefault.filter parameter in the PumaService.properties file. This parameter should contain the attribute name, which is passed to the custom CMR implementation. The WAS Connector uses this attribute to ensure the Portal Admin user is a valid user.

Check the trace.log file for exception details.

Problem

While starting the Portal Server, the CMR gets invalid Portal Administrator credentials.

Solution

Run the Wpconfig. {bat/sh} action-secure-portal-ldap script from the *$WPS_install_dir*/config folder.

Problem

Why am I unable to install a portlet using the administrator ID with Oracle Access Manager security enabled?

Solution

Solution: Complete the following steps to ensure that the proper credentials are being used.

1. Verify that the correct values for WasUserid and WasPassword are present in the following file:

   wpconfig.properties

   WasUserid must be the administrative user id and not the administrative DN. As necessary, use any plain-text editor to open wpconfig.properties and correct the values for WasUserid and WasPassword. Save and close wpconfig.properties.

2. Confirm that you have executed the following command:

   wpconfig.bat\.sh action-create-deployment-credentials

   If you discover that you have not run the command, execute it, then proceed with portlet installation.

   If you discover that you previously ran the script using incorrect values for WasUserid or WasPassword, correct the invalid values by executing the following two commands:

   wpconfig.bat\.sh action-remove-deployment-credentials

wpconfig.bat\.sh action-create-deployment-credentials

3. Verify that the administrator id is defined correctly by navigating to Portal Administration, Access, Credential Vaults, Manage system vault slots, deployment.user.Vault.

Problem

For WebSphere 6.0, no users are returned when a search is conducted through the WebSphere Application Server Administration Console.

Typically, the Oracle Access Manager log will contain an error message that begins with the following:

```
Error making SOAP request . . .
```

Solution

Make sure that $LANG and all $LC_* variables are set to en-US. You can check the current values of these variables through the "locale" command on SLES 9.

Problem

The Client Cert Authentication feature for the WebSphere Application Server 6.0 is unable to access the Snoop applet on the https port of the Web server.

Solution

Ensure that the SSL port used by the Web server has been added to the "default_host" virtual host configuration on the Application Server. If it does not exist, perform the following steps:

1. Launch the WebSphere Administration Console.
2. Navigate to Environment, Virtual Hosts, default_host, host aliases.
3. Add the hostname * Port: 443 (This is the SSL port used by the Web server).
4. Save your changes.
5. Regenerate the plugin by completing the following steps:
   a. Navigate to Servers, Web servers.
   b. Select the Web server for which you will create the plugin.
   c. Select Generate Plugin.

      You can view the generated plugin by navigating to: *WebServerName*, plugin properties, View, *PlugInFileName*.
   d. Restart the Web server.
6. To verify that the SSL port is operational, complete the following steps:
   a. Disable the Access System policy protecting the test resource.
   b. Attempt to access the test resource using WebSphere Authentication exclusively.

      Use the following URL:

      `https://WebServerHostMachineName:sslPortNumber/snoop`
   c. If this succeeds, re-enable the Access System policy.

12

Integrating Plumtree Corporate Portal

This chapter provides an overview of integrating Oracle Access Manager with Plumtree Corporate Portal. Note that this product is now called BEA Aqualogic Interaction, however, the currently certified integration is with the Plumtree Corporate Portal product.

This chapter covers the following topics:

About the Integration with Plumtree Corporate Portal

The integration between Oracle Access Manager and Plumtree Corporate Portal provides companies with a Web enterprise solution for building customized, secure business portals with integrated, identity-based Web access management.

In the integrated solution, Plumtree Corporate Portal acts as a gateway to an enterprise intranet or extranet, providing users centralized access to a broad variety of applications and content hosted by the enterprise.

Oracle Access Manager provides a robust identity management and access security system to accurately track and manage the identities of Plumtree Corporate Portal's employees, customers, and partners. Oracle Access Manager also provides a common enterprise security and user identity infrastructure that controls access to Plumtree Corporate Portal as well as to other enterprise applications and resources.

The integration supports single sign-on (SSO) between the applications within the portal framework and the enterprise Web applications that are secured by the Access System.

This integration offers these major benefits:

- **Single Sign-On**: The Access System's single sign-on services offer authorized users a secure connection with minimal authentication challenges to the resources that they need. Users need to log in only once to gain access to all resources that they need at a given level of authentication. The access provided by Access System's single sign-on services improves user efficiency, productivity, and user satisfaction.

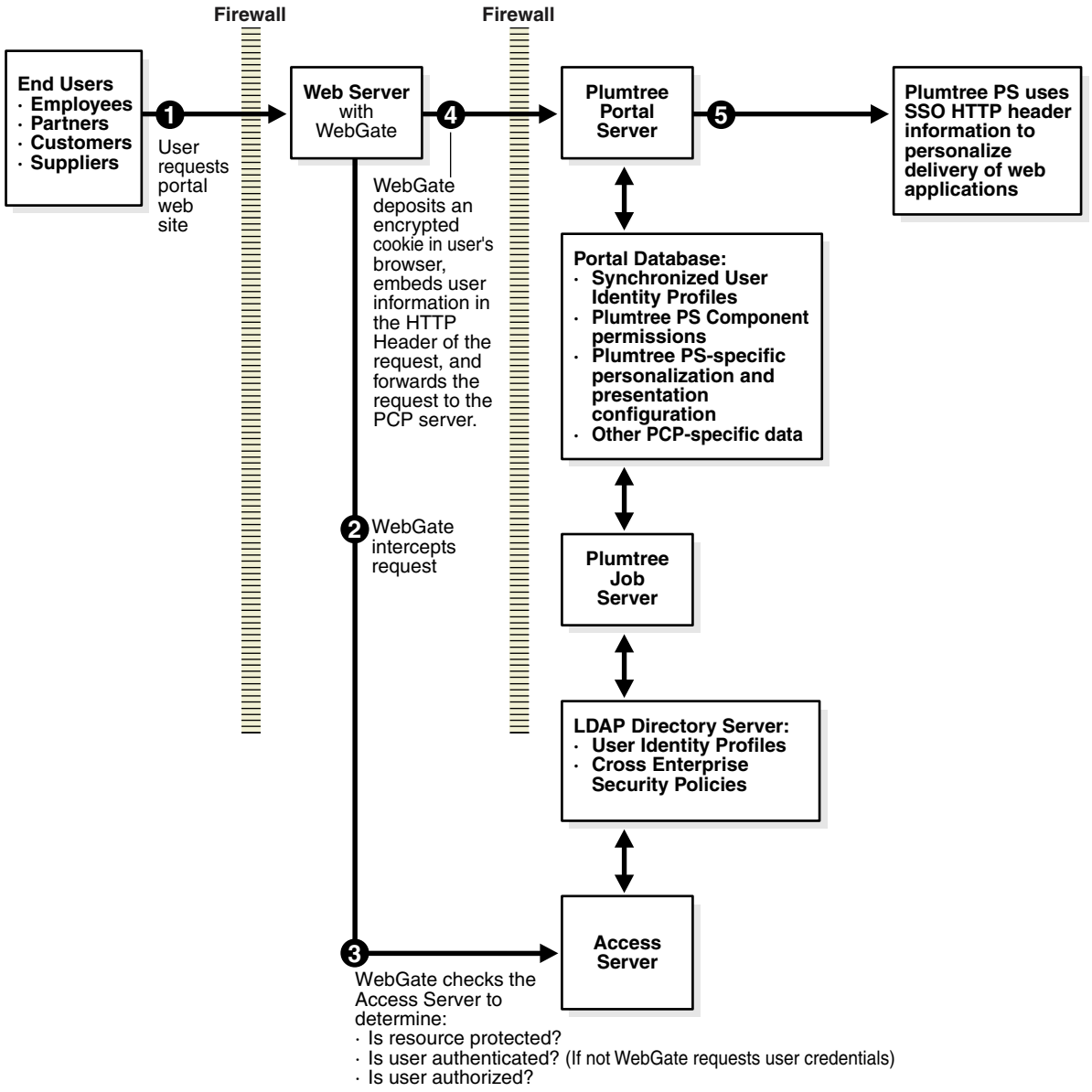
- **Delegated Administration**: Oracle Access Manager offers delegated administration capabilities that complement those of Plumtree Corporate Portal. Oracle Access Manager's delegated administration distributes responsibilities for managing identity and security policy information. Users can update some of their information; managers, suppliers, and partners can set up, change, and delete user identity information.
- **User and Group Management**: Oracle Access Manager manages an LDAP directory server that contains user identity information. Plumtree Corporate Portal synchronizes its own user database with the LDAP directory. This eliminates the need to create and manage separate identity profiles for each application.

  Oracle Access Manager also provides strong group management capabilities, including support for static, dynamic, nested, and hybrid groups. The Identity System manages the group information that the Plumtree Corporate Portal application uses to personalize the portal. Group affiliation is immediately reflected in the portal without manual changes.
- **Centralized Security Management**: Oracle Access Manager provides a common security management platform for all applications within an enterprise. This facilitates the maintenance of consistent security policies across an entire enterprise.
- **User Personalization**: Oracle Access Manager enables you to personalize portal content based on any user attribute, such as job title.

This integration does not change the users' experience in Plumtree Corporate Portal. Users can continue to access the portal guest pages without logging into Plumtree Corporate Portal. When a user attempts to log in, the Access System intercepts the request and uses an authentication scheme to determine whether the user is authorized to access the portal. Those who are not authorized are denied access to Plumtree Corporate Portal.

Figure 12–1 illustrates how Oracle Access Manager components protect the Plumtree Corporate Portal.

Figure 12–1 Overview of Plumtree integration



The following summarizes the integration.

Task overview: Integrating with Plumtree

1. Enabling single sign-on for the Plumtree Corporate portal as described in "Enabling Single Sign-on in PlumTree 5.0.4" on page 12-4.

   In the Plumtree installation, you set up single sign-on and LDAP authentication sources, edit configuration files to support single sign-on, and then synchronize data from the Oracle Access Manager LDAP directory with the data in the Plumtree database.

2. Setting up the Access System to protect the Plumtree Corporate portal as described in "Setting Up the Access System to Protect Plumtree 5.0.4" on page 12-8.

   In the Access System, you create policies that specify the content that you want to protect. Policies are created in the Policy Manager.

3. (Optional.) Allow anonymous users to view the portal guest pages and creating a banner for the portal as described in "Integrating Other Features" on page 12-11.
4. (Optional.) Personalize user pages, and embed Identity System identity management functions as described in "Integrating Other Features" on page 12-11.

Supported Versions and Platforms

You can find support and certification information at the following URL:

`http://www.oracle.com/technology/documentation/`

You must register with OTN to view this information.

Also, you can see the supported versions and platforms for this integration on Metalink, as follows.

To view information on Metalink

1. In your browser, enter the following URL:

   `https://metalink.oracle.com`
2. Log in to MetaLink.
3. Click the **Certify** tab.
4. Click **View Certifications by Product**.
5. Select the **Application Server** option and click **Submit**.
6. Choose **Oracle Identity Management** and click **Submit**.
7. Click **Oracle Identity Management Certification Information 10g (10.1.4.0.1) (html)** to display the Oracle Identity Management page.
8. Click the link for **Section 6, Oracle Access Manager Certification** to display the certification matrix.

Enabling Single Sign-on in PlumTree 5.0.4

This section describes the following tasks for enabling single sign-on on the Plumtree portal:

- Creating a Single Sign-On Authentication Source
- Creating an LDAP Authentication Source
- Editing Configuration Files to Support Single Sign-On
- Synchronizing LDAP Data with Plumtree Database
- Enabling Single Sign-On Logout

Creating a Single Sign-On Authentication Source

Authentication is the process of users proving their identity to a server. After users present their credentials to the server, authentication plug-ins process those credentials. To enable the Access System to authenticate users and groups on Plumtree, you must create a single sign-on authentication source so that the Access System can authenticate users and groups in the Plumtree portal. To do this, you must first create a single sign-on password to use when you configure the single sign-on authentication source.

To create a single sign-on password

1. Launch the Plumtree Administrator Control Panel application and navigate to **Start**, select **Settings**, select **Control Panel**, then click **Plumtree Administrator**.

   The Plumtree Administrator dialog box appears.

2. Click **Single Sign-On**.

3. Enter an SSO secret key.

   Note: This secret key can be any string of characters. Make note of the string.

4. Click **OK** to close the Plumtree Administrator dialog box.

To create a single sign-on authentication source on Plumtree

1. Log in to Plumtree as the administrator. Click **Administration**. The Administration Menu appears.

2. Click **Authentication Folder**. In the **Create Object** drop down box select **Authentication Source-SSO**. The Authentication Source Wizard appears.

3. Enter the single sign-on password.

4. Click **Validate Options** to confirm that this password matches the secret key you entered in the Plumtree Administrator Control Panel.

5. On the **Properties and Names page**, enter a name and description for the new authentication source. Describe the source carefully, as this description appears in a list during Authentication Source setup.

6. Specify the properties for the Authentication Source.

Creating an LDAP Authentication Source

To update the Plumtree database with the current user and group information, you must synchronize users and groups in the Plumtree database with information in the LDAP directory. To do this, you must first create an LDAP Authentication Source to import users and groups data from the LDAP directory into your Plumtree portal. To create an LDAP authentication source on Plumtree.

1. Log in to Plumtree as the administrator and click **Administration**.

   The Administration Menu appears.

2. Click **Authentication Folder**. In the Create Object drop down box, select **Authentication Source-SSO**. The Authentication Source Wizard appears.

3. In the Authentication Source Category box, type the prefix used to distinguish the users and groups imported from this domain. For example, if you enter myDomain, each user name and each group name will be prefixed by the string myDomain. Thus, myUser becomes myDomain\myUser and myGroup becomes myDomain\myGroup.

   You can set the category to any value you want, but after you create this authentication source, you cannot change this value.

4. Templates can populate configuration options with common default values. To use an existing template to populate the values in this editor, choose one from the Template list.

5. Add values under LDAP Settings to facilitate portal access to the LDAP server from which you are importing users. Consult online help for instructions.
6. To confirm the domain you entered, click **Validate Options**.

   After the portal has attempted to find the domain, it displays a message stating whether it connected successfully or not.

   - If the validation fails, check for errors in your settings.
   - If the validation succeeds, go to the **Synchronization** page.
7. Select the single sign-on authentication source you created earlier from the list labelled **Select an Authentication Partner**. Select Full Synchronization.
8. On the **Schedule** page, select **Create a Job/Add a Job for this Authentication Source**. Consult online help for instruction on creating or adding a job.
9. Add appropriate properties on the Properties and Names page.

Editing Configuration Files to Support Single Sign-On

By default, the portal expects Oracle Access Manager to forward the user name header named uid. If you configure Oracle Access Manager to forward a user name header with a different name, you must configure your single sign-on implementation as BasicSSO service. For details about BasicSSO service, see the PlumTree Administration Guide.

Configure elements under the <Authentication> parent element in the `PTConfig.xml`, as detailed in Table 12–1.

Table 12–1 Parameters In Ptconfig.xml

Parameter	Value
SSOVendor	For Oracle Access Manager the SSOVendor value is 3.
DefaultAuthSourcePrefix	Use the same value you specified for Authentication Source Category when you configured your authentication source.
CookiePath	" /" (Specify a different setting only if your single sign-on authentication server requires a different convention.)
CookieDomain	Specify the fully qualified domain name to which you want the cookie forwarded. For example, if you specify `.company.com`, the cookie enables access to all domains that end in `.company.com`. If you specify `.sub.company.com`, the cookie enables access only to domains that end in `.sub.company.com`. The string must start with a period (.) and include a minimum of two periods.
SSOCookieIsSecure	0 or 1. 0 (the default) specifies that the connection to the remote server does not require SSL for the cookie to be forwarded. 1 specifies SSL is required.

The following example enables integration with an Oracle Access Manager authentication server:

```
<SSOVendor value="3"/>
<DefaultAuthSourcePrefix value="HQ"/>
<CookiePath value="/"/>
<CookieDomain value=".company.com"/>
<SSOCookieIsSecure value="0"/>
```

Synchronizing LDAP Data with Plumtree Database

Data synchronization ensures the updating of information on group memberships and users, which is crucial for portal access and personalization. You can synchronize data manually or automatically at specific time intervals. To synchronize manually, you schedule a job and run it once for immediate synchronization. For periodic synchronization, you schedule a job to run at specified time intervals.

After you run a job, you can view its status to see if it ran successfully. When a job has run successfully, you can view the replicated LDAP data in the Plumtree database.

Caution: Initial full synchronization requires a long time if you have many entries.

You must fully synchronize at least once to enable single sign-on between Plumtree and Oracle Access Manager.

To automatically synchronize data

1. On the Plumtree portal host, navigate to **Control Pane**l, then **Services**, and verify that the Plumtree Job Dispatcher service has started.
2. Log in to Plumtree as the administrator, then click **Administration**.
3. In the **Administration** Menu, navigate to the Administration folder, click **Jobs**, then select the job you earlier created / added for the LDAP Authentication Source. The main settings page appears.
4. Specify the time period when the synchronization job is to be run.
5. The Edit Job User page also provides options for checking the job log and history through the Job History tab. It also lets you check security-related options.

To manually synchronize data

1. On the Plumtree portal host, navigate to **Control Panel**, then select **Services**, and verify that the Plumtree Job Dispatcher service has started.
2. Log in to Plumtree as the administrator, then click **Administration**.
3. In the **Administration** Menu, click the **Administration folder**, select **Jobs,** then select the job you earlier created / added for the LDAP Authentication Source. The main settings page appears.
4. Select **Run Once - Now** option in the settings. This job runs once, starting immediately.

Viewing Synchronized Information

After synchronization has been completed, you should be able to see all the users and groups from the LDAP data source replicated in the Plumtree database.

To view the updated Plumtree database

1. In the portal, click **Administration**.
2. In the **Administration** Menu, click your administration folder.
3. Expand the **Users** tab. The user names will be prefixed with the LDAP source name.

For example:

```
LDAPUsers\Accounting Managers
```

In a Group folder, you can also view the members of the groups.

Enabling Single Sign-On Logout

When a user clicks the PlumTree Log Off button in an Access System-protected PlumTree session, users are logged out from PlumTree, but the ObSSOCookie generated for the users is not killed. Hence, the Oracle Access Manager session for that particular user remains active. Users need to customize the PlumTree logout functionality to facilitate logout from PlumTree as well as Oracle Access Manager.

Consult "Customizing the Portal UI: Using Plumtree Event Interfaces (PEIs)" section in "Enterprise Web Development Documentation" on the PlumTree Web site for information on modifying the PlumTree UI. Users need to implement the "OnBeforeLogout" interface as:

```
public virtual Redirect OnBeforeLogout(Object _oUserSession, ApplicationData
_appData)
    {
          PTDebug.Trace(Component.Portal_UI_Infrastructure, TraceType.Error,
"Before logout event");
          String myCookie = _appData.GetCookie("ObSSOCookie");

          if(myCookie != null)
          {
               Redirect myredirect = new Redirect();
              myredirect.SetLinkToExternalURL("http://<AccessManager_Server_
Name:port>/access/oblix/lang/en-us/logout.html");
               return myredirect;
          }
          return null;
    }
```

Setting Up the Access System to Protect Plumtree 5.0.4

Typically, users click the Login button to log in to the Plumtree Corporate Portal. Once they are authenticated, they can view their personalized pages. To log out of the portal, users click the Logout button. In an alternate configuration, you might want all users to see a guest portal, then authenticate users only when they log in to the portal. (If you have enabled user access to the guest pages, then any user can go the main portal page and view those guest pages without ever logging into the portal).

When a user attempts to login, the Access System authentication policy challenges the user. Once the Access System authenticates the user, it checks to see if the user is authorized. If the user is authorized, an ObSSOCookie and a header variable are sent to enable single sign-on into Plumtree. The user is then logged in to Plumtree.

You can configure the Plumtree Corporate Portal so that when users log out, they are automatically logged out of both Plumtree and Oracle Access Manager. To set up the Access System's single sign-on service for Plumtree, after you have installed the Access System, you must create policies in the Policy Manager that specify the content you want to protect.

Setting up the Access System to protect Plumtree consists of the following tasks:

- Installing Oracle Access Manager Components
- Creating a Policy Domain

- Configuring the WebGate
- Configuring WebGate for IIS

Installing Oracle Access Manager Components

To integrate with Plumtree, you must install the following applications:

- Identity Server
- Access Server
- WebPass
- Policy Manager
- WebGate

To enable the Access System to protect the portal, install a WebGate on the Plumtree Corporate Portal Web server.

Note: You can install the Identity server, WebPass, and Access System on the same server. However, Oracle recommends that you do not install the Identity Server and WebPass on the server where Plumtree has been installed. For more information on installing and configuring Oracle Access Manager, see the *Oracle Access Manager Installation Guide*.

Creating a Policy Domain

To create the policy domain, follow the procedure in "Creating a Policy Domain" on page 12-9.

For more information on policy domains, see the *Oracle Access Manager Access System Administration Guide*.

To create the policy domain

1. Install Oracle Access Manager, which includes the following components: Identity Server, WebPass, Access Server, and Policy Manager.
2. Launch the Access System, as follows:

   `http://Access_Server_install_dir:port/access/oblix`.
3. Click the link for the **Policy Manager** application.
4. In the Policy Manager, create a policy domain.

   In the **Resources** tab for this policy domain, select a backslash ("/") as the URL prefix and type the path to the SSO servlet in the adjacent text box.

   For example: `portal/SSOServlet` or portal/sso/ssologin.aspx
5. Click the **Authorization Rules** tab for this policy domain and enter basic information for the new authorization rule.

   Click the **Actions** sub-tab for the authorization rule and add the following action:

   - In the **Type** field, enter a descriptive name, for example, headerVar.
   - In the **Name** field, enter a descriptive name, for example, UID.
   - In the **Return** field, enter the name of an attribute that is used by the authentication source to map to the user name in the user directory.

For example, iPlanet LDAP uses the uid attribute by default. Other LDAP directories, including Active Directory, use cn by default.

Configuring the WebGate

Configure your WebGate following the procedure appropriate for your portal deployment in the *Oracle Access Manager Installation Guide*:

- WebGate for Apache
- WebGate for IIS

Use the version of the WebGate that matches the version of your Oracle Access Manager.

You can find support and certification information at the following URL:

`http://www.oracle.com/technology/documentation/`

You must register with OTN to view this information.

Also, you can see the supported versions and platforms for this integration on Metalink, as follows.

To view information on Metalink

1. In your browser, enter the following URL:

   `https://metalink.oracle.com`

2. Log in to MetaLink.
3. Click the **Certify** tab.
4. Click **View Certifications by Product**.
5. Select the **Application Server** option and click **Submit**.
6. Choose **Oracle Identity Management** and click **Submit**.
7. Click **Oracle Identity Management Certification Information 10g (10.1.4.0.1) (html)** to display the Oracle Identity Management page.
8. Click the link for **Section 6, Oracle Access Manager Certification** to display the certification matrix.

To set up the WebGate for Apache

1. On the host computer for the Portal Server, install the version of Apache required by the WebGate.

   Note: The version of Apache provided by Plumtree and described in the Installation Guide for Plumtree Corporate Portal cannot be used with the WebGate. You must download the required Apache version from the Apache Web site.

2. On the host computer for the Portal Server, install the WebGate for Apache. For details, see the*Oracle Access Manager Installation Guide*.
3. On the Web application server to which the portal application is deployed, modify the Web application server setting to turn off URL rewrites. For details, refer to your Web application server documentation or Plumtree Knowledge Base article DA_239501, "Configuring Web Application Servers to not Rewrite URLs."

Configuring WebGate for IIS

Install the version of the WebGate that matches your Oracle Access Manager.

To set up WebGate for IIS, run the WebGate for IIS installer on the host computer for the Portal Server.

Integrating Other Features

Oracle Access Manager offers several other features that you can integrate with Plumtree such as allowing guest users to view portal pages, personalizing user pages, and embedding other identity management functions into your portal. You can also set up single sign-on to other portals, and manage passwords and self-registration.

The following tasks are discussed in this section:

- Enabling Anonymous Users to View Portal Guest Pages
- Using the Knowledge Directory
- Password Management
- Self-Registration

Enabling Anonymous Users to View Portal Guest Pages

You can allow anonymous users to access guest pages without logging into the portal. To do this, you must create a policy domain in the Access System that uses the Anonymous authentication scheme for the Oracle Access Manager Anonymous user. The Anonymous authentication scheme is a default that is supplied with the Access System.

This allows users to go to the main portal page and view guest pages without being challenged by the Access System.

You then lock the NetPointAnonymous account to allow anonymous users to view the guest pages without logging into the Plumtree portal. When a user goes to the main portal page but does not log in, the Access System logs in the user as NetPointAnonymous and creates an ObSSOCookie for this anonymous user. The ObSSOCookie is sent to the Plumtree portal but the portal treats the user as a guest because the account is locked. Thus, the user can view guest pages.

To create a policy domain for guest access

1. Launch the **Policy Manager** and click **Create Policy Domain**.
2. In the **General** tab, enter a name and description for the new policy domain.
3. In the Resources tab, select **HTTP** as the resource type.
4. To use a host identifier, create a host identifier in the **Access System Console**.

   You must add a fully qualified host name as one of the host name variations; for example, plum1.oracle.com.

   See the *Oracle Access Manager Access System Administration Guide* for more information.
5. Select the portal URL prefix from the list or create a new one. For example, /portal.
6. In the Default Rules tab, click **Authentication Rule** and from the **Authentication Scheme** list, select **Anonymous Authentication**.

This enables a user to view guest pages without logging in to Plumtree.

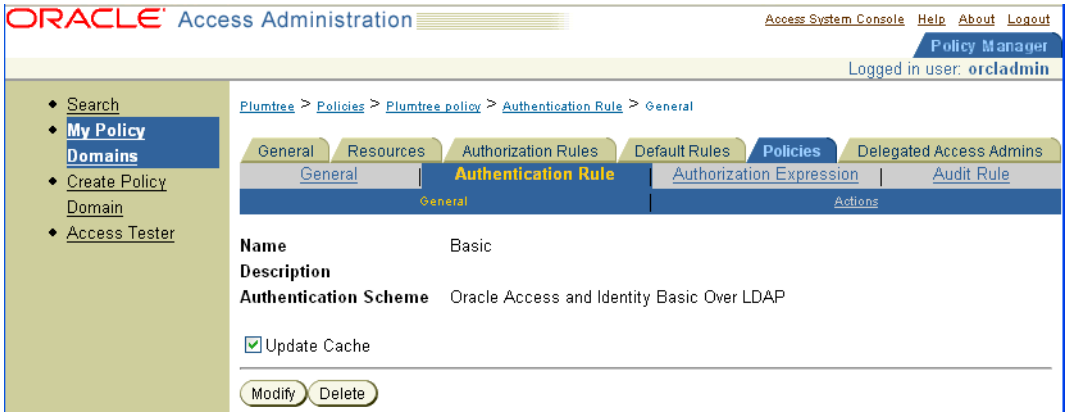
7. In the Policies tab, enter the following information:
   - Name: Enter a name for the policy.
   - Description: Enter a description of the policy.
   - Resource Type: Select HTTP.
   - Resource Operations: Select GET.
   - Host ID: Enter the ID of the portal host.
   - Resource: Select /portal.
   - URL pattern: Enter admin/loginoblix.asp.
   - Host identifiers: Enter the host identity.
   - Query String: Enter a query string.
   - Query String Variable: Enter User ID as the name and 2 as the value.

   Note: The Netscape Web server is case-sensitive. Do not change the case of the query string variable name.

8. Click **Save** to save the policy.
9. In the **Policies** tab, click the name of your policy.

   The policy details page appears.
10. Click **Authentication Rule** and create an authentication rule.

    You can use any authentication scheme such as Basic over LDAP or any custom authentication scheme.



11. In the **Authorization Rules** tab, select **Actions**, and add a header variable that you specified in the file oblix.asp.

    To facilitate single sign-on, you must specify this header variable to direct Plumtree to look for this header variable as described in "Editing Configuration Files to Support Single Sign-On" on page 12-6.
12. In the **Return Attribute** field, enter the **Login ID** attribute and click **Save**.
13. Enable the policy domain.

14. On the Plumtree Portal, lockout the user as described in the section "To lock the NetPointAnonymous account" on page 12-13.

To lock the NetPointAnonymous account

1. On Plumtree, log in as the administrator to the portal.
2. Click **Users** and click the folder that matches the authentication source category you created as your LDAP authentication source.
3. Click the user for the Anonymous authentication scheme.

   It will have the prefix of the name you gave to the LDAP authentication source.
4. In the user page, select **Lock this User** so it cannot be used for login.

   The NetPointAnonymous user becomes the equivalent of the Plumtree guest user.

Note: If you want to log in to Plumtree with the Plumtree database credentials (such as administrator), you must first authenticate yourself to Oracle Access Manager and then click Login as a different User. The Plumtree Login screen appears and you can log in as a different Plumtree user.

To log in to Oracle Access Manager as a different user, first click Logoff to log out of Plumtree and then click Login. The Login box appears, and you can log in as a different user.

Using the Knowledge Directory

The PlumTree 5.0.4 Knowledge Directory is a portal area that users browse to discover document records containing links to documents that have been uploaded by users or crawlers. Users can add Identity System links and Access System-protected resources links to the knowledge directory. Whenever a user tries to access links in the knowledge directory, the ObSSOCookie for that user, if it exists, is passed to the resource for user authentication, thus facilitating single sign-on.

The knowledge directory information is organized into subfolders in a manner similar to file storage volumes and shares. The default portal installation includes a Knowledge Directory root folder with one sub-folder named Unclassified Documents. Before you create additional subfolders, you must define your Knowledge Directory taxonomy, as described in the Deployment Guide for the Plumtree Enterprise Web.

This section discusses the following topics:

- Setting Preferences in the Knowledge Directory
- Creating Folders
- Uploading Documents

Setting Preferences in the Knowledge Directory

You specify how the Knowledge Directory displays documents and folders, including whether to generate the display of contents from a Search Server search or a database query, by setting Knowledge Directory preferences.

To set Knowledge Directory preferences

1. Click **Administration**.
2. In the Select Utility list, click **Knowledge Directory Preferences**.

3. Specify preferences according to the instructions provided through online help.
4. Click **Finish**.

Creating Folders

To create a Knowledge Directory folder, complete the following procedure.

To create a Knowledge Directory folder

1. Navigate to **Directory**, then click **Edit Directory**.
2. Navigate to the folder into which you want to place a new subfolder.
3. Launch the **Folder Editor**.
4. Specify a name and description, then click **OK**.
5. Select the **Edit Details** icon, then complete the settings according to the instructions supplied in the online help.

Uploading Documents

To upload documents to the Knowledge Directory folder, complete the following procedure.

To upload a document

1. Browse to the folder where you want to upload the document.
2. From the **Submit a Document** list, choose **Simple Submit** or choose a data source.
3. Complete the submission forms as described in the online help.

Use filters to control what content goes into which folder. A filter sets conditions to sort documents into associated folders in the Knowledge Directory. Please see PlumTree documentation for more details.

Password Management

The password policies that you set on the Access System are always enforced because Oracle Access Manager manages passwords and the Access System evaluates the policies when a user logs in to the portal.

Users who are not logged in to the Plumtree portal must have access to the Lost Password feature in Oracle Access Manager. This enables users who have lost their passwords to set a new one. See the *Oracle Access Manager Identity and Common Administration Guide* for details.

To enable password management, you must create appropriate policies in the Access System. You also need to use the knowledge directory to upload a document that contains a link to the Lost Password Management feature in Oracle Access Manager. See "Using the Knowledge Directory" on page 12-13 for details.

Note: Oracle recommends that you back up configuration files before you modify them.

Self-Registration

You need to enable, manage, and configure self registration. To enable self-registration, the appropriate self-registration workflow must be created in the User Manager. See

the *Oracle Access Manager Identity and Common Administration Guide* for details. This workflow is used in the self-registration portal insert.

To manage self-registration, you need to use knowledge directory to upload a document that invokes the Identity System self-registration feature. See "Using the Knowledge Directory" on page 12-13 for details. The Plumtree database must then be synchronized to be updated with the newly created user.

13

Integrating mySAP Applications

This chapter describes the integration of Oracle Access Manager with mySAP.com e-business platform.

This chapter covers the following topics:

About Integrating Oracle Access Manager with mySAP

Integrating Oracle Access Manager with mySAP enables the use of Oracle Access Manager functionality across all mySAP Web-based applications and other Oracle Access Manager-protected enterprise resources and applications.

Integrating Oracle Access Manager with mySAP provides the following Oracle Access Manager features to mySAP implementations:

- Access System single sign-on (SSO) for mySAP applications and other Access System-protected resources.
- Authentication, authorization, and auditing services for mySAP applications.
- The following Access System authentication schemes to provide single sign-on for mySAP applications:
  - Form
  - Basic
  - Custom
  - X509 Certificates
  - Integrated Windows Authentication
- Ability to use the Identity System for identity management.

  The Identity System provides identity management features such as portal inserts, delegated administration, workflows, and self-registration to applications such as mySAP.

SAP Components

The integration of Access System single sign-on with mySAP involves the SAP components described in the following sections.

SAP Internet Transaction Server

SAP Internet Transaction Server (ITS) is a mySAP.com component that provides a Web front-end and allows access to data from the SAP R/3 applications. SAP R/3 provides Enterprise Resource Planning (ERP) functionality for the mySAP.com e-business platform.

SAP ITS consists of two major components: AGate and WGate.

The AGate is responsible for session management including mapping of SAP R/3 screens or function modules to HTML. AGate manages Web sessions including timeout handling and SAP R/3 connection pooling. Based on SAP R/3 information, it generates HTML documents that are forwarded to WGate.

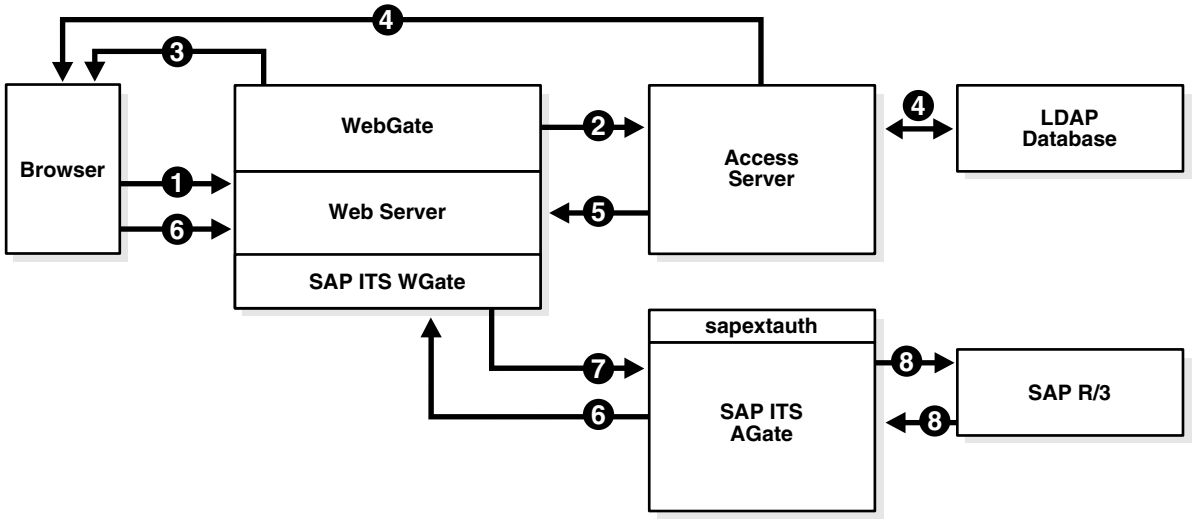
The WGate passes requests to AGate and receives HTML pages back from AGate. The WGate supports various HTTP server interfaces such as Apache, Netscape Server Application Programming Interface (NSAPI), and Internet Server Application Programming Interface (ISAPI).

Pluggable Authentication Service

The Pluggable Authentication Service (PAS) is a part of the Internet Transaction Server that is used for single sign-on between SAP and third-party security providers. PAS enables Oracle Access Manager to authenticate users who attempt to access SAP.com resources.

Integration Architecture

The following figure illustrates the integration between Oracle Access Manager and SAP ITS and the SAP Enterprise Portal. The process overview that follows describes a scenario where the user first authenticates to a resource that is protected by the Access System. The user is then granted access to an SAP resource.



Process overview: Integration with SAP ITS

1. A user attempts to access content or an application on a company's server.
2. The WebGate intercepts the request and queries the Access Server for the security policy that determines if the resource is protected.

   The security policy consists of an authentication scheme, authorization rules, and allowed operations. Based on the authentication and authorization success or failure, specified actions are performed.
3. If the resource is protected, the WebGate prompts the user for authentication credentials.

   The credentials that the WebGate requests depend on the authentication scheme configured in the Access System, for example, Basic over LDAP or Form-based authentication.
4. If the credentials are validated, the Access System authenticates the user and sets an encrypted ObSSOCookie in the user's browser.
5. After authenticating, the authorization rules defined in the Access System are applied based on the security policy.

   Specific actions are performed based on the authorization rules. If the user is authorized, access to the requested content is allowed.

   If the user is not authenticated or authorized, he or she is denied access and redirected to another URL, as determined by the administrator.
6. The user enters the URL for the Oracle Access Manager-specific ITS PAS service.

   For integration with mySAP, an ITS-specific HTTP header variable is created and filled with unique Oracle Access Manager-SAP R/3-mapped user ID information.
7. The Oracle Access Manager-specific ITS service uses the sapextauth module to extract the HTTP header variable and identify the Oracle Access Manager-SAP R/3-mapped user ID.
8. Optionally, the SAP Workplace Server maps the external Oracle Access Manager user ID to the SAP System ID.

   It is recommended that Oracle Access Manager extract the correct SAP user ID from the LDAP directory that is used for the initial authenticated user. In this case, no lockup table is required in the SAP system.
9. If mapping is successful, the AccessGate issue the user an SAP Logon Ticket and redirects the user to either the Workplace service or any other ITS service. Future ITS URLs will use issued SAP Logon Tickets for passing logon information to the SAP R/3 system.

Supported Versions and Platforms

Oracle Access Manager supports the following versions of SAP Server and the SAP Enterprise Portal:

SAP	SAP Portal
SAP R3 v4.6D	v 6.0 SP2
SAP ITS v6.10 and v6.20	

Preparing to Integrate Oracle Access Manager with SAP

Before you can integrate Oracle Access Manager with SAP, you must complete the following tasks.

To prepare for the integration with SAP

1. Install the following SAP applications:
   - SAP ITS v 6.10 with Patch Level 11, Compilation 4 with Patch Level 340 or later (refer to SAPNet Note: 494984).
   - An SAP ITS component configured to talk to the R3 System through Secure NetWork Communications (SNC).

     SNC is required to generate SSO2 security tickets.
   - sapntauth library (refer to SAPNet Note: 493107).

     Refer to SAP documentation for information on installing SAP applications and components.
2. Install the following Oracle Access Manager components:
   - Identity Server
   - WebPass
   - Access Server
   - Policy Manager
   - WebGate

   See the *Oracle Access Manager Installation Guide* for information on installation.
3. For each Web server instance that has ITS installed, install and configure a WebGate.
4. Ensure that mySAP and Oracle Access Manager components are able to communicate with each other through TCP/IP.
5. Ensure that the servers on which SAP ITS and Oracle Access Manager components are installed have a fully qualified domain name.

   For example, integrate-1.oblix.net.
6. Synchronize the time on all servers where SAP and Oracle Access Manager components are installed.
7. Ensure that the users exist in the Oracle Access Manager LDAP directory as well as on the SAP R3 system database.

   The user ID in Oracle Access Manager and SAP must be same or be mapped to each other. Any attribute in a user's profile can be configured as the SAP ID and passed directly to SAP. Alternatively, SAP can be configured to map the SAP ID to any user attribute that it receives from Oracle Access Manager.
8. Configure the Web browser to allow cookies.

Setting up Oracle Access Manager Single Sign-on for mySAP

Setting up Oracle Access Manager single sign-on for mySAP requires the installation and configuration of several SAP and Oracle Access Manager components.

Task overview: Setting Up Oracle Access Manager single sign-on for mySAP

1. Set up SAP, including the following items, as described in "Setting Up SAP for Integration with Oracle Access Manager" on page 13-5:
   - Install the SAP GUI.
   - Install and configure Web server instances for SAP ITS.
   - Install SAP ITS.
   - Test connections between ITS and SAP R/3.
   - Install and configure SAP SNC.
   - Configure SAP PAS for Access System header variables.

   Note: For instructions on installing SAP components, refer to your SAP documentation.

2. Set up the Access System, including the following items, as described in "Setting Up Oracle Access Manager for Integration with SAP" on page 13-7.
   - Install a WebGate
   - Create access control policies in the Access System to protect SAP resources
3. Complete activities in "Testing Integration Between Oracle Access Manager and SAP" on page 13-7.

Setting Up SAP for Integration with Oracle Access Manager

The following procedures describe setting up SAP for the integration.

To set up SAP for integration with Oracle Access Manager

1. Install the SAP Graphic User Interface (GUI) on the client machine.

   This is the Web interface for SAP R/3 applications. It dynamically converts SAP transaction screens to HTML pages.
2. Install and configure two Web server instances; one instance for administrative (ADM) purposes and the second instance as an interface to SAP R/3 applications.

After you have configured the instances, test the connection from ITS to SAP R/3 for both instances.

To test the ADM instance installation

1. Open a Web browser and enter the URL to log in to the ADM instance; for example:

   `http://host:port/scripts/wgate/admin/!`

   or

   `http://host:port/scripts/wgate/adminremote/!`

   where *host* is a fully qualified name of the host machine such as xyz.domain.com, and *port* is the port number of the host machine.

To test SAP R/3 instance installation

1. Open a Web browser and enter the following URL to access the GUI of the SAP R/3 instance:

   `http://host:port/scripts/wgate/webgui/!`

   Where *host* is a fully qualified name of the host machine such as xyz.domain.com and *port* is the port number of the host machine.

   The mySAP.com login screen appears.

2. Install and configure SAP Secure Network Communication (SNC).

   SAP SNC provides secure connectivity from the AGate to the SAP R/3 applications. SNC is recommended because Oracle Access Manager provides an authenticated user ID to SAP.

   If the WGate is installed on a different server than the AGate, it is recommended that you configure SAP SNC between the two servers.

To set up SAP PAS for integration with Oracle Access Manager

1. Configure the SAP PAS system to use Header Variables for SSO:

   Configure WGate to pass the Access System header variables to AGate. To do this, use the parameter PassHeader located in the wgate.conf file.

   For example:

   ```
   Ex.PassHeader HTTP_SAPUID
   ```

   Refer to the SAP documentation for the location of the wgate.conf file.

2. Define the information that PAS requires to use the Access System as an external authentication provider.

   To do this, you must configure the PAS Service for Oracle Access Manager in the Oblix.srvc file located in *SAP_install_dir*\ITS\2.0*ITSInstanceName*\templates

   Where *SAP_install_dir* is the directory where you installed SAP, and *ITSInstanceName* is the name of the ITS instance that you configured.

3. Create and configure PAS templates to handle login, error, and redirect actions that may occur when using the Access System's authentication service.

   Save these templates in the *SAP_install_dir*\ITS\2.0*ITSInstanceName*\templates directory.

   Create the directory structure and files as follows:

   <Name of Service>

   <Name of Theme>

   login.html

   extautherror.html

   redirect.html

   Name of Service is the name of the service file, for example, oblix.srvc.

   Name of Theme is the name of the theme parameter in the oblix.srvc file.

Setting Up Oracle Access Manager for Integration with SAP

The following procedures describe setting up Oracle Access Manager for this integration.

To set up Oracle Access Manager for integration with SAP

1. Install a WebGate on the Web server instance supporting the ITS connection to the SAP R/3 system.

   See the *Oracle Access Manager Installation Guide* for information on installing a WebGate.

2. In the Access System, create a policy domain to protect SAP resources under /scripts/wgate.

   To do this, create a policy domain that protects the Web servers where SAP ITS and WebGate are installed. The Access System sets header variables that are passed on to the Oracle Access Manager-specific ITS service, allowing access only to specified users.

   See the *Oracle Access Manager Access System Administration Guide* for information on creating policy domains.

3. In the Authentication Rule, Actions page of the policy domain, configure the action to set an Access System header variable uid to the SAP uid.

   The following example maps uid to the SAPUID:

   ```
   HeaderVar HTTP_SAPUID uid
   ```

4. In the Authorization Rules, Allow Access page of the policy domain, select the Oracle Access Manager/SAP users to whom you want to grant access to the resources that are protected by the policy domain.

   The single sign-on configuration is now complete.

Testing Integration Between Oracle Access Manager and SAP

After you have integrated Oracle Access Manager with SAP, test for successful Access System authentication and single sign-on with mySAP.

The following procedures test the following scenarios:

- A valid login to an SAP R/3 application with a user ID that is authorized both in the Access System and in SAP.
- A valid login to an SAP R/3 application with a user ID that is authorized in the Access System but is unauthorized in SAP.
- A valid login to the Identity System and an SAP R/3 application with a user ID that is authorized in both the Access System and SAP.
- A valid login to the Identity System and an SAP R/3 application with a user ID that is authorized in the Access System but is unauthorized in SAP.

If Access System authentication was set up correctly, as an authorized user in both Oracle Access Manager and SAP, you will be allowed to access the Identity System as well as any SAP R/3 application. If you are an authorized user only in Oracle Access Manager, you will be allowed to access only the Identity System but not a SAP R/3 application.

If single sign-on has been set up correctly in the Access System, as an authorized user in both Oracle Access Manager and SAP you will need to authenticate to Oracle

Access Manager only once. After successful authentication, you will be able to access the Identity System and multiple SAP R/3 applications without authenticating again.

To test Access System authentication

1. Access any SAP R/3 application.

   If integration was successful, the Access System will challenge you for your credentials.

2. Log in with an authorized Oracle Access Manager/SAP user ID.

   You will be allowed to access the R/3 application.

3. Attempt to log in to a SAP R/3 application with a user ID that is authorized in Oracle Access Manager but is unauthorized in SAP.

   Login will fail with message stating that your password is invalid.

   Note: The message is incorrect. The message should state that your user ID is invalid.

To test Oracle Access Manager single sign-on

1. Access any SAP R/3 application.

   If the integration was successful, the Access System will challenge you for credentials.

2. Log in with an authorized Oracle Access Manager/SAP user ID.

   You will be allowed to access the R/3 application.

3. Attempt to log in to the Identity System.

   If single sign-on is successful, you will be able to log in to the Identity System without being challenged by the Access System.

4. Attempt to log in to the Identity System and a SAP R/3 application with a user ID that is authorized in Oracle Access Manager but is unauthorized in SAP.

   You will be able to log in to the Identity System but not into SAP. SAP will display a message stating that your password is invalid.

Integrating the SAP Enterprise Portal 6.0 and SAP NetWeaver Enterprise Portal SP9

A portal provides a single point of access to enterprise data and applications, presenting a unified and personalized view of information to employees, customers, and business partners.

The SAP Enterprise Portal, which runs on top of SAP R/3, provides unified information from enterprise applications, data warehouses, unstructured document collections, and the Internet.

Integrating Oracle Access Manager with the SAP Enterprise Portal and SAP NetWeaver Enterprise Portal provides the following Oracle Access Manager functionality:

- Ability to use the Identity System to manage users and groups.

Oracle Access Manager and SAP Enterprise Portal share the same LDAP directory. When a new user or group is created in the Identity System, the SAP user repository is updated with the new data.

> **Note:** The SAP Portal supports only static groups.

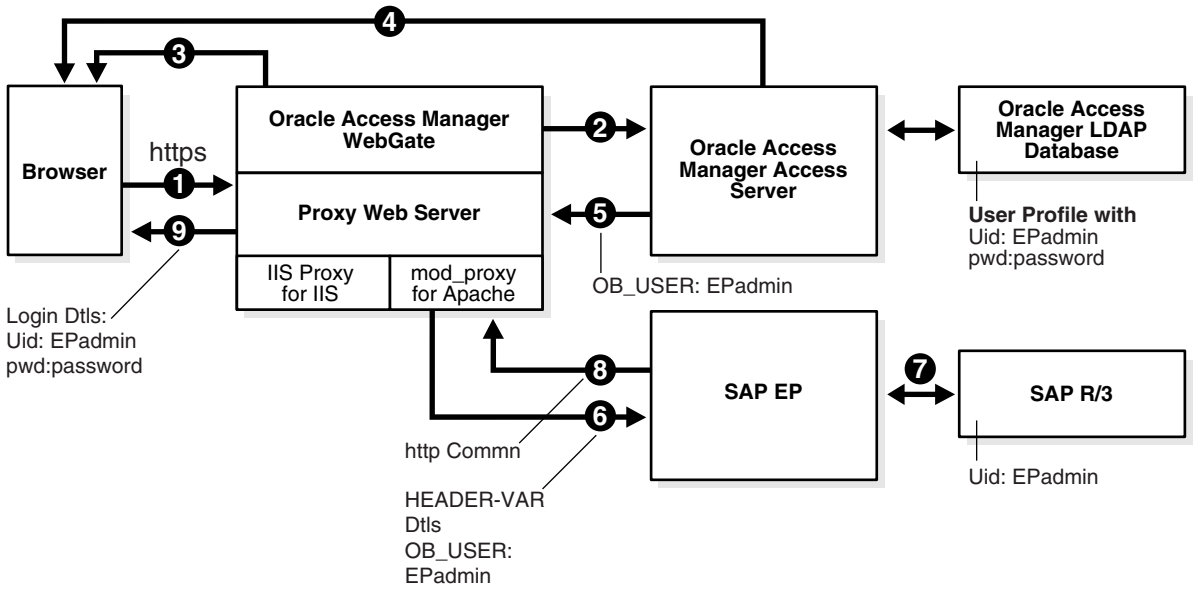
- Access System single sign-on for SAP Enterprise Portal and other Access System-protected resources.

  The Access System authenticates and authorizes users who attempt to access the SAP Portal. After successful authentication and authorization, users can access any Access System-protected resource or application without being prompted again for credentials.

This section discusses the following topics:

Architecture for the Integration with SAP Enterprise Portal 6.0 and SAP NetWeaver Enterprise Portal

The following diagram illustrates this integration:



Process overview: Integration with SAP

1. A user attempts to access content via the SAP Enterprise Portal.

   For example, the user may enter the following URL to access an HR application through a proxy server:

   https://*host*:*port*/irj

2. The WebGate intercepts the request and queries the Access Server for the security policy that determines if the resource is protected.

   The security policy consists of an authentication scheme, authorization rules, and allowed operations. Based on the authentication and authorization success or failure, specified actions are performed.

   The Access System security policy for the SAP /irj login URL is applicable to all resources accessed via the https://*host*:*port*/irj URL.

   Note that the SAP Enterprise Portal has its own authorization system that can be configured to set user access to iViews.

3. If the resource is protected, the WebGate prompts the user for authentication credentials.

   The credentials that the WebGate requests depend on the authentication scheme configured in the Access System, for example, Basic over LDAP or Form-based authentication.

4. If the credentials are validated, the Access System authenticates the user and sets an encrypted ObSSOCookie in the user's browser.

5. After authenticating, the authorization rules defined in the Access System are applied based on the security policy.

   Specific actions are performed based on the authorization rules. If the user is authorized, access to the SAP Portal login (the requested content) is allowed. For SAP Enterprise Portal header variable integration, the Access Server sets the authenticated user ID in a header variable.

   If the user is not authenticated or authorized, he or she is denied access and redirected to another URL, as determined by the administrator. For example, the user may be redirected to an "invalid credentials" page.

6. For the integration with SAP Enterprise Portal, the proxy Web server redirects the request to the SAP Enterprise Portal internal Web server that contains the header variable details.

7. SAP Enterprise Portal uses the header variable value to check the mapping of the user ID against the configured back-end, for example, SAP R/3.

   Both the Oracle Access Manager and SAP Enterprise Portal back-ends must contain the same user ID value.

8. Upon successful mapping, SAP Enterprise Portal allows the user to access the requested resource.

   SAP Enterprise Portal sends a response to the proxy, and the proxy redirects to the client browser.

9. All interaction with the SAP Enterprise Portal takes place through the proxy server.

Supported Platforms for Integrating with SAP Enterprise Portal 6.0 and SAP NetWeaver Enterprise Portal

The following tables list the supported platforms.

Table 13–1 Integration with Oracle Access Manager 6.x

SAP Enterprise Portal	Oracle Access Manager Identity Server	SAP Enterprise Portal Proxy	Oracle Access Manager WebGate
SAP Enterprise Portal 6.0 SP2 patch 4 or higher on Windows	6.5.x	Apache 1.3.x on AIX	6.1.1.x WebGate (SSL or non-SSL)
SAP Enterprise Portal 6.0 SP2 patch 4 or higher on Windows	6.5.x	Apache 2.0.x on Windows	6.5.2 WebGate

Table 13–2 Integration with Oracle Access Manager 7.x

SAP Enterprise Portal	Oracle Access Manager Identity Server	SAP Enterprise Portal Proxy	Oracle Access Manager Access Server
SAP Enterprise Portal 6.0 SP2 patch 4 or higher on Windows	7.0.4	Apache 1.3.x on AIX	7.0.4 WebGate (SSL or non-SSL)
SAP Enterprise Portal 6.0 SP2 patch 4 or higher on Windows	7.0.4	Apache 2.0.x on Windows	7.0.4 WebGate

Table 13–3 Integration with Oracle Access Manager 10.1.4.x

SAP Enterprise Portal	Oracle Access Manager Identity Server	SAP Enterprise Portal Proxy	Oracle Access Manager Access Server
SAP NetWeaver Enterprise Portal SP9 on Windows Server 2003	10.1.4	Apache 2.0.x on Linux	10.1.4 WebGate (SSL or non-SSL)

Integrating SAP Enterprise Portal 6.0

The following sections describe how to integrate the SAP Enterprise Portal 6.0:

- Prerequisites
- Configuring a Proxy to Access SAP Enterprise Portal 6.0
- Configure Oracle Access Manager for SAP Enterprise Portal 6.0
- Configure WebGate on the Proxy Server
- Configure SAP Enterprise Portal 6.0 for External Authentication
- Testing the Integration with SAP Enterprise Portal 6.0
- Troubleshooting the Integration with SAP Enterprise Portal 6.0

Prerequisites

The following are tasks that you must complete before integrating the SAP Enterprise Portal 6.0 with Oracle Access Manager.

Task overview: Integration prerequisites for SAP Portal 6.0:

1. Confirm the installation of SAP Enterprise Portal 6.0 SP2 and its components and applications:
   - Ensure SAP J2EE Engine version 6.2 patch level 26 or higher is installed, according to the instructions in the SAP note 616501.
   - Apply SAP Enterprise Portal 6.0 patch level 4 or higher. This is required for the SAP logout URL redirection functionality.
   - Confirm that SAP Enterprise Portal is functional and able to access the applications.
2. Ensure that the Identity System and Access System are installed and running.
3. Ensure that the Oracle Access Manager and the SAP Enterprise Portal back-ends contain the same user ID information.

   This is essential for this integration.

Task overview: Integration prerequisites for Oracle Access Manager integration:

1. Configure a proxy server to access SAP Enterprise Portal 6.0.
2. Configure Oracle Access Manager for the SAP Enterprise Portal 6.0.
3. Configure a WebGate on the proxy server.
4. Configure SAP Enterprise Portal 6.0 for external authentication.

Configuring a Proxy to Access SAP Enterprise Portal 6.0

The following procedure describes how to configure a proxy to access SAP Enterprise Portal 6.0.

To configure Apache Web server 1.3.x or 2.0.1

1. Set up the Apache proxy in non-SSL mode or SSL mode, as described in the Apache documentation.

   If https communication is used with the SAP Enterprise Portal 6.0, use SSL mode.
2. To enable the proxy to the SAP Enterprise Portal 6.0, enter the following in httpd.conf:

   ```
   ProxyRequests Off
   ProxyPass /irj http://sap_host:port/irj
   ProxyPassReverse /irj http://sap_host:port/irj
   ProxyPreserveHost On
   ```

   Where *sap_host* is the name of the machine hosting the SAP Enterprise Portal 6.0 instance and *port* is the listen port for the SAP Enterprise Portal 6.0 instance. This set of directives specifies that all of the requests to this Web server of the form http://*apache_host*:*port*/irj or https://*apache_host*:*port*/irj are redirected to http://*sap_host*:*port*/irj or https://*sap_host*:*port*/irj.
3. Restart the proxy Web server.
4. Access the following URL:

   Non-SSL—`http://apachehost:port/irj`

   SSL—`https://apachehost:port/irj`

   This request should be redirected to the SAP Enterprise Portal 6.0 login.

5. Log in using the SAP Enterprise Portal 6.0 administrator login ID.

   The administrator should be able to perform the available administrative functions.

6. Log in as a non-administrative user.

   This user should be able to perform non-administrative functions.

Configure Oracle Access Manager for SAP Enterprise Portal 6.0

The following procedure describes configuration of the security policy in Oracle Access Manager to protect logins to SAP Enterprise Portal 6.0. For more information on configuring policy domains, see *Oracle Access Manager Access System Administration Guide*.

To configure Oracle Access Manager for SAP Enterprise Portal 6.0

1. Log in to the Access System Console as a Master Access Administrator.
2. Click the Access System Configuration tab.
3. Click Add New AccessGate in the left navigation pane.
4. Configure a WebGate that you will install on the proxy server, as follows:

   AccessGate name—Enter any meaningful name, for example, SAP_AG. Use an alphanumeric string, and do not include spaces in the name.

   Host name—Enter the name of the Apache proxy machine.

   Access Management Service—Click the option to enable this service.

5. Click Save, then click List Access Servers at the bottom of the page to associate this WebGate with a defined Access Server.
6. Click Host Identifiers in the left navigation pane and configure the host identifiers using the fully qualified proxy machine name and port for the Apache proxy.
7. Click the link for the Policy Manager at the top of the page.
8. Click Create Policy Domain in the left navigation pane and create a new policy named SAP EP Security Policy.
9. Click the Resources tab, click Add, and define the resources for the policy as follows:

   Name: SAP EP Security Policy

   Type: http

   Host identifiers: Enter the proxy host URL prefix: /irj.

   Description: SAP EP Login URL

10. Click the authorization rules tab, then click Add, and define the authorization rules for the policy as follows:

    Name: SAP Authorization Rule

    Enabled: Yes

    Allow takes precedence: Yes

11. Click Save, then click the Allow Access tab, then click Add and add a valid user or group list.
12. Click Save, then click the Actions tab, and configure the following actions:

Authorization Success Returns: Configure a Type of HeaderVar, a Name of OB_USER, and a Return Attribute of uid.

Authorization Failure: Configure an appropriate failure action. For example, you can configure a redirect to a proxy URL page that displays an "Invalid login credentials" message.

13. Click the Default Rules tab, click the Authentication Rule sub-tab, click Add, and define the default authentication rule as follows:

    Name: SAP Authentication Rule

    Authentication Scheme: Select either Basic over LDAP or Form based authentication. Oracle recommends that you use a form-based authentication scheme. If you use the basic authentication scheme, also set the Challenge Redirect field to another WebGate to ensure that the ObSSOCookie is set.

14. In the Default Rules tab, click the Authorization expression sub-tab, click Add, and create an authorization expression that uses the SAP Authorization Rule.

15. If you configured a form-based authentication scheme, ensure that a login.html page is configured in the proxy server document root.

    This form will be used to get the user credentials. See the *Oracle Access Manager Access System Administration Guide* for details.

16. Also, if you configured a form-based authentication scheme, ensure that a logout.html page is present on the proxy Web server document root.

    You can create a custom logout page using HTML, JSP, or a CGI protocol.

    The default logout page, named logout.html, is in the following location:

    ```
    WebGate_install_dir\access\oblix\apps\common\bin
    ```

    Where *WebGate_install_dir* is the directory where the WebGate will be installed. Ensure that the name of the logout page contains the string "`logout.`".

17. Ensure that the user ID that is returned by the OB_USER header variable exists in the user management data sources for SAP Enterprise 6.0.

Configure WebGate on the Proxy Server

Install a WebGate on the Web server instance that supports the proxy connection to the SAP Enterprise Portal 6.0 instance. See the *Oracle Access Manager Installation Guide* for details.

Configure SAP Enterprise Portal 6.0 for External Authentication

The following steps describe enabling external authentication in SAP Enterprise Portal 6.0 using the OB_USER header variable.

For more information on configuring authentication schemes for SAP Enterprise Portal, see the *SAP Enterprise Portal 6.0 SP2 Enterprise Postal Security Guide*.

To configure SAP Enterprise Portal 6.0 for external authentication

1. To enable logout from a single sign-on session in both SAP Enterprise Portal 6.0 and Oracle Access Manager, configure a logout URL in SAP Enterprise Portal 6.0 from the administration interface.

   The URL for the administration interface is as follows:

   http://*SAP_host*:*port*/irj/

Where *SAP_host* is the name of the machine hosting the SAP Enterprise Portal 6.0 and *port* is the listen port for the portal.

2. From the administration interface, click System Administration, then System Configuration, then UM Configuration, then Direct Editing.
3. Add the following lines to the end of the configuration file:

```
ume.logoff.redirect.url=http(s)://proxy_host:port/logout.html
ume.logoff.redirect.silent=false
```

Where *http(s)* is either http or https, *proxy_host* is the name of the proxy Web server and *port* is the listen port for the proxy.

4. Save the changes and log out.
5. Stop the SAP J2EE dispatcher and server.
6. Browse to the following directory:

SAP_J2EE_engine_install_dir\ume

7. Back up the file authschemes.xml.bak to another directory.
8. Rename authschemes.xml.bak to authschemes.xml.
9. Open authschemes.xml in an editor and change the reference of the default authentication scheme to the authentication scheme header as follows:

```
<authscheme-refs>
     <authscheme-ref name="default">
          <authscheme>header</authscheme>
     </authscheme-ref>
</authscheme-refs>
```

10. In the authentication scheme header of authschemes.xml, specify the name of the HTTP header variable where the Access System provides the user ID.

As described in "Configure Oracle Access Manager for SAP Enterprise Portal 6.0" on page 13-13, this is the OB_USER header variable. You configure this header variable as follows:

```
<authscheme name="header">
     <loginmodule>
          <loginModuleName>
               com.sap.security.core.logon.imp.HeaderVariableLoginModule
          </loginModuleName>
               <controlFlag>REQUISITE</controlFlag>
               <options>Header=OB_USER</options>
     </loginmodule>
     <priority>5</priority>
     <frontEndType>2</frontEndType>
     <frontEndTarget>com.sap.portal.runtime.logon.header</frontEndTarget>
</authscheme>
```

The control flag value REQUISITE means the login module must succeed. If login succeeds, authentication continues through the list of login modules. If it fails, control immediately returns to the application and authentication does not continue through the list of login modules.

11. Restart the portal server and J2EE engine.

The modified authschemes.xml file will be loaded into the Portal Content Directory (PCD). SAP Enterprise Portal 6.0 will rename it as authschemes.xml.bak.

Testing the Integration with SAP Enterprise Portal 6.0

The following are recommended tests for the integration.

To test the integration

1. Enter the appropriate URL for your proxy in a browser:

   `http(s)://proxy_host:port/irj`

   Oracle Access Manager should prompt for user credentials depending on the authentication scheme (form-based or Basic over LDAP).

2. Enter valid user credentials at the prompt.

   These credentials should belong to a user in Oracle Access Manager who is authorized to access /irj. This user ID must also be present in the SAP Enterprise Portal 6.0 User Management System.

   The user should be logged into the SAP Enterprise Portal 6.0 having supplied the login credentials to Oracle Access Manager.

3. If the user have administrative privileges in SAP Enterprise Portal 6.0, ensure that the usual administrative functions are available for this user.

   For example, search for other SAP Enterprise Portal users through the user Management System.

4. Log off from the SAP Enterprise Portal.

   You should be redirected to the `logout.html` page that you configured. Both the SAP Enterprise Portal and the Oracle Access Manager sessions should end.

5. Access SAP Enterprise Portal login again in the same browser.

   Oracle Access Manager should prompt for login credentials again.

6. Access the SAP Enterprise Portal login page without using the proxy by entering the following in a browser:

   `https://sap_host:port/irj`

   You should receive the following error:

   ```
   Cannot logon user defined in header variable!
   ```

Troubleshooting the Integration with SAP Enterprise Portal 6.0

The following information is intended to help you troubleshoot issues with this integration.

Problem: The browser has problems displaying the SAP administration interface through the proxy server. You may receive an "object not found" error and related javascript errors.

Solution: See the following SAP document for supported browsers, "*SAP NetWeaver '04 SR1 PAM: Browsers for end users and admin functionality*." SAP has recommended Internet Explorer 6, and it supports almost all proxy operations. In the case of Internet Explorer, the Microsoft Security patch can sometimes affect the display of the user interface. See SAP Note 785308 for details.

Integrating SAP NetWeaver Enterprise Portal SP9

The following sections describe the integration between Oracle Access Manager and SAP NetWeaver Enterprise Portal SP9:

Prerequisites for Integrating with SAP NetWeaver Enterprise Portal

Complete the following tasks before configuring the integration with SAP NetWeaver Enterprise Portal:

- Verify the installation of SAP Netweaver Enterprise Portal SP9 and its associated components and applications.

  Ensure that the portal is functional and can access the applications.

- Install Oracle Access Manager is installed and running.

  See the *Oracle Access Manager Installation Guide* for details.

- Ensure that both Oracle Access Manager and the SAP Enterprise Portal back-ends contain the same user ID information.

  This is essential for integrating the SAP HEADER-VARs.

Configuring the Proxy Servers and Oracle Access Manager for integration with SAP NetWeaver Enterprise Portal

The following procedures describe how to set up the integration with SAP NetWeaver Enterprise Portal using an OB_USER header variable. These procedures describe required configurations for Oracle Access Manager and SAP NetWeaver Enterprise Portal.

Note: For additional information on configuring policies, authentication schemes, and form-based authentication, see the *Oracle Access Manager Access System Administration Guide*.

To configure the proxy server to access NetWeaver

1. Set up an Apache proxy Web server in SSL or non-SSL mode, as required in your environment.

   If you want HTTPS to be used for client-to-SAP Enterprise Portal communication, set up the proxy in SSL mode. See the Apache documentation for details.

2. To enable the proxy for the SAP Enterprise Portal, add the following information to httpd.conf:

   ```
   ProxyRequests off
   ProxyPass /irj http://SAPHost:port/irj
   ProxyPass /webdynpro http://SAPHost:port/webdynpro
   ProxyPassReverse /irj http://SAPHost:port/irj
   ProxyPassReverse /webdynpro http://SAPHost:port/webdynpro
   ProxyPreserveHost On
   ```

   Where *SAPHost* and *port* are the host and port for the SAP Enterprise Portal server. These directives specify that all requests to this WebServer of the form http://*apachehost*:*port*/irj or https://*apachehost*:*port*/irj are redirected to http://*saphost*:*port*/irj or https://*saphost*:*port*/irj.

3. Restart the proxy Web server.
4. Access the following URL:
   - Non-SSL: http://*apachehost*:*port*/irj
   - SSL: https://*apachehost*:*port*/irj

   This request should be redirected to the SAP Enterprise Portal login page.
5. Log in using the SAP Enterprise Portal administrator ID.

   The administrative user should be logged into the SAP Enterprise Portal. This use must be able to perform all administrative operations.
6. Confirm SAP Enterprise Portal login for a non-administrative user.

To configure an SAP Enterprise Portal security policy in Oracle Access Manager

1. Log in to Oracle Access Manager's Access System as a Master Administrator.
2. From the Access System Console, click Access System Configuration, then click Add New AccessGate.
3. Configure a WebGate that will be installed on the proxy server, as follows:
   - Provide a name, for example, SAP_AccessGate.
   - This host name for this WebGate is the machine name of the proxy server.
   - Enable the Access Management Service.
   - Associate this WebGate with an existing Access Server.
4. In the left navigation pane, click Host Identifiers and configure the host identifiers for the fully qualified proxy machine name and port.
5. Click the link for the Policy Manager and create a new security policy named, for example, SAP EP Security Policy.
6. Configure the resources for this policy as follows:
   - **Type**: http
   - **Host identifiers**: The host identifiers that you created for the proxy host
   - **URL prefix**: /irj
   - **Description**: SAP EP Login URL
7. Configure the authorization rules as follows:
   - **Name**: SAP Authz Rule
   - **Enabled**: Yes
   - **Allow takes precedence**: Yes
   - **Allow access**: Add a valid user and group list for Allow Access
   - **On success returns**:
     Type: HeaderVar
     Name: OB_USER
     Return Attribute: uid
     The user ID that Oracle Access Manager returns in the OB_USER header variable must exist in the user management data sources of the SAP portal.

- **On failure**: For authorization failures, point to a URL that displays an appropriate error message, for example, "Logon credentials cannot be recognized."

8. Create a default rule for this policy:
   - **Name**: SAP Authen Rule
   - **Scheme**: The scheme can be Basic over LDAP or form-based authentication.

     Oracle recommends that you use a form-based authentication scheme. If you use basic authentication, be sure to set the Challenge Redirect field to point to another WebGate. This ensures that the ObSSOCookie is set.

To configure a form-based authentication scheme for NetWeaver

1. Be sure that the login.html page resides in the proxy server document root.

   This form is used to collect user credentials.

2. Be sure that the logout.html page resides on the proxy Web server document root.

   You can create a custom logout page using HTML, JSP or the CGI protocol. The default Oracle Access Manager logout page is located in the following directory:

   WebGate_install_dir\access\oblix\apps\common\bin

   Where *WebGate_install_dir* is the directory where the WebGate is installed. Ensure that the name of the logout page contains the string "logout.".

To configure an Oracle Access Manager WebGate on the proxy server

Install the Oracle Access Manager WebGate on the Web server instance that supports the proxy connection to the SAP Enterprise Portal instance.

See the *Oracle Access Manager Installation Guide* for details.

Configuring SAP NetWeaver Enterprise Portal for the Integration

The following procedures describe how to configure SAP NetWeaver for the integration using the OB_USER header variable. You must also configure single sign-on logout to ensure that the user session is ended in Oracle Access Manager.

The following procedures describe modifying UME properties to configure logout, changing the associated Visual Administrator properties, and adjusting the login module stack to use header variables.

To configure the UME properties

1. Start the following configuration tool:

   SAPJ2EEEngine_install_dir\j2ee\configtool\configtool.bat

2. In the tree view, navigate to Global Server Configuration, then to services, then to com.sap.security.core.ume.service.

   A list of UME properties appears.

3. Select the property for the logoff page that you want to modify.

   The property is ume.logoff.redirect.url.

4. In the Value field at the bottom of the page, enter a new value, as follows:

   http(s)://*proxy_host*:*port*/logout.html

5. Select the property for silent logoff.

The property is ume.logoff.redirect.silent.

6. In the Value field at the bottom of the page, enter a a value of FALSE.
7. Click Set.
8. Choose Save with the quick information text.
9. Choose Apply changes.
10. Restart the AS Java.

To configure the Visual Administrator properties

1. Run the Visual Administrator tool, in the following location:

   SAPJ2EEEngine_install_dir\j2ee\admin\go.bat
2. Click the service for the J2EE server, then click Ume Provider.
3. Set the same properties that you set when running the configuration tool in the previous procedure.
4. Restart the AS Java.

To modify the login module stack to use header variables

1. Run the Visual Administrator tool, in the following location:

   SAPJ2EEEngine_install_dir\j2ee\admin\go.bat
2. In the Visual Administrator, choose Security Provider, then choose the User Management tab, then choose Manager Security Stores.

   The currently active user store and the login modules for the user store appear.
3. Choose Add Login Module.

   A dialog box appears prompting you to choose an editor for the login module.
4. Click OK.

   A dialog box prompting you to add a login module appears.
5. Add the following information:
   - ClassName: com.sap.security.core.server.jaas.HeaderVariableLoginModule
   - Display Name: HeaderVariableLoginModule
6. Click OK.

   The HeaderVariableLoginModule now appears in the list of login modules for the active user store.
7. To add the HeaderVariableLoginModule to the appropriate login module stack or template, in the Visual Administrator, choose Security Provider, then choose Policy Configurations Authentication.
8. To configure the options, add the header variable login module to the ticket login module.

   The ticket login module stack should be as follows:

Table 13–4 Ticket Login Module Stack

Module	Key	Value
com.sap.security.core.server.jaas.EvaluateTicketLoginModule	SUFFICIENT	{ume.configuration.active=true}

Table 13–4 (Cont.) Ticket Login Module Stack

Module	Key	Value
com.sap.security.core.server.jaas.HeaderVariableLoginModule	REQUISITE	{ume.configuration.active=true, Header=OB_USER}
com.sap.security.core.server.jaas.CreateTicketLoginModule	SUFFICIENT	{ume.configuration.active=true}

Note that the Control Flag value REQUISITE means that the LoginModule is required for succeed. If it succeeds, authentication continues down the list of configured modules. If it fails, control immediately returns to the application, and authentication does not proceed down the list of login modules.

Testing the Integration with SAP NetWeaver Enterprise Portal

The following procedure describes how to test the integration.

To test the integration with SAP NetWeaver Portal

1. Access the proxy URL:

   http(s)://*proxy_host*:*port*/logout.html

   Oracle Access Manager should prompt for login credentials, based on the configured Basic or form-based authentication scheme.

2. Enter valid user credentials for an Oracle Access Manager user who is also authorized to access /irj.

   This user ID must also exist in the SAP NetWeaver Enterprise Portal user management system.

3. The user should be logged into the SAP NetWeaver Enterprise Portal without being prompted for credentials a second dimte.

4. Log in as an administrator, access the SAP User Management System, and search for other SAP NetWeaver Enterprise Portal Users.

5. Click Logoff in SAP NetWeaver Enterprise Portal.

   You should be redirected to the logout.html page. The logoff should end both the SAP NetWeaver Enterprise Portal and Oracle Access Manager sessions. If you access the SAP Enterprise Portal login again in the same browser, Oracle Access Manager should prompt for the credentials again as long as you have protected /irj with a form login cookie.

6. Access the SAP NetWeaver Enterprise Portal login page without using a proxy, for example, go to the following URL:

   https://*SAP_host*:*port*/irj

   You should not be able to log in directly.

Troubleshooting the Integration with SAP NetWeaver Enterprise Portal

The following information is intended to help you troubleshoot issues with this integration.

Problem: The browser has problems displaying the SAP administration interface through the proxy server. You may receive an "object not found" error and related javascript errors.

Solution: See the following SAP document for supported browsers, "*SAP NetWeaver '04 SR1 PAM: Browsers for end users and admin functionality*." SAP has recommended Internet Explorer 6, and it supports almost all proxy operations. In the case of Internet Explorer, the Microsoft Security patch can sometimes affect the display of the user interface. See SAP Note 785308 for details.

14

Integrating the RSA SecurID Authentication Plug-In

Oracle provides components that interface with RSA Security products to provide native RSA SecurID® authentication for Oracle Access Manager-protected resources. This chapter introduces SecurID authentication and the components, requirements, and processes needed to successfully integrate SecurID authentication with Oracle Access Manager 10*g* (10.1.4.0.1).

About Oracle Access Manager and SecurID Authentication

Oracle Access Manager integrates with RSA components to provide SecurID authentication.

RSA SecurID authentication is based on two factors: something the user knows and something the user has.

- **Something the User Knows**: This is a secret personal identification number (PIN), similar in concept to a personal bank code PIN. In this case, the PIN may be system generated or personally chosen and registered with the RSA ACE/Server®, which has been renamed to the Authentication Manager.
- **Something the User Has**: This is the current code generated by a hand held device known as a token. Oracle Access Manager supports all RSA SecurID tokens including RSA SecurID Standard Card, Key Fob, PINPAD Card, and a software-based security token (SoftID) that resides on a user's computer.

The random unpredictable code generated by the token is known as a tokencode. Together, the user's PIN and the SecurID tokencode become the user's Passcode.

The following components are needed for the integration:

- Oracle Access Manager Components

See also, "Integration Summary" on page 14-4.

Supported Versions and Platforms

You can find support and certification information at the following URL:

`http://www.oracle.com/technology/documentation/`

You must register with OTN to view this information.

Also, you can see the supported versions and platforms for this integration on Metalink, as follows.

To view information on Metalink

1. In your browser, enter the following URL:

   `https://metalink.oracle.com`

2. Log in to MetaLink.
3. Click the **Certify** tab.
4. Click **View Certifications by Product**.
5. Select the **Application Server** option and click **Submit**.
6. Choose **Oracle Identity Management** and click **Submit**.
7. Click **Oracle Identity Management Certification Information 10g (10.1.4.0.1) (html)** to display the Oracle Identity Management page.
8. Click the link for **Section 6, Oracle Access Manager Certification** to display the certification matrix.

RSA Components

During the SecurID authentication process, users must submit their username and passcode using an HTML form. The RSA ACE/Server authenticates the identity of each user through a computer that is registered with the ACE/Server as a client (ACE/Agent). In this case, one Oracle Access Manager SecurID Access Server must be registered and set up as an ACE/Agent. See "Oracle SecurID Access Server and ACE/Agent Requirements" on page 14-7 for more information.

Note: While the RSA ACE/Server has been renamed to the RSA Authentication Manager, this chapter uses the original naming convention.

The RSA ACE/Server compares the tokencode it has generated with the tokencode the user has entered. Tokencodes change at a specified interval, typically 60 seconds. Time synchronization ensures that the tokencode displayed on a user's token is the same code the ACE/Server software has generated for that moment. Authentication is successful when the tokencodes match.

Two-factor authentication provides stronger legal evidence of who performed the task. When properly implemented, the ACE/Server tracks all login requests and operations to reliably identify the user who is responsible for each logged action. For example, see the administration documentation for RSA 5.1 for details about RSA audit trail reports, the automated log database feature, and monitoring activity in real time.

Oracle Access Manager enables integration of SecurID authentication by providing the following:

- The HTML forms required for SecurID authentication operations
- The CGI script required to authenticate users with the RSA ACE/Server
- The SecurID authentication plug-in, authn_securid, required for the Oracle Access Manager SecurID authentication scheme

Oracle Access Manager uses and supports the following RSA security features:

- Two-factor SecurID authentication
- RSA BSAFE® SSL-C and BSAFE® Crypto-C
- RC6 encryption for cookies passed between a WebGate and the user's browser
- Optional RSA Keon® Certificate Server and X.509v3 digital certificates (RSA's Keon public key infrastructure (PKI))

Note: Oracle Access Manager supports, but does not require, certificate publishing. SecurID may be used as an authentication scheme for both Oracle Access Manager and Keon Web Passport. However, separate authentications must be completed for each product. See the appropriate implementation guide for RSA Keon Certificate Authority for your LDAP directory server.

Oracle Access Manager Components

The Identity System provides the applications you need to manage users, groups, organizations, identity-based workflows, and delegated administration.

- All ACE/Server users must be added to the Identity System.
- Delegated Identity Administrators can create a workflow definition to add ACE/Server users to the Oracle Access Manager directory. See the *Oracle Access Manager Identity and Common Administration Guide* for more information.

The Access System provides policy-based authentication, authorization, auditing, and Web single sign-on. Access System components for SecurID authentication include the Policy Manager, Access System Console, Access Server, and WebGate(s), described in the following list.

- **The Policy Manager**: The Policy Manager provides the applications for policy management, designation of resources (Web and non-Web) and policy testing. Master Access Administrators define policy domains and Delegated Access Administrators define the resources to be protected by a policy domain.
- **The Access System Console**: Master Oracle Access Manager Administrators and Master Access Administrators use the Access System Console to define authentication schemes, including the SecurID authentication scheme required for this integration, and authorization rules that allow or deny access to resources. The Access system, host identifiers, and master audit settings are also configured in the Access System Console.
- **The Access Server**: The Access Server receives requests from a WebGate and queries authentication, authorization, and auditing rules stored in the directory server used by Oracle Access Manager. The Access Server returns the authentication scheme, user credentials, and authorization to the requesting WebGate.

The Access Server installation includes the SecurID authentication plug-in, which is a shared library that makes outbound calls to verify the user's authentication credentials against those on the RSA ACE/Server. To accomplish this, one Access Server must be set up as an ACE/Agent. See "Oracle SecurID Access Server and ACE/Agent Requirements" on page 14-7.

- **WebGate**: WebGate intercepts and forwards HTTP requests for Web resources to the Access Server for authentication and authorization. The WebGate also starts the user's session, creates session cookies, and passes these to the user's browser.

The WebGate installation includes the SecurID forms and the CGI script needed to authenticate users with the ACE/Server and to support two special modes of operation. See "Next Tokencode Mode Support" on page 14-6 and "New PIN Mode Support" on page 14-6 for more information.

Integration Summary

Oracle Access Manager 10*g* (10.1.4.0.1) integrates with SecurID ACE Server 5.0 and supports SDI encryption mode. This version is used for demonstration purposes only.

Table 14–1 summarizes Oracle Access Manager SecurID integration features.

Table 14–1 SecurID Integration Summary

Feature	Support for the feature
Authentication method	Native SecurID authentication
New PIN Support	All
Next tokencode support	Yes
Secondary server support	Yes
Location of node secret on Windows client	%windir%\system32
Location of node secret on Unix client	ACE/Agent installation directory
ACE/Agent installation directory	Net OS Agent and Unix Agent
SecurID user specification	Designated users
SecurID protection of administrators	Yes
Identity System features and functions	All
Access System features and functions	All

Support and Requirements

Requirements for SecurID authentication are discussed in the following topics.

- RSA ACE/Server Requirements
- Oracle SecurID Access Server and ACE/Agent Requirements
- Access Server and ACE/Agent Requirements
- WebGate Requirements

Note: Oracle Access Manager does not support multiple ACE realms. The ACE Agent uses an automatic response time load balancing algorithm to determine where to send an authentication request. Such requests go to either a primary Ace Server or a replica. The automatic algorithm can be overridden by creating a manual load balancing configuration file, sdopts.rec. However manually weighting an ACE server as a server of last resort does not preclude the Agent from communicating with it. As such, a true failover setup cannot be achieved with this method. For more information, see your ACE server documentation.

Supported Versions and Platforms

You can find support and certification information at the following URL:

http://www.oracle.com/technology/documentation/

You must register with OTN to view this information.

Also, you can see the supported versions and platforms for this integration on Metalink, as follows.

To view information on Metalink

1. In your browser, enter the following URL:

   https://metalink.oracle.com

2. Log in to MetaLink.
3. Click the **Certify** tab.
4. Click **View Certifications by Product**.
5. Select the **Application Server** option and click **Submit**.
6. Choose **Oracle Identity Management** and click **Submit**.
7. Click **Oracle Identity Management Certification Information 10g (10.1.4.0.1) (html)** to display the Oracle Identity Management page.
8. Click the link for **Section 6, Oracle Access Manager Certification** to display the certification matrix.

RSA ACE/Server Requirements

The RSA ACE/Server software provides SecurID identification and authentication of users in the ACE/Server data directory. Data from ACE/Server user records in the directory are validated with the ACE/Server's token records. The ACE/Server's native LDAP support is separate from, yet compatible with, Oracle Access Manager.

RSA ACE/Server Installation and Configuration Requirements

The following installation and configuration must be completed before you begin SecurID integration with Oracle Access Manager.

RSA ACE/Server installation and configuration guidelines

- The SecurID ACE/Server software must be installed on a supported platform.
- The system time must be correct to prevent the server and client from being out of sync.

- The SecurID tokens or key fobs must be installed, SecurID users must be created on the ACE/Server, and tokens must be assigned.
- Each user name must be mappable through an LDAP filter to a Distinguished Name in the directory.
- A user who authenticates through a RADIUS server must have a profile in the ACE/Server database that provides a list of requirements the user must meet before the ACE/Server challenges the RADIUS user for a passcode.
- An ACE/Server slave and/or replicated ACE/Server can provide failover if the primary ACE/Server is down.

Setting up your RSA ACE/Server, SecurID tokens, and users is outside the scope of this manual. See the installation and administration documentation for the supported RSA ACE/Server for details and troubleshooting tips.

As discussed earlier, Oracle Access Manager supports all RSA SecurID tokens and SecurID authentication. The following modes are also supported.

- Next Tokencode Mode Support
- New PIN Mode Support

Next Tokencode Mode Support

During authentication, the ACE/Server may direct the user to provide the next tokencode that appears on their SecurID token to prove that they have the assigned token. This operation is known as Next Tokencode mode, which may be triggered by one of the following situations:

- An incorrect Passcode was provided repeatedly during login.

  When a user attempts authentication with incorrect Passcodes four consecutive times, the ACE/Server turns on Next Tokencode mode, as noted in the ACE/Server's Activity Report. The next time the user successfully authenticates with their correct Passcode, they are challenged for the next tokencode that appears on their SecurID token.

- The ACE/Server requires confirmation of, or synchronization with, the token.

  Even with a correct Passcode, the ACE/Server administrator may set the Next Tokencode mode On to force the user to confirm that they have the SecurID token or to synchronize the token with the ACE/Server.

  When Next Tokencode mode is On, the Next Tokencode challenge form is presented to the user immediately following a successful login. See "Next Tokencode Sequence" on page 14-11 for details.

New PIN Mode Support

The token may be in New PIN mode the first time the user logs in or the ACE/Server administrator may enable New PIN mode. New PIN mode requires the user to complete a sequence of forms to define, or have the system generate, a new PIN number. Table 14–2 provides a description of the New PIN forms and their functions.

Table 14–2 Oracle-Provided New PIN Forms and Functions

New PIN Function	Description
New PIN Query form	Tells the user a new PIN is required. Provides instructions and fields to fill in.

Table 14–2 (Cont.) Oracle-Provided New PIN Forms and Functions

New PIN Function	Description
New PIN Confirmation form (system-generated PINs)	Confirms the system-generated PIN for login. Redirects to the resource within 30 seconds
New PIN form (user-generated PINs)	Asks for a username, Passcode, and new PIN.

Each PIN may be:

- Four to eight alphanumeric or numeric characters
- All the same length or of varying lengths
- Defined by the user or by generated the system

See "New PIN Sequence" on page 14-11 for details.

Oracle SecurID Access Server and ACE/Agent Requirements

All SecurID authentication requests must be directed to a single Access Server (also known as the Oracle SecurID Access Server). The RSA ACE/Agent v5.x, formerly known as the ACE/Client, is an RSA Security program that must be installed on all installed Access Servers. However, only one Access Server may be registered as an RSA ACE/Agent to perform the authentication dialog with the RSA ACE/Server.

Access Server-compatible Web servers installed on the Operating Systems shown in "Access Server and ACE/Agent Requirements" on page 14-7 will support RSA ACE/Agent software v5.x.

Access Server and ACE/Agent Requirements

The RSA ACE/Agent software is included with the Access Server on Unix systems and must be installed manually on Windows-based Access Servers.

Oracle SecurID Access Server guidelines

- All Access Servers in the installation must have the RSA ACE/Agent software installed.
- One Access Server must be registered as an ACE/Agent Host on the ACE/Server and must have the RSA ACE/Agent software installed to:
- Enable the Access Server to be recognized as an ACE/Server client
- Manage authentication requests from the client to the ACE/Server
- Enforce two-factor authentication and block unauthorized access
- Provide automatic load balancing by detecting replica ACE/Server response times and routing authentication requests accordingly

See "Setting up the Access Server as an ACE/Agent" on page 14-15 for details.

- The Access Server on Windows systems must have a certificate from the same CA root as the ACE/Server. This is not needed on Unix systems.
- The system time on the client must be correct to prevent the server and client from being out of sync.
- At least one Access Server and one WebGate must be paired for SecurID authentication.

The Oracle SecurID Access Server may have multiple WebGates that communicate with it; however, all of these WebGates must be configured to communicate with the one Oracle SecurID Access Server only.

Important: Failover is not supported for Oracle SecurID Access Servers. Only one Access Server can complete SecurID authentication.

Each Access Server installation includes the SecurID authentication plug-in, authn_securid, located in the following directory. For example, on a Windows system:

\AccessServer_install_dir\access\oblix\lib\authn_securid

This plug-in is required in the SecurID authentication scheme. See "Creating a SecurID Authentication Scheme" on page 14-21 for details about using the plug-in. See also "Oracle Access Manager Authentication Plug-In Parameters" on page 14-30.

WebGate Requirements

Each WebGate Web server used for SecurID authentication must support and pass header variables to CGI scripts, as follows:

- Each WebGate that communicates with the Oracle SecurID Access Server must be configured to communicate with this Access Server only.
- Only Oracle-provided WebGates are allowed for SecurID authentication. Do not use any other type of AccessGate.
- Lotus Domino Web servers do not pass header variables to CGIs and cannot be used for SecurID authentication.
- Older WebGates may coexist with 10*g* (10.1.4.0.1) Access Servers. However, encryption schemes differ:
  - Use RC4 as the encryption scheme if you have Oracle Access Manager Release 5.x WebGates co-existing in the same system with 10*g* (10.1.4.0.1) WebGates.
  - Use RC6 as the encryption scheme if you have Oracle Access Manager Release 6.x WebGates co-existing in the same system with 10*g* (10.1.4.0.1) WebGates.
  - Use the AES encryption scheme if you have only Oracle Access Manager Release 7.0 WebGates co-existing in the same system with 10*g* (10.1.4.0.1) WebGates.
- The Perl plug-in or programming language, v5.005_03, is required on the each WebGate host that communicates with the Oracle SecurID Access Server that validates credentials with the ACE/Server.
- A pointer to the location of Perl is required in the Oracle-provided SecurID CGI script on each WebGate involved in SecurID authentication (SecurID WebGate). See "Setting Up a SecurID WebGate" on page 14-18 for details.

Included with each WebGate installation are the Oracle-provided SecurID authentication forms in the following directories:

WebGate_install_dir\access\oblix\lang\langTag\securid-forms

WebGate_install_dir\access\oblix\lang\langTag\securid-forms-adforest

The following forms are required to validate a user's SecurID credentials and to support New PIN and New Tokencode modes.

- securid-accept-new-pin.html is not used today

- securid-enter-new-pin.html
- securid-new-pin.html
- securid-new-pin-query.html
- securid-next-tokencode.html
- securid-std-login.html

With the exception of a domain name list for the Active Directory Forest that appears on certain forms, the forms in the two directories are the same. See "SecurID Authentication Scenarios" on page 14-9 to see the forms.

Also included in the WebGate installation is the SecurID CGI script.

SecurID CGI Script

Each WebGate installation includes a SecurID CGI script in the following directory:

WebGate_install_dir\access\oblix\lang\langTag\securid-cgi

During SecurID authentication operations, the WebGate uses the CGI script to present the appropriate SecurID form to the user based on information received from the Oracle SecurID Access Server and the ACE/Agent that communicates with the ACE/Server.

See "SecurID Authentication Sequence" on page 14-9 for information on the standard SecurID login form.

See "New PIN Mode Support" on page 14-6 and"New PIN Sequence" for more information about this mode of operation.

See "Next Tokencode Sequence" on page 14-11 for more information about this mode of operation.

SecurID Authentication Scenarios

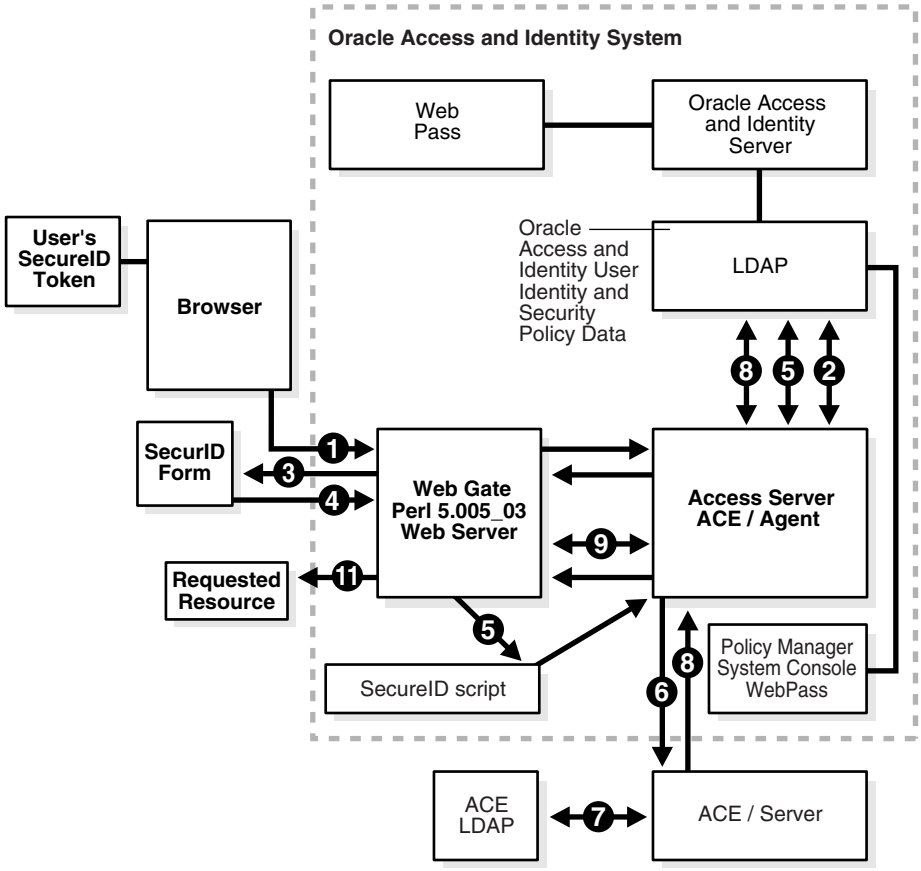
The following scenarios illustrate the three modes of operation:

- SecurID Authentication Sequence
- Next Tokencode Sequence
- New PIN Sequence

SecurID Authentication Sequence

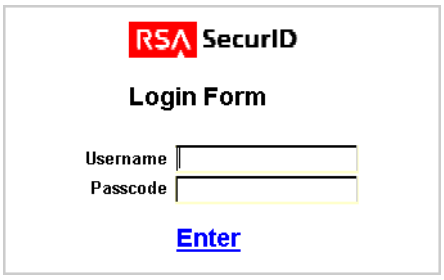
When a user attempts to access a resource protected by the SecurID authentication scheme, the following process occurs. Figure 14–1 illustrates the sequence and is followed by a detailed description.

Figure 14–1 SecurID Authentication Sequence



Process overview: When the user requests a resource

1. The WebGate intercepts the resource request and queries the Access Server to determine if and how the resource is protected, and if the user is authenticated.



2. The Access Server queries the directory for the authentication scheme, and receives authentication information from the directory.
3. The Access Server responds to the WebGate, which presents a form challenging the user for a two-part SecurID Passcode.

   See "Active Directory Forest Considerations" on page 14-33 to see the forms that include a domain for the Active Directory Forest.
4. The user submits credentials to the WebGate.
5. The WebGate presents the credentials to the Oracle SecurID Access Server.

6. The ACE/Agent on the Oracle SecurID Access Server performs the authentication dialog and sends an LDAP bind to the ACE/Server.
7. The ACE/Server database matches the SecurID passcode to the user ID and returns a success response to the ACE/Server, which matches the user's PIN.
8. The ACE/Server returns the response to its Agent, the Access Server. When the user's credentials are valid, SecurID authentication is successful.
9. The Oracle SecurID Access Server provides the response to the WebGate. A session is started for the user, so the same authentication method is not required for other Web servers in the domain. The WebGate then queries the Access Server for resource authorization:
   - Under certain conditions a New Tokencode mode may be initiated. See "Next Tokencode Sequence" on page 14-11.
   - Under certain conditions a New Pin mode may be initiated. See "New PIN Sequence" on page 14-11.
10. The Access Server queries the directory server used by Oracle Access Manager for authorization information which allows or denies access based upon the authorization rule.
11. When access is granted, the Access Server passes authorization to the WebGate, which presents the resource to the user.

The Master Access Administrator generates a shared secret key to encrypt cookies.

As discussed earlier, two modes may be triggered during the authentication sequence that will alter the user's experience. See "Active Directory Forest Considerations" on page 14-33 for the forms specific to this environment.

Next Tokencode Sequence

When Next Tokencode mode is On, the user must supply the next tokencode on their SecurID token.

Process overview: When Next Tokencode is On

1. The WebGate CGI script presents a form to challenge the user for the next tokencode on the token following a successful login.
2. The user enters a username, waits 60 seconds, then enters the next tokencode on the SecurID token.
3. When the tokencode is correct, the Passcode the user originally entered is accepted and the user is authenticated.

See "Next Tokencode Mode Support" on page 14-6 for details.

New PIN Sequence

When the user is required to have a new PIN, the WebGate prompts the user with specific forms. The sequence is for generating a new PIN is provided in the following process overviews. See "New PIN Mode Support" on page 14-6 for details.

Note: When working in new PIN mode, if the user closes the New PIN Query page that appears upon accessing a resource and immediately tries to access the same resource from another browser instance, he or she can receive an error similar to the following:

```
Oracle Access Manager Operation Error
A plugin for the authentication scheme SecurID AuthT Scheme has
aborted processing credentials (login=user123 password=(omitted)
choice= newpin=newpin2= Resource=/securid-cgi/securid.pl
RequesterIP=10.10.100.00 HostTarget=http://myhost.com:3333
Operation=POST rh=http://myhost.com:3333 ru=/test.html).
Contact your website administrator to remedy this problem.
```

In most cases, if the user waits for more than 135 seconds before accessing the resource again, the new pin query page appears as expected.

Process overview: When New PIN mode is On

1. The WebGate SecurID CGI script presents the New PIN Query form to the user, as follows.

2. The user waits for the tokencode change, then completes the form and submits it to the WebGate.
3. The WebGate presents this to the Agent on the Oracle SecurID Access Server for submission to the ACE/Server.

The result is governed by the type of PIN the user requests. In either case, the New PIN process continues.

Process overview: When the user chooses to define a new PIN

1. WebGate presents the following form so the user can enter the PIN they want.

2. The user enters a username, then waits 60 seconds and enters the new tokencode and a new PIN to complete the form.

> **Important:** The user enters the next tokencode, not the passcode.

3. The WebGate submits the information to the Agent on the Oracle SecurID Access Server, which is forwarded to the ACE/Server.
4. The ACE/Server registers the new PIN, which becomes part of the Pincode the user must supply during subsequent logins.
5. The requested resource is provided.

Process overview: When the user requests a system-generated PIN

1. The ACE/Server generates a new PIN and the WebGate presents the New PIN confirmation form to the user.

2. The user has 30 seconds to document the new PIN before the confirmation form is replaced with the following form, which prompts the user to accept the new PIN.
3. The requested resource is provided.

Integrating SecurID Authentication

Before starting the integration you must complete all prerequisites. After that, you are ready to begin integrating SecurID authentication. The following is provided as an example. Any references to specific versions and/or platforms should be validated against the latest support information. See "Supported Versions and Platforms" on page 14-5 for details.

Task overview: Integrating SecurID authentication

1. Set up your environment, as described in "Preparing Your Environment" on page 14-14.
2. Configure the Access Server, as described in "Setting up the Access Server as an ACE/Agent" on page 14-15.
3. Configure the WebGate, as described in "Setting Up a SecurID WebGate" on page 14-18.
4. Configure an authentication scheme, as described in "Creating a SecurID Authentication Scheme" on page 14-21.
5. Identify resources to be protected, as described in "Protecting SecurID Resources" on page 14-26.
6. Test the policy domain, as described in "Testing the Policy Domain" on page 14-29.
7. Add users, as described in "Adding ACE/Server Users to Oracle Access Manager" on page 14-30.

Preparing Your Environment

Each of the following steps identifies a process that must be completed before you begin integrating SecurID authentication in Oracle Access Manager.

To prepare your environment for SecurID integration

1. Ensure that your RSA ACE/Server installation includes the latest patches and is running properly.

   See "RSA ACE/Server Requirements" on page 14-5 for details. See the installation instructions for RSA ACE/Server 5.1 for details and troubleshooting.

2. Confirm that RSA SecurID user authentication has been properly integrated with your RSA ACE/Server, and add users.

   See the installation and configuration instructions for RSA SecurID for details and troubleshooting.

3. Ensure that Oracle Access Manager is set up and properly running, including the latest patches. Components include:

   - Identity Server and WebPass
   - Policy Manager and Access System Console
   - Access Server and WebGate(s)

   See the *Oracle Access Manager Installation Guide* and *Oracle Access Manager Identity and Common Administration Guide* for general information.

4. Install the Perl plug-in or programming language, v5.005_03, on each WebGate that will communicate with the Oracle SecurID Access Server.

> **Note:** The following steps must be completed only when your Oracle Access Manager installation includes an Active Directory Forest. The Oracle-provided SecurID forms for the Active Directory Forest include place holders named Domain 1, Domain 2 and Domain 3. These must be changed to valid domain names that accurately reflect your Active Directory Forest installation.

To prepare an Active Directory Forest

1. Set up and ensure that Oracle Access Manager works with your Active Directory Forest.

   See the *Oracle Access Manager Installation Guide* for details about installing and deploying with Active Directory.

2. Edit the forms to display domain names for your Active Directory Forest. For example:

   WebGate_install_dir\access\oblix\lang\langTag\securid-forms-adforest

   - securid-std-login.html
   - securid-nexttokencode.html
   - securid-enter-new-pin.html

Setting up the Access Server as an ACE/Agent

As discussed earlier, this task enables the Oracle SecurID Access Server to authenticate locally known users with the ACE/Server. The first time a user tries to authenticate through the registered Access Server, a node secret (password between the agent and the ACE/Server) is sent to the Agent in encrypted form and used to encrypt future communications.

Note: Only one Access Server may communicate with the ACE/Server. However, all Access Servers require the ACE/Agent software.

Task overview: Setting up the Access Server as an ACE/Agent

1. Register the host, as described in "Registering an ACE/Agent Host" on page 14-15.
2. Set up the host, as described in "Setting up the ACE/Agent Host" on page 14-16.

The following information focuses on successful SecurID integration with Oracle Access Manager. Providing complete details about ACE/Agents is outside the scope of this manual. See the administration guide for RSA ACE/Server 5.1 for details about ACE/Agents.

Registering an ACE/Agent Host

Before you install an ACE/Agent on the Oracle SecurID Access Server, you must add the server name, IP address, Agent and encryption types to the ACE/Server database. The Oracle SecurID Access Server must be designated as "open to all locally known users" and Agent Host auto registration should be enabled.

To register an Access Server as an ACE/Agent Host

1. Record the name and IP address of the one Access Server that will communicate with the ACE/Server for SecurID authentication.
2. Launch the ACE/Server database administration tool on the ACE/Server:
   - On Unix: sdadmin
   - On Windows: From the Start menu click Programs, then click ACE/Server Database Administration Host
3. Add your Agent Host name, IP address, Agent type, and encryption type.

For example:

```
Name: host_name
Network address: 192.168.1.140
Agent type: Unix Agent or Net OS Agent
Encryption Type: DES
```

4. Ensure that the Access Server is designated as Open to All Locally Known Users.
5. Ensure that Sent Node Secret is disabled.
6. From the System menu, select Edit System Parameters.
7. Enable the Allow Agent Host Auto-Registration parameter, then confirm changing the system parameters.

Note: Other parameters do not apply to the Access Server Agent.

The ACE/Server's sdconf.rec file contains settings for all configurable ACE/Server system parameters and Agent Host settings. By default, this file resides in the following directory.

```
/ace/data/sdconf.rec
```

You will need a copy of this file on the Oracle SecurID Access Server that communicates with the ACE/Server.

You are ready to set up the Oracle SecurID Access Server as an ACE/Agent Host.

Setting up the ACE/Agent Host

The steps in this procedure vary depending upon the platform you are using. See one of the following:

- On Unix Machines
- On Windows Machines

Path names in the following examples may not reflect the actual path names in your environment.

See "Oracle SecurID Access Server and ACE/Agent Requirements" on page 14-7 for additional information.

On Unix Machines On a Unix host, you must copy information from the ACE/Server to the Access Server. Then it is a good idea to verify the ACE/Agent installation on the Access Server, although this verification may be skipped.

To prepare a Unix-based Oracle SecurID Access Server

1. Locate and copy appropriate lines from the ACE/Server %systemroot%/drivers/etc/system to the Access Server /etc/system, for example:

Table 14–3 ACE/Server %systemroot%/drivers/etc/system to Access Server

ACE/Server added to NIS	ACE/Server not added to NIS
securid ...	set shmsys:shminfo_shmmni=100
securidprop ...	set shmsys:shminfo_shmseg=16
adlog ...	set shmsys:shminfo_shmmax=16777216
sdserv ...	set semsys:seminfo_semmni=64
sdadmind ...	set semsys:seminfo_semmsl=50
sdreport ...	set semsys:seminfo_semmns=100
sdxauthd ...	set semsys:seminfo_semmnu=100
tacacs ...	
radius ...	
radacct ...	

2. Copy or FTP the sdconf.rec file from the ACE/Server to the Access Server. For example:

   From ACE/Server: /ace/data/sdconf.rec

   To Access Server:

   /AccessServer_install_dir/access/oblix/config/securid/sdconf.rec

To verify the ACE/Agent installation on the Unix-based host (optional)

1. Locate and start the Agent installation program on the Access Server.

   ./sdsetup -agent [-p path]

2. Review configuration information and the Agent Host address field to confirm that you entered the correct hostname in the Access Server /etc/hosts file.

   The first time the client is used for authentication, the node secret will be copied into a file named securid in the VAR_ACE directory on the client. Typically this is stored in the default path /OracleAccessManager/access/oblix/config/securid.

3. Ensure the VAR_ACE environment variable is set properly.

   If it is not set properly, the node secret will not be copied the first time the client is used for authentication.

   See the Unix installation instructions for the RSA ACE/Agent for additional information and troubleshooting tips.

On Windows Machines

On the Windows-based Oracle Access Manager SecurID Access Server registered as an ACE/Agent host, you must create a root certificate to define the encryption protocol between the Agent and the ACE/Server and copy the sdconf.rec file to the Access Server before you install the ACE/Agent software on the Access Server. Complete the following procedures:

- Complete the steps in "To prepare a Windows-based Oracle SecurID Access Server" on page 14-18 for the Oracle SecurID Access Server registered as an ACE/Agent host.
- Complete the steps in "To install the ACE/Agent on each Windows-based Access Server" on page 14-18 on every Access Server.

To prepare a Windows-based Oracle SecurID Access Server

1. Install the ACE/Agent Certificate Agent utility on the Access Server using the ACE/Server CD.
2. To start the ACE/Agent Certificate utility on the Access Server, click Start, then Programs, then ACE Agent , then ACE Agent Certificate Utility.
3. Create a root certificate for the machine and make a note of where this is stored.

   New_Root

   Certificate and keys

   Host name: host_name

   Organization: Name

   Country: US

   A certificate file is created for this host and, by default, is stored in the following directory path.

   `\Program Files\SDTI\ACE Agent Certificate\host_name.crt.`

To install the ACE/Agent on each Windows-based Access Server

1. Copy or FTP the sdconf.rec file from the ACE/Server to the Access server.

   For example:

   From: `D:\ace\data\sdconf.rec`

   To: `C:\%systemroot%\system32\sdconf.rec`

   You are ready to install the ACE/Agent software on the Access Server. A Windows client requires ACE/Agent .dlls in the \WINNT\system32 directory. The minimum Agent installation is all you need to obtain the appropriate files: aceclnt.dll and sdmsg.dll.
2. Run agent.exe from the ACE/Server CD and select Common shared files and User documentation.
3. Specify the path to the root certificate you created earlier:

   `\Program Files\SDTI\ACE Agent Certificate\host_name.crt`

4. Give the path to the sdconf.rec file that you copied to this machine:

   `\%systemroot%\system32\sdconf.rec`

5. Repeat step 2 for the ACE/Server to the Access server through step 4 on each Access Server in your installation to include the aceclnt.dll and sdmsg.dll required for authn_securid plug-in initialization.

When you have the Access Server set up as an ACE/Agent you are ready to set up the SecurID WebGate(s).

Setting Up a SecurID WebGate

Before you can use SecurID authentication, you must set up the SecurID WebGate Web server to successfully locate and use the Oracle-provided SecurID forms and the CGI script.

Task overview: Setting up a SecurID WebGate

1. Relocate the directories, as described in "Relocating Oracle SecurID Directories" on page 14-19.
2. Set up the cgi script, as described in "Setting up the SecurID CGI Script" on page 14-20.
3. Configure the cgi directory, as described in "Configuring the CGI Directory" on page 14-20.

> **Note:** You must repeat the three procedures in this task on each WebGate used for SecurID authentication and ensure that each WebGate communicates with only the one Oracle SecurID Access Server.

Relocating Oracle SecurID Directories

Successful SecurID authentication requires that the three Oracle-provided securid directories installed with the WebGate are located in a directory that is configured as the Web server's document directory. This can be:

- The primary document root

or

- A virtual document root

Unless the WebGate was installed under the Web server's document root, you must relocate a copy of the Oracle-provided securid directories.

To relocate the Oracle-provided SecurID directories

1. Locate the three securid directories on your WebGate host. For example, on a Windows system:

   WebGate_install_dir\access\oblix\lang\langTag\securid-cgi

   WebGate_install_dir\access\oblix\lang\langTag\securid-forms

   WebGate_install_dir\access\oblix\lang\langTag\securid-forms-adforest

   > **Note:** If the three SecurID directories are in the Web server's primary document or a virtual document root directory, skip to "Setting up the SecurID CGI Script" on page 14-20. Otherwise, complete step 2.

2. Copy the three securid subdirectories under a Web server document directory.

   For example:

   ```
   \iPlanet\WS6sp4\docs\OracleAccessManager\securid-cgi
   \iPlanet\WS6sp4\docs\OracleAccessManager\securid-forms
   \iPlanet\WS6sp4\docs\OracleAccessManager\securid-forms-adforest
   ```

   Later, when you create a SecurID authentication scheme, you may need to adjust your scheme challenge and authn_securid plug-in parameters accordingly. See "Creating a SecurID Authentication Scheme" on page 14-21.

Setting up the SecurID CGI Script

To operate properly, the Oracle-provided SecurID CGI script must point to the correct location of Perl v5.005_03 on the WebGate.

As shown in the following paragraphs, the three SecurID directories were copied from their original installation directory to the Web server's document root using the previous steps.

To define the path to Perl

1. Open the Oracle-provided securid.pl script.

   For example:

   ```
   \iPlanet\WS6sp4\docs\OracleAccessManager\securid-cgi\securid.pl
   ```

2. Ensure that the first line points to the correct location of Perl on this WebGate.

   For example:

   ```
   #!/usr/bin/perl -w
   ```

Next, you will set up the CGI directory on the WebGate Web server.

Configuring the CGI Directory

The Oracle-provided SecurID CGI directory must be configured as a CGI directory on the Web server. This process will vary depending on your Web server platform:

- **On IIS Web servers**: You need only ensure that the Oracle-provided script is an executable.
- **On iPlanet Enterprise Web servers**: You need only configure the CGI directory as a Web server CGI directory.

  On Unix: Specify a Programs/CGI directory

  On Windows: Specify a Shell/CGI Directory
- **On Apache Web servers**: You must modify the httpd.conf file as follows:
  - Add the Oracle-provided CGI script to the AddHandler section, or uncomment it.
  - Add ExecCGI to a *Directory* container that applies to the directory where the Oracle-provided script is located.
  - Add PassEnv lines outside all *Directory* and *VirtualHost* containers.

See the documentation that accompanies your specific Web server for more information.

To configure the CGI script on IIS Web servers

1. Locate the securid.pl file.
2. Make the securid.pl file an executable.

   See the Microsoft IIS Web Server Administration documentation for details about converting the script to an executable.

To configure a CGI directory on the iPlanet Enterprise Server

1. Log in as the Web Server Administrator, then select your server name and click Manage.

2. Select the Virtual Server Class tab, click the Manage button, and then click the Programs tab.
3. Select the CGI directory for your platform. For example:

   Unix: Programs/CGI Directory

   Windows: ShellCGI Directory
4. Add the URL prefix and the full path to the CGI directory, as shown in the following Windows example:

   URL prefix: OracleAccessManager\securid-cgi

   CGI directory: C:\iPlanet\WS6sp4\docs\OracleAccessManager\securid-cgi

   See the documentation for the iPlanet Web Server Administration Server for more information.

To configure Apache Web servers for the SecurID CGI script

1. Locate the httpd.conf file and add the following information to the AddHandler section, or uncomment the line if it is already there.

   AddHandler cgi-script .pl

   Note: Be sure to include the space in AddHandler cgi-script .pl; otherwise, the Web server won't start.

2. Find a <Directory> container that applies to the directory where you have stored the securid.pl file, and add "ExecCGI" to the end of the line that reads

   "Options Indexes FollowSymLinks MultiViews," as follows:

   ```
   Options Indexes FollowSymLinks MultiViews ExecCGI
   ```

3. Add the following lines outside all <Directory> and <VirtualHost> containers.

   ```
   PassEnv HTTP_COOKIE
   PassEnv HTTP_REDIRECTURL
   PassEnv HTTP_FULLFORMDIR
   PassEnv HTTP_HTTPTYPE
   PassEnv HTTP_NEWPINRETURN
   ```

   See the Apache Web Server documentation for additional details.

Next you will create a SecurID authentication scheme that uses the Oracle-provided SecurID plug-in.

Creating a SecurID Authentication Scheme

This section provides the following topics.

- Background
- Defining an Authentication Scheme for SecurID

Even if you are familiar with Oracle Access Manager authentication plug-ins, you may want to focus on specific SecurID requirements presented earlier.

Some information offered in the following examples should be replaced with the appropriate information for your environment.

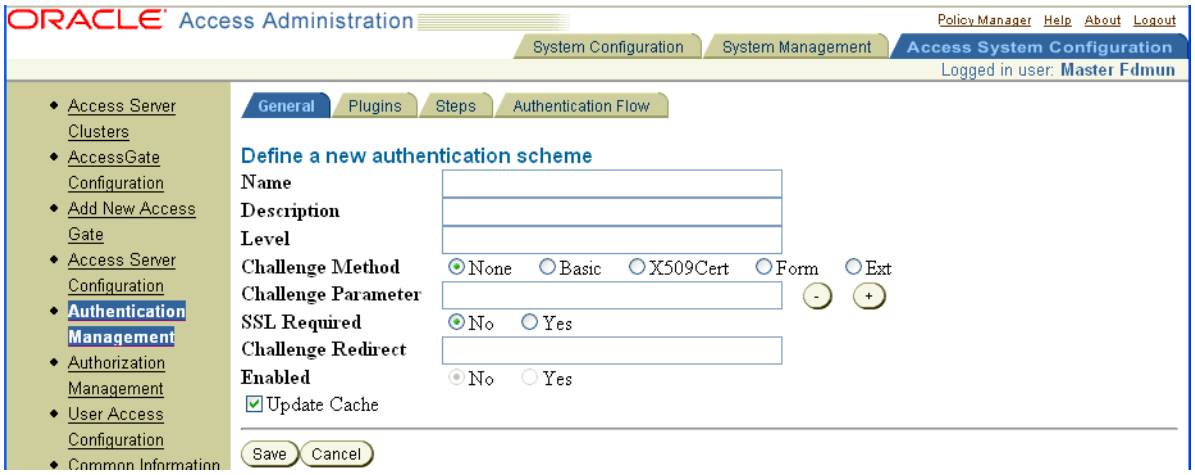
Background

The Access System protects resources according to policy domains. Each policy domain identifies the resources to be protected and must include one and only one authentication rule. That rule, which is considered the default authentication rule, must contain an authentication scheme which specifies the challenge method used to obtain credentials from the user. Each authentication scheme can include one or more plug-ins to perform additional processing. For Smart Card authentication, you must use the Client Certificate authentication scheme.

A policy domain can include policies to protect resources within the domain in a different or more specific way. Each of these policies can have its own authentication rule, but one is not required. If an authentication rule is not configured for a policy, the default authentication rule for the policy domain applies.

Until the Master Access Administrator delegates administration rights to a policy domain, he or she is the only person who can access that policy domain.

The following form is used to define the authentication scheme. This is available in the Access System Console, Access System Configuration Authentication Management function.



An authentication scheme name is required. The description is optional. The security level definition is the same for all authentication schemes. For more information about authentication schemes, see the *Oracle Access Manager Access System Administration Guide*.

The following discussions provide additional details for SecurID:

- SecurID Challenge Method
- SecurID Challenge Parameters
- SecurID Authentication Scheme Plug-Ins

SecurID Challenge Method

Each authentication scheme requires a challenge method to obtain user credentials for authentication. Only one challenge method is allowed per authentication scheme.

SecurID requires the form-based challenge method, which means that the user must complete an HTML form during the authentication process. Form-based authentication schemes can pass authorization actions in header variables but cannot pass authentication actions in header variables.

The Basic challenge method does not support SecurID Pincodes, Next Tokencode Mode, nor New PIN Mode.

See the *Oracle Access Manager Access System Administration Guide* for more information about authentication scheme challenge methods.

Note: Do not protect a challenge form or any of its components, such as .gifs and links.

SecurID Challenge Parameters

SecurID requires four challenge parameters to identify what will occur when a user logs in. The four challenge parameters are action, passthrough, creds, and form.

- **action:** You can use the action parameter to present a form to authenticate the user after receiving an initial request for a resource. For SecurID authentication, the form action is initiated by the Oracle-provided CGI script.

  The default location of the SecurID CGI script is as follows:

  WebGate_install_dir\access\oblix\lang\langTag\securid-cgi\securid.pl

  Relocated to:
  \iPlanet\WS6sp4\docs\OracleAccessManager\securid-cgi\securid.pl

  - When the script is installed on a single Web server instance, the relative path is sufficient.
  - When the script is on a different Web server instance, the full URL path is required.

- **passthrough**: Passthrough is set to No by default. For SecurID authentication, set passthrough to Yes to pass the login credentials to a post-processing program.
- **creds**: The creds parameter identifies all fields used for login in the HTML forms, in a space-separated list. The parameters needed for SecurID authentication must correspond to fields in the SecurID authentication HTML forms and SecurID plug-in parameters. For example:

  ```
  login username password passcode choice
  choice_label_such_as_system-generated newpin
  PIN_entered_by_the_user newpin2 PIN_re-entered_by_the_user
  ```

The creds challenge parameter for the user ID should match the user ID specified in the credential_mapping plug-in that is required with SecurID authentication. For example:

Challenge Parameter: creds:login

credential_mapping plug-in parameter: obMappingFilter="(&(...userid=%login%))"

- **form**: The form parameter indicates where the standard login HTML form is located relative to the Web server's document directory.

  For example, when the full path is:
  \iPlanet\WS6sp4\docs\OracleAccessManager\securid\securid-forms\securid-std-login.html

  The form parameter is: \securid-forms\securid-std-login.html

SecurID Authentication Scheme Plug-Ins

Two plug-ins are required in the SecurID authentication scheme, authn_securid and credential_mapping. Each plug-in defines how information will be looked up in the directory server. Again, the following examples illustrate concepts and may not portray your environment.

The authn_securid plug-in authenticates the user's SecurID credentials against their credentials on the ACE/Server. The two mandatory parameters include fullformdir and machine.

- **fullformdir**: This parameter identifies the full and complete path from the Web server root to the authentication form directory. For example:

  ```
  fullformdir=C:\iPlanet\WS6sp4\docs\OracleAccessManager\securid-forms
  fullformdir=C:\Webserver_root\OracleAccessManager\securid-forms-adforest
  ```

- **machine**: This parameter identifies the fully qualified machine name, including the domain name and port of the WebGate Web server instance that will communicate with the Oracle SecurID Access Server.

  ```
  machine=host_name.domain.com:port
  ```

  The credential_mapping plug-in maps the user-provided information to a valid Distinguished Name (DN) in the directory used by Oracle Access Manager. Ensure that the user ID you specify matches the creds parameter you specified in the challenge parameter.

  ```
  obMappingBase="o=company,c=us"
  obMappingFilter="(&(objectclass=...orgperson)(...userid=%login%)"
  ```

  A number of parameters are available with each plug-in. See "Oracle Access Manager Authentication Plug-In Parameters" on page 14-30 for more information. With these considerations in mind, you are ready to define the SecurID authentication scheme.

Defining an Authentication Scheme for SecurID

Only a Master Access Administrator may create authentication schemes. The following steps walk you through the process you must complete to define a SecurID authentication scheme. Differences for an Active Directory Forest are noted where appropriate. See also "Active Directory Forest Considerations" on page 14-33.

In the following example, the action URL points to the new location of the securid-cgi directory as discussed in "Relocating Oracle SecurID Directories" on page 14-19. Path names may differ in your environment.

To define the SecurID authentication scheme

1. From the Access System Console, click Access System Configuration, from Access System Configuration menu click Authentication Management
2. Click the Add button at the bottom of the panel.
3. Enter a name, optional description, and security level for your SecurID authentication scheme.

   For example:

   Name: SecurID Authentication

   Description: This scheme requires a user to enter a SecurID username (login) and passcode. This scheme also handles Next Tokencode mode and New PIN mode.

Level: An integer between 1 and 5 defines the security level.

4. Select Form as the challenge method and enter challenge parameters for this authentication scheme.

   For example:

   Challenge Method: Form

   Challenge Parameters:

   - **action**:\OracleAccessManager\securid-cgi\securid.pl
   - **passthrough**:yes
   - **creds**:login password choice newpin newpin2
   - **form excluding Active Directory Forests**: `form:\iPlanet\WS6sp4\docs\OracleAccessManager\securid-forms\securid-std-login.html`
   - **form for Active Directory Forests only**: `form:\Webserver_root\OracleAccessManager\securid-forms-adforest\securid-std-login.html`

   Next, you will create a customized challenge scheme using the authn_securid and credential_mapping plug-ins.

   The authn_securid plug-in should be the first. Be sure that the machine parameter includes the fully qualified domain name and port or the host identifier or one of its aliases as specified in the Access System Console. See "SecurID Plug-In Parameters" on page 14-30 and "Credential Mapping Plug-In Parameters" on page 14-33 for details.

5. Select No beside SSL Required.

   If you have more than one WebGate/Access Server pair, redirect to a WebGate that communicates with the dedicated Oracle SecurID Access Server. Use the fully qualified machine name. Syntax:

   `http://host_name.domain.com:port/`

   Note: When you have only one WebGate/Access Server pair, leave the Challenge Redirect field blank and skip to step 8.

6. Enter a challenge redirect, as needed for your environment.
7. Save.
8. Click the Plug-Ins tab, the Modify button, then select Custom Plugins from the drop down list and specify parameters.

   For example:

   - **Plug-in Name**: authn_securid

     Plug-in Name Parameters (excluding Active Directory):

     `fullformdir="c:\iPlanet\WS6sp4\docs\OracleAccessManager\securid-forms", machine="host_name.domain.com:port"`

     Plug-in Name Parameters for Active Directory:

```
fullfordir="c:\Webserver_
root\OracleAccessManager\securid-forms-adforest",machine="
host_name.domain.com:port"
```

9. Select the credential_mapping plug-in from the drop down list and specify parameters.

   For example:

   - **Plug-in Name**: credential_mapping

     Plug-in Name Parameters (excluding Active Directory) :
     ```
     obMappingBase="o=company,c=us",
     obMappingFilter="(&(objectclass=...orgperson)
     (...userid=%login%))"
     ```

     Plug-in Name Parameters for Active Directory:
     ```
     obMappingBase="%domain%",
     obMappingFilter="(?(objectclass=user)
     (samaccountname=%login%))"
     ```

10. Check Update Cache to have this take effect immediately, then click Save.
11. Restart the Access Server to load the plug-ins.

You have finished creating a SecurID authentication scheme that will appear in the Authentication Scheme list when you assign rules to a policy domain. See the *Oracle Access Manager Access System Administration Guide*.

Important: Before you use this scheme, the form's action URL must be protected by a Oracle Access Manager policy domain and the action challenge parameter of the form scheme must match the form action URL.

Protecting SecurID Resources

Before you can use the SecurID authentication scheme in a policy domain, you must protect the Oracle-provided SecurID CGI script specified in the action URL of your SecurID authentication scheme. Once protected, you may use the scheme in new policy domains to protect other resources with SecurID authentication.

Note: As shown in the following paragraphs, when protecting the SecurID CGI script you may use any authentication scheme except the SecurID authentication scheme. Also, do not protect the forms or any elements in the forms (.gifs, for example).

Task overview: Protecting Securid Resources

1. Create a policy domain, as described in "Creating a Policy Domain" on page 14-27.
2. Add resources to the policy domain, as described in "Adding a Resource to Your Policy Domain" on page 14-27.
3. Define rules for the domain, as described in "Defining Rules for this Domain" on page 14-28.

Creating a Policy Domain

The key to creating an effective policy domain is to group the content that you want to manage in the same way. Each policy domain is defined using the Policy Manager and each policy domain includes a definition of:

- Resources to protect
- Schemes, rules, and optional policies (exceptions) for protection
- Administrative rights, optional

For example, you need one policy domain to protect the SecurID authentication script and action URL for the Scheme. This cannot be protected by the SecurID authentication scheme. You will want another policy domain to protect resources using the SecurID authentication scheme.

The following procedure shows how to create a policy domain to protect the SecurID CGI script. Following each sequence is a brief example of a second policy domain that will use the SecurID authentication scheme to protect other resources. The information provided is a sample to illustrate concepts.

The following procedure requires specific policy domain management rights. See the *Oracle Access Manager Access System Administration Guide* for details.

To create a policy domain to protect the SecurID script

1. Launch the Access System Console:

   `http://host_name.domain.com:port/access`

2. Select the Policy Manager.

   The My Policy Domains page appears with functions on the left and current policy domains on the right.

3. Click Create Policy Domain on the left.

   The General tab is highlighted and the Name field is active.

4. Enter a name and an optional description for the new policy, then save it.

   For example:

   Name: SecurID-script

   Description: Oracle-provided

   Note: The policy domain cannot be enabled until you add a resource type.

Adding a Resource to Your Policy Domain

Only a Master Access or Master Oracle Access Manager Administrator may add resources to a policy domain. When Oracle Access Manager is initially installed, no resources are defined. Your environment may include resources.

A resource may be either static or dynamic content.

- Static content includes HTML pages, .gifs, .pdfs
- Dynamic content includes scripts, applications, EJBs

For this integration, you will add the securid-cgi as a resource for the policy domain to protect the Oracle-provided SecurID script.

The administrator who created the first policy domain set the existing root URL used as a base for the policy domain. You may append a different region to the same prefix to define a new URL prefix that is available to other policy domains.

Again, sample specifications to protect a resource with the SecurID authentication scheme are included with the following procedures. However, you cannot use the SecurID authentication scheme until you protect the SecurID script.

To add a resource to your policy domain

1. From the Policy Manager policy domain page, click the Resources tab then click the Add button and add a resource.

   For example:

   ```
   Resource Type: http
   URL Prefix (and region): / OracleAccessManager/securid-cgi
   Description: Optional
   ```

2. Ensure that Update Cache is enabled, then save.
3. Click the Save button and review the information you supplied.

Defining Rules for this Domain

All Administrators may create an authentication expression for a policy domain or policy. An existing authentication scheme must be specified as the building block.

As discussed earlier, you must protect the SecurID authentication script specified in the action URL of the authentication scheme before you can use the scheme.

When completing the following steps, skip to step 2 if the Default Rules tab is available.

To define who has access

1. From the landing page for the Access System, select the Policy Manager application, click My Policy Domains, and from My Policy Domains select Securid-script.
2. Click the Default Rules tab, then click the Add button.
3. Enter the details and confirm that you are using the Oracle Access and Identity authentication scheme.

   For example:

   Name: Securid Authentication

   Description: Optional

   Authentication Scheme: Oracle Access and Identity

4. Ensure that Update Cache is enabled, click the Save button, then review the information summary.

   When protecting the SecurID CGI script, you do not want to deny access to anyone. When protecting the SecurID CGI script, allowing everyone enables the Access Server to check each user's credentials with the ACE/Server. By default, nobody is authorized. Allowing access should take precedence.

5. Click the Authorization Rules tab, choose Oracle Authorization Scheme from the drop down list, then save.

   For example:

Authorization Scheme: Oracle Authorization Scheme

6. Enter a name, an optional description, enable this scheme, ensure that Allow takes precedence, then save.

   For example:

   Name: SecurID allow_all

   Description: Optional

   Enabled: Yes

   Allow takes Precedence: Yes

   In every case, including SecurID CGI script protection, you must specify the Role of those being granted access to this policy domain. For your environment, you may want to grant access only to specific users or groups.

7. Click the Allow Access tab, click the Add button, fill in the form, and save.

   For example, to grant access to anyone from the root directory down:

   Role: Anyone

   Rule: ldap:///**Update Cache**

   Though not required to protect the SecurID CGI script, you may apply one or more *policies* to fine-tune access control for the protected resources and apply *auditing rules* to record access requests and resource use. For these activities, you must have authorization granted by an Access Administrator.

8. Click the General tab and enable the policy domain.

   The policy domain is active and the resource is protected. In this case, the SecurID CGI script is protected by the Oracle Access and Identity authentication scheme and ready to use in policy domains that protect your resources.

9. Repeat "Protecting SecurID Resources" on page 14-26 to protect your own resources using the SecurID authentication scheme.

   See the *Oracle Access Manager Access System Administration Guide* for details about policy domains.

Testing the Policy Domain

The best way to help you identify and resolve any problems that might arise is to test the policy domain and check various log files to ensure that everything is working properly. See the appropriate manuals for your systems for details.

To enable logging and testing

1. Enable logging on your Web servers to help track any anomalies during operation.
2. Enable logging your Oracle SecurID Access Server.
3. Enable logging on your SecurID WebGate or WebGates.
4. Enable logging on your RSA ACE/Server to report activity in real time or to create an activity log, as usual.
5. Test your policy domains and Web single sign-on, as usual, to ensure that all are working as expected.

Adding ACE/Server Users to Oracle Access Manager

You must add ACE/Server users to the directory used by Oracle Access Manager to enable access to the protected resource after the user is authenticated.

One way to do this is to use the Identity System User Manager application to create a workflow definition. Creating a workflow for this purpose is no different than creating a workflow to add other users, as long as you include the following attributes as a minimum.

- Name
- Last Name
- Login
- Password

See the *Oracle Access Manager Identity and Common Administration Guide* for details about creating a workflow definition.

Oracle Access Manager Authentication Plug-In Parameters

SecurID authentication requires the SecurID plug-in and the Credential Mapping plug-in. Each plug-in provides a number of parameters to direct how information is looked up in the directory server.

This section discusses the following topics:

- SecurID Plug-In Parameters
- Credential Mapping Plug-In Parameters

SecurID Plug-In Parameters

The parameters in Table 14–4 apply to the authn_securid plug-in when defining the SecurID authentication scheme in Oracle. You may customize the parameter name according to the rules specified in the comments in Table 14–4 These parameters are case sensitive.

Table 14–4 authn_securid Plug-In Parameters

Parameter Name	Default Value	Status	Comments
httpType	http://	Optional	If the webgate doing the SecurID authentication is in SSL, the value should be changed to https:// by passing additional parameter httptype="https://"

Table 14–4 (Cont.) authn_securid Plug-In Parameters

Parameter Name	Default Value	Status	Comments
fullformdir	<none>	Mandatory	This is the full path to the SecurID forms used for authentication, from the Web root to the directory that contains the forms. The value requires a trailing slash. For example: fullformdir="C:/iPlanet/WS6sp4/docs/OracleAccessManager/securid-forms/" or, for Active Directory fullformdir="C:\Webserver_root/OracleAccessManager/securid-forms-adforest/" By default, the forms directories are installed as follows and should be moved to the Web server's document directory: \OracleAccessManager\webcomponent\access\oblix lang\langTag\securidxxx
machine	<none>	Mandatory	This is the fully qualified domain name and port number of the Web server instance that will communicate with the Oracle SecurID Access Server. For example: machine="machine.domain.com:port" ■ This name must match the host identifier specified in the Access System Console or one of its aliases. ■ If you are redirecting all SecurID authentications, this should be the Web server name that you are redirecting to.

Table 14–4 (Cont.) authn_securid Plug-In Parameters

Parameter Name	Default Value	Status	Comments
formdir	access/oblix/securid-forms	Optional	This is the relative path to the SecurID forms and requires a trailing slash. Note: If you customize this value, you must also change it in the Challenge Parameter, form, and SecurID plug-in parameter, fullformdir. In other words, if you place the SecurID forms anywhere other than: webserver_ docroot/access/oblix/securid-forms you must pass the formdir parameter to the authn_securid plug-in with the value appropriately changed. The new value should be the location of the forms relative to the doc_root. Also, the value should not include a trailing slash. Also, if the securid cgi script is not accessible at *webserver_ docroot*/access/oblix/securid-cgi/securid.pl, you must edit the various SecurID html forms to point to the correct location. You need to change the following text in each form to point to the correct location of the script. action="/access/oblix/securid-cgi/securid.pl"
username	login	Optional	If you are using the sample forms, set this value to: login. If you customize this value, you must also change it in the Challenge Parameter, creds and also in the credential_mapping obMappingFilter.
passcode	password	Optional	If you are using the sample forms, set this value to: password. If you customize this value, you must also change it in the Challenge Parameter, creds.
choiceLabel	choice	Optional	This is the name of the field in the HTML form corresponding to the user's choice of how a PIN is generated. If you customize this value, you must also change it in the Challenge Parameter, creds.
newpinLabel	newpin	Optional	This is the name of the field in the HTML form corresponding to the new PIN entered by the user. If you customize this value, you must also change it in the Challenge Parameter, creds.
newpinLabel2	newpin2	Optional	This is the name of the field in the HTML form corresponding to the new PIN that is re-entered by the user. If you customize this value, you must also change it in the Challenge Parameter, creds.

Credential Mapping Plug-In Parameters

This plug-in maps the user ID to a valid distinguished name (DN) in the directory used by Oracle Access Manager. You can configure the attribute to which the user ID is mapped to find the DN. The most common attribute that is mapped to is uid. However, it is possible to map the user ID to a profile attribute other than uid by changing the obMappingFilter parameter. See Table 14–5. These parameters are case-sensitive.

Table 14–5 Credential_mapping Plug-In Parameters

Parameter	Usage Rule	Description
obMappingBase		Defines the Base DN in the LDAP search. If omitted or empty, the directory base is used.
obMappingFilter	Mandatory	Defines the Filter in the LDAP search. This parameter enables use of the obMappingFilter term to filter for categories of end users.
obdomain	Needed with Active Directory Forests	Authenticates a user against an Active Directory Forest when the challenge method is Basic.
EnableCredential Cache		Turns off the credential mapping cache in the credential_mapping plug-in. You may want to turn off the cache if the same user ID may be mapped to different DNs. See the *Oracle Access Manager Developer Guide* for more information.

Two subordinate parameters may be used with the obMappingFilter. These parameters can only be used with the obMappingFilter parameter. See Table 14–6.

Table 14–6 obMappingFilter Subordinate Parameters

Parameter	Description
obuseraccountcontrol	When this parameter is activated, or if there is no value, two categories of end users are filtered out: ■ Those who have been added but not yet activated in the directory. ■ Those who have been deactivated but remain in the directory.
obEnableCredentialCache	Turns off the credential mapping cache in the plug-in to deactivate the user the next time they authenticate.

Active Directory Forest Considerations

The following information can be found throughout discussions in this chapter. They are repeated here for quick reference.

This section discusses the following topics:

- Prerequisites
- Integrating SecurID with an Active Directory Forest

Prerequisites

Before integrating SecurID authentication with an Active Directory Forest, you must complete the tasks in the following task overview.

Task overview: To prepare your environment

1. Complete all tasks in "Preparing Your Environment" on page 14-14 to set up the RSA ACE/Server, SecurID tokens and users, and Identity and Access Systems.
2. Set up and ensure that Oracle Access Manager works with an Active Directory Forest.

   See the *Oracle Access Manager Installation Guide* and *Oracle Access Manager Identity and Common Administration Guide* for details about installing and deploying with an Active Directory Forest.
3. Edit the following forms to replace the place holder domain names with actual domain names for your Active Directory Forest installation. For example, *WebGate_install_dir*\access\oblix\lang\langTag\securid-forms-adforest.
   - securid-std-login.html
   - securid-nexttokencode.html
   - securid-enter-new-pin.html

Integrating SecurID with an Active Directory Forest

Any differences for an Active Directory Forest are included for quick reference and are embedded in discussions elsewhere in this chapter.

In the following example, the action URL points to the securid-cgi directory as shown in "Relocating Oracle SecurID Directories" on page 14-19. Path names may differ in your environment. Bold indicates the items that are different for an Active Directory Forest.

Note: This section assumes you have completed the following tasks:

- "Setting up the Access Server as an ACE/Agent" on page 14-15
- "Setting Up a SecurID WebGate" on page 14-18

To integrate SecurID authentication

1. Follow the instructions in "Creating a SecurID Authentication Scheme" on page 14-21 with the following changes for the Active Directory Forest.

   Changes for Active Directory Forest Challenge parameters:

   - **form**: \Webserver_ root\OracleAccessManager\securid-forms-adforest\securid-std-login.html

   Changes for Active Directory Forest plug-ins and parameters:

   - **Plug-in**: authn_securid

     Parameters: fullformdir="c:\Webserver_ root\OracleAccessManager\securid-forms-adforest",machine="host_ name.domain.com:port"

   - **Plug-in**: credential_mapping

     Parameters: obMappingBase="%domain%", obMappingFilter="(?(objectclass=user) (samaccountname=%login%))"
2. Select Update Cache, then Save to complete your SecurID authentication scheme.
3. Restart the Access Server to load the plug-in.

4. Complete the tasks described in the following sections:
   - Protecting SecurID Resources
   - Testing the Policy Domain.
   - Adding ACE/Server Users to Oracle Access Manager

SecurID Forms for an Active Directory Forest

As you can see on the following Oracle-provided forms, the only difference between standard forms for SecurID authentication and the forms provided for SecurID authentication with an Active Directory Forest is the inclusion of the Domain name on the Login form, Next Tokencode form, and New PIN Entry form.

The form in Figure 14–2 includes the Domain list. See "SecurID Authentication Scenarios" for the standard forms.

Figure 14–2 Standard SecurID Login Form for Active Directory Forest

The form in Figure 14–3, "Next Tokencode Form for Active Directory Forest" includes the Domain list. See "Next Tokencode Sequence" on page 14-11 for the standard form and a description of the sequence of events that occurs in this mode.

Figure 14–3 Next Tokencode Form for Active Directory Forest

See "New PIN Sequence" on page 14-11 for the standard forms for the New PIN sequences and a description of the events that occur with this sequence. As shown in Figure 14–4, "New PIN Query", the New PIN Query form is slightly different.

Figure 14–4 New PIN Query

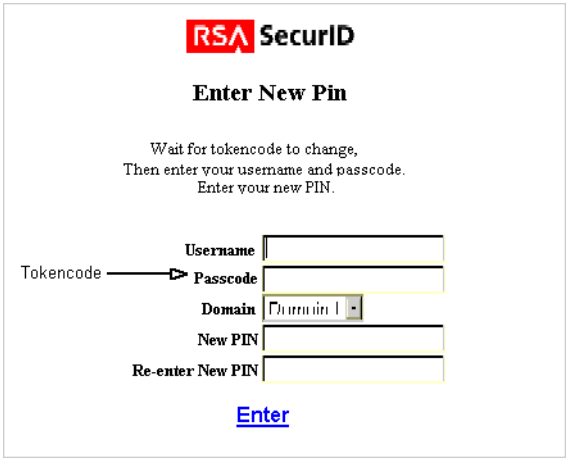
User-Defined PIN Form

If the PIN was defined by the user, they are challenged by the form and asked to enter their:

- Username
- Tokencode (shown in the form as Passcode)
- New PIN
- Active Directory Domain

The form in Figure 14–5 includes a Domain list.

Figure 14–5 New User-Specified Pin Validation Form for Active Directory Forest



Troubleshooting

Following is a brief list of some of the things to check if SecurID authentication is not working as expected.

- ACE/Agent Issues
- ACE/Server Configuration File
- "CGI Directory on SecurID WebGates"

- Environment Variable on Unix Systems
- Form-Based Authentication
- Access Server Log
- Web Server Logs
- RSA ACE/Server Logs
- Permissions
- SecurID Plug-In Parameters with Modified HTML Fields
- Login Can Fail if the Login Attribute Contains an "@" Character

ACE/Agent Issues

If an "unable to resolve" or "Error:gethostbyname failed" message appear in the CLIENT ADDRESS field when configuration data is displayed, you probably entered the client hostname incorrectly in the /etc/hosts file. In this case, you must edit the host entry to solve the problem.

If the authn_securid plug-in fails to initialize in an environment with multiple Windows-based Access Servers, verify the status of each Access Server as discussed in the following procedure.

To verify the status of each Windows-based Access Server

1. Confirm that each Windows-based Access Server in your environment has the RSA ACE/Agent software installed.

   See "Setting up the Access Server as an ACE/Agent" on page 14-15 for details.

2. Confirm that only one Access Server is registered with the RSA ACE/Server as the Oracle SecurID Access Server.

   See "Registering an ACE/Agent Host" on page 14-15 and "Setting up the ACE/Agent Host" on page 14-16 for details.

ACE/Server Configuration File

The RSA ACE/Server sdconf.rec file is required on the Oracle SecurID Access Server before you can install the RSA ACE/Agent on the Access Server.

This RSA ACE/Server file contains all configurable items for the RSA ACE/Server, including Agent Host specifications.

You must copy this file to the Access Server that will validate user credentials with the ACE/Server before you add the ACE/Agent to the Access server. See "Setting up the Access Server as an ACE/Agent" on page 14-15 for details.

CGI Directory on SecurID WebGates

Ensure that the securid-cgi directory is set up properly on the WebGate.

Task overview: Testing the securid-cgi directory

1. Unprotect a different CGI in the same directory.
2. Access the unprotected CGI to be sure you set up the CGI directory properly.

   See "Setting Up a SecurID WebGate" on page 14-18 for details.

Environment Variable on Unix Systems

When setting up your Oracle SecurID Access Server as an ACE/Agent host, ensure the VAR_ACE environment variable is properly set on your Unix system. See "Setting up the Access Server as an ACE/Agent" on page 14-15 for more information.

Form-Based Authentication

Ensure that form-based authentication is set up properly, as described in the *Oracle Access Manager Access System Administration Guide*.

If this the first time the authn_securid plug-in has been configured, you must restart the Access Server to load the plug-in.

Access Server Log

If an authentication plug-in returns an error, it is logged in the Access Server log. You configure this in the Access System Console.

To set up the Access Server log

1. From the Access System Console, click Access System Configuration, then click Access Server Configuration
2. Select the server name from the List of All Access Servers, then click Modify.
3. Set Debug On and enter a file name.
4. Restart the Access Server.

Web Server Logs

These can provide many clues as to what is going wrong. Be sure the enable logging on your Web server.

See the documentation for your Web server for details.

RSA ACE/Server Logs

If communication has been established between the Access Server and ACE/Server, the sdadmin tool provides access to logs under the Report menu. Both Activity and Exception reports may give you helpful information.

To verify the ACE/Server log configuration

1. Confirm that you have added the user and assigned a token using the ACE/Server Administrator tool, sdadmin.
2. Verify that you have copied the sdconf.rec file to the Access Server before installing the ACE/Agent.

   See "Setting up the Access Server as an ACE/Agent" on page 14-15 for details.
3. Locate the following file in the Web server's document root directory to ensure that the shared secret was downloaded on the first connection between the Oracle SecurID Access Server and the ACE/Server. For example:

   ```
   \iPlanet\WS6sp4\doc\OracleAccessManager\securid
   ```

Permissions

Permissions can sometimes cause problems.

Confirm that the following permissions are appropriate:

- On the SecurID CGI script, securid.pl
- On the SecurID HTML forms
- On all files
- On the page you are trying to reach

> **Note:** Do protect the securid.pl script on the WebGate or WebGates. Do not protect the SecurID forms or their directories.

SecurID Plug-In Parameters with Modified HTML Fields

If you have modified the HTML field names in the HTML forms, make sure you have modified the SecurID plug-in parameters to match.

Login Can Fail if the Login Attribute Contains an "@" Character

User logins can fail if you provide an at-sign ("@") in the login attribute value. If you have an at-sign in the login attribute value, you must remove the `-w` switch from the first line of the securid.pl script.

This is a known issue with SecurID.

15

Integrating Smart Cards

Smart cards enable you to support two types of credentials, as follows:

- **Something the user knows**: This is the user's secret personal identification number (PIN), similar in concept to a personal bank code PIN.
- **Something the user has**: This is a cryptographically-based identification and proof-of-possession token.

  This can be a token that is generated by a smart card device that you insert into a card reader that is attached to a computer.

You can configure a certificate-based authentication scheme in the Access System that processes an X.509 certificate-based from a smart card. This scheme can be used when a user accesses a resource and the Web server challenges the browser for an X.509 certificate.

This chapter discusses how to implement certificate-based authentication for smart card systems. If you are already familiar with certificate-based authentication in Oracle Access Manager, smart card integration is an instance of this type of authentication scheme.

This chapter discusses the following topics:

> **Note:** This chapter provides an example of configuring smart card integration using a Windows-based system. However, smart card integration can be done on any system that supports certificate-based authentication. See the chapter on configuring authentication in the *Oracle Access Manager Access System Administration Guide* for details.

About Smart Cards and the Access System

Various regulations require strong authentication for people who want to access physical resources, for example buildings and resources, as well as logical assets, for

example, Web applications that are protected by Oracle Access Manager. These regulations include the Homeland Security Presidential Directive (HSPD-12) and Signatures and Authentication for Everyone (SAFE). Special hardware tokens are often used to satisfy these regulations, including smart cards. Smart cards are also referred to as smart badges or common access cards (CACs).

Oracle Access Manager supports smart cards. When a user authenticates to a smart card application, the smart card engine produces a certificate-based authentication token. You can configure a certificate-based authentication scheme in the Access System that uses information from the smart card certificate. Certificate-based authentication works with any smard card or similar device that presents an X.509 certificate.

For example, in the case of ActivCard, smart card authentication is triggered when you do either of the following:

- Insert an ActivCard that contains a public key certificate previously issued by a Certification Authority (CA) into a reader attached to your computer.
- Request access to a resource protected by the Oracle Access Manager Client Certificate authentication scheme before inserting your ActivCard into the reader.

The first method displays a window prompting you for your PIN, rather than requesting a username, password, and domain. The second method displays a window prompting you to insert the ActivCard and provide your PIN.

Note: When you initialize a smart card, you are asked to supply a PIN. If the PIN is incorrectly entered a specific number of times, the card locks. To restore a locked certificate, either use the unlock code provided during smart card initialization or re-initialize the card.

About Oracle Access Manager Components

Within Oracle Access Manager, the Access System provides policy-based authentication, authorization, auditing, and Web single sign-on. For an overview of all Oracle Access Manager components, see the *Oracle Access Manager Introduction*.

If you are familiar with basic components of the Access System within Oracle Access Manager, integration with certificate-based strong authentication systems can be thought of as a particular case of ordinary client certificate authentication. To configure Oracle Access Manager to parse the information in a certificate presented by a strong authentication hardware token, you would do the following:

- Create a policy domain to protect various resources that are accessed by users who have a strong authentication token.
- Configure a client-certificate authentication scheme.
- In the policy domain, include a rule that makes use of the client-certificate authentication scheme.

For more information about policy domains, authentication schemes, and rules, see the *Oracle Access Manager Access System Administration Guide.*

About Client Certificate Authentication Schemes

During Access System installation, a Master Administrator can request automatic configure of a default Client Certificate authentication scheme. You can configure and modify this scheme after installation.

A user must supply a digital certificate when he or she attempts to access a resource that is protected by a policy domain that contains a client certificate authentication scheme. Oracle Access Manager supports client certificate authentication using public key encryption cryptography and X.509 certificates.

You determine how to obtain a certificate. There are no Oracle Access Manager requirements for this.

When you configure client certificate authentication, you must consider the following:

- Challenge Method, Challenge Parameter, and SSL Configuration for Smart Cards
- Plug-Ins for Certificate-Based Authentication that You Use for Smart Cards
- *Oracle Access Manager Access System Administration Guide* for details on protecting resources using policy domains.

Challenge Method, Challenge Parameter, and SSL Configuration for Smart Cards

Each authentication scheme requires a challenge method to obtain user credentials for authentication. Only one challenge method is allowed per authentication scheme. The following is required for smart card authentication:

- The X509Cert Challenge Method and X509 Challenge Parameter, which support public key encryption cryptography and X.509 certificates.
- An SSL connection.

  The X509Cert challenge method uses the Secure Sockets Layer (SSL) version 3 certificate authentication protocol (SSLv3) certificate authentication protocol built into browsers and Web servers. Authenticating users with a client certificate requires the client to establish an SSL connection with a Web server that has been configured to process client certificates.

> **Note:** Smart card authentication has no Challenge Redirect requirement.

Plug-Ins for Certificate-Based Authentication that You Use for Smart Cards

Two plug-ins supplied with Oracle Access Manager are required when you configure the Client Certificate authentication scheme for a smart card. The order of execution in the Client Certificate authentication scheme for smart card logon is as follows.

Authentication Scheme	Plug-Ins and Order of Execution
Client Certificate	1. cert_decode
	2. credential_mapping

Each plug-in defines how information is looked up in the directory server. A number of parameters are available depending upon the plug-in. For more information, see "cert_decode Plug-In" on page 15-4 and "credential_mapping Plug-In" on page 15-4.

If your certificate is stored in the browser, you can view the certificate details. For more information, including the OIDs of the attributes that are supported by the Access Server with the corresponding suffix used to retrieve the attribute, see the *Oracle Access Manager Access System Administration Guide.*

cert_decode Plug-In

The cert_decode plug-in can be used with the X509Cert challenge method. It must be included in the Client Certificate authentication scheme for smart card authentication.

The cert_decode plug-in has no parameters and does not use a data source. This should be the first plug-in in the Client Certificate authentication scheme for smart card authentication.

cert_decode decodes the certificate and extracts the components of the certificate subject's and issuer's Distinguished Name. For each component, the plug-in inserts a credential with a certSubject or certIssuer prefix. For example, if your certificates have the subject name givenName=somename, the plug-in adds the credential certSubject.givenName=somename to the credential list.

If decoding is successful, the elements of the certificate's subject and issuer DN are added to the list of credentials. If not, authentication fails.

credential_mapping Plug-In

The credential_mapping plug-in can be used with the X509Cert challenge method. It must be included in the Client Certificate authentication scheme for smart card authentication.

The credential_mapping plug-in should be second in the Client Certificate authentication scheme for smart card authentication. This plug-in maps the user-provided information to a valid Distinguished Name (DN) in the directory using the following parameters:

```
obMappingBase="ou=company,dc=yourdc,dc=yourdc,dc=com"
```

```
obMappingFilter="(&(objectclass=user=)(mail=%certSubject.E%))"
```

You can configure the attribute to which the user ID is mapped to find the DN by changing the obMappingFilter parameter as shown in the previous paragraph, where:

```
dc=the Active Directory Domain Controller
```

```
mail=%certSubject.E%=maps the email in the Active Directory to
the email in the certificate
```

See "To protect resources" on page 15-9 for details.

Integration Architecture

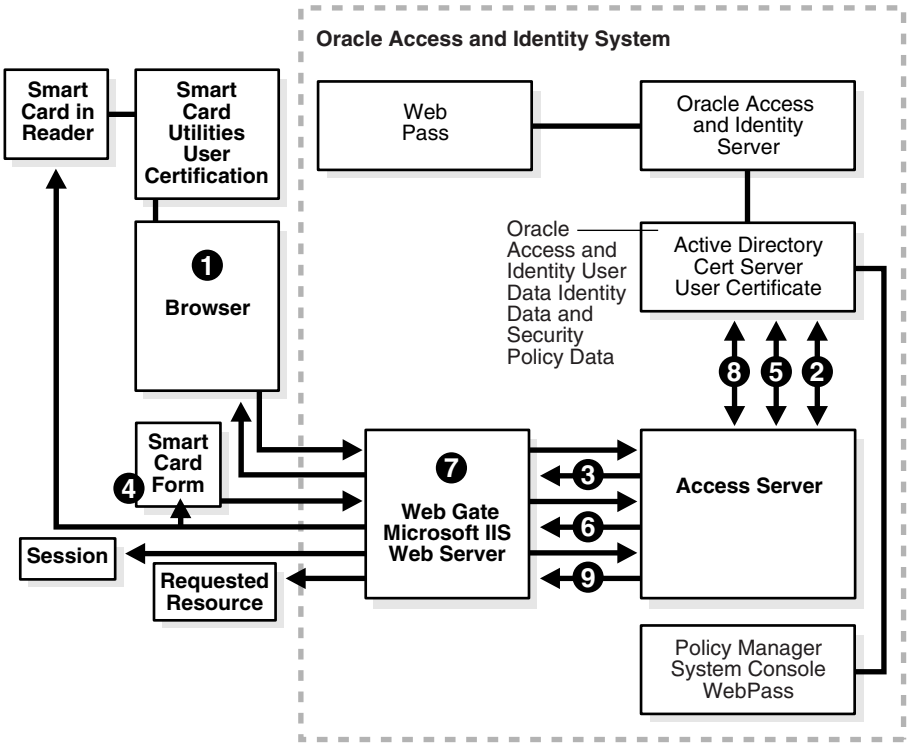
This section focuses on an example of a smart card implementation.

Example Integration Architecture: ActivCard Authentication

In this scenario, Oracle Access Manager support is shown for smart card authentication with Active Directory and IIS Web servers using ActivCard Cryptographic Service Provider (CSP) for Windows 2000, ActivCard Gold utilities, and ActivCard USB Reader v2.0 in homogeneous Windows environments.

The following process occurs during Smart Card authentication with Oracle Access Manager. Figure 15–1 illustrates the sequence and is followed by a process overview.

Figure 15–1 ActivCard Authentication Sequence



Process overview: Smart Card authentication in the ActivCard example

1. The browser prompts the user for the smart card and the WebGate intercepts the user's resource request and queries the Access Server to determine if and how the resource is protected, and if the user is authenticated.
2. The Access Server queries the Active Directory server for authentication information and receives information from the directory.
3. The Access Server responds to the WebGate, which prompts the browser to challenge the user to either insert their ActivCard and/or enter their PIN.
4. The user submits their credentials, which the browser passes to the WebGate and the WebGate presents to the Access Server, at which point one or more authentication plug-ins are used.

   The cert_decode and credential_mapping plug-ins are required with the Client Certificate authentication scheme.
5. The Access Server performs the authentication dialog with the Active Directory, which maps the certificate information stored in the smart card to the user certificate in the directory and returns a success response to the Access Server.
6. When the user's credentials are valid, the Access Server provides the response to the WebGate, which starts a session for the user.
7. The WebGate queries the Access Server for resource authorization.
8. The Access Server queries Active Directory for authorization information that allows or denies access based upon the policy domain's authentication and authorization rules.
9. When access is granted, the Access Server passes authorization to the WebGate, which presents the resource to the user.

Supported Versions and Platforms

Oracle Access Manager supports certificate-based authentication, and as a result, can be integrated with any certificate-based smart card or similar strong authentication token.

Examples of Setting Up Smart Card Authentication

Several procedures must be completed to set up smart card authentication with Oracle Access Manager. The following sections provide examples of how you would go about this task.

Setting Up Smart Cards for ActivCard

The following sections describe how you would configure client-certificate authentication in an ActivCard environment.

Task overview: Setting up smart card authentication

1. Confirm your environment meets requirements in "Supported Versions and Platforms" on page 15-6.
2. Set up Active Directory, as described in "Preparing Active Directory" on page 15-6.
3. Set up a certificate, as described in "Preparing the CA and Enrolling for a Certificate" on page 15-7.
4. Set up the IIS Web Servers, as described in "Preparing IIS Web Servers" on page 15-7.
5. Set up Oracle Access Manager, as described in "Preparing Oracle Access Manager for Smart Card Authentication" on page 15-8.
6. Configure your protected resources, as described in "Protecting Resources with Oracle Access Manager" on page 15-8.
7. Set up IIS Manager, as described in "Setting Up the IIS Manager" on page 15-10.

Preparing Active Directory

The following sections discuss preparing Active Directory.

> **Tip:** For more information about this procedure, see the Active Directory manual.
>
> For details about setting up your Active Directory to operate with Oracle Access Manager, see the *Oracle Access Manager Installation Guide* and *Oracle Access Manager Identity and Common Administration Guide*.

To prepare Active Directory

1. Ensure that you have a domain controller and Active Directory installed and properly running.
2. Ensure that you have a Domain Name System (DNS) server installed and properly running.

Note: You must install a Microsoft certification server with Active Directory, as discussed next.

Preparing the CA and Enrolling for a Certificate

The following sections discuss preparing the CA and enrolling for a certificate.

See also: See the ActivCard documentation, *Configuring Smart Card logon with ActivCard CSP for Windows 2000* for details.

To prepare a certification authority

1. Confirm that you have met all setup requirements for certification authorities (CAs), installed ActivCard Gold utilities, and set up the Certificate Authority (CA).

   If you want to install the user's certificate on the ActivCard only, rather than on both the computer and the ActivCard, you need at least two installations of the ActivCard Gold utilities because you need an administrator's certificate to digitally sign the user's certificate.

2. Establish the certificate types that an enterprise CA can use.
3. Prepare a CA to issue smart card certificates.

To complete smart card certificate enrollment

1. Prepare a smart card certificate enrollment station on a computer that you will use to set up smart cards and install a ActivCard USB reader v2.0.

   If you want the user's certificate installed on the ActivCard only, rather than on both the computer and the ActivCard, you need multiple ActivCard USB Readers and at least two ActivCard Gold.

2. Connect a smart card reader.
3. Enroll for a Smart Card Logon or Smart Card User certificate, initialize the card, and digitally sign the request.

   For more information about downloading certificates onto ActivCards, see the *ActivCard Gold User Guide*.

4. Log on with an ActivCard, as described in *Configuring Smart Card logon with ActivCard CSP for Windows 2000*.
5. Set policies for smart card removal behavior.

Preparing IIS Web Servers

The following sections describe preparing IIS Web Servers.

Tip: For more information about the following tasks, see the ActivCard documentation, *Configuring Smart Card logon with ActivCard CSP for Windows 2000*.

To prepare the IIS Web server for certification authentication

1. Deploy a certificate and the CA that issued the certificate within IIS on the Web server that hosts the WebGate.

2. Enable SSL to protect communication on port 443 on the Web server that hosts the WebGate.
3. Enable client certificate authentication within IIS.
4. Download a 1024-bit-length Web server certificate from your Microsoft certificate server.

Note: Do not use a 512-bit-length certificate.

Preparing Oracle Access Manager for Smart Card Authentication

The following sections describe preparing Oracle Access Manager for smart card authentication.

Tip: For more information, see the *Oracle Access Manager Installation Guide*

To prepare Oracle Access Manager for smart card authentication

1. Ensure that Oracle Access Manager is properly installed and running with Active Directory, including the latest patches, for example:
   - Identity Server and WebPass
   - Policy Manager and Access System Console
   - Access Server and WebGates
2. Confirm that SSL is enabled on the IIS Web server hosting the WebGate.

Protecting Resources with Oracle Access Manager

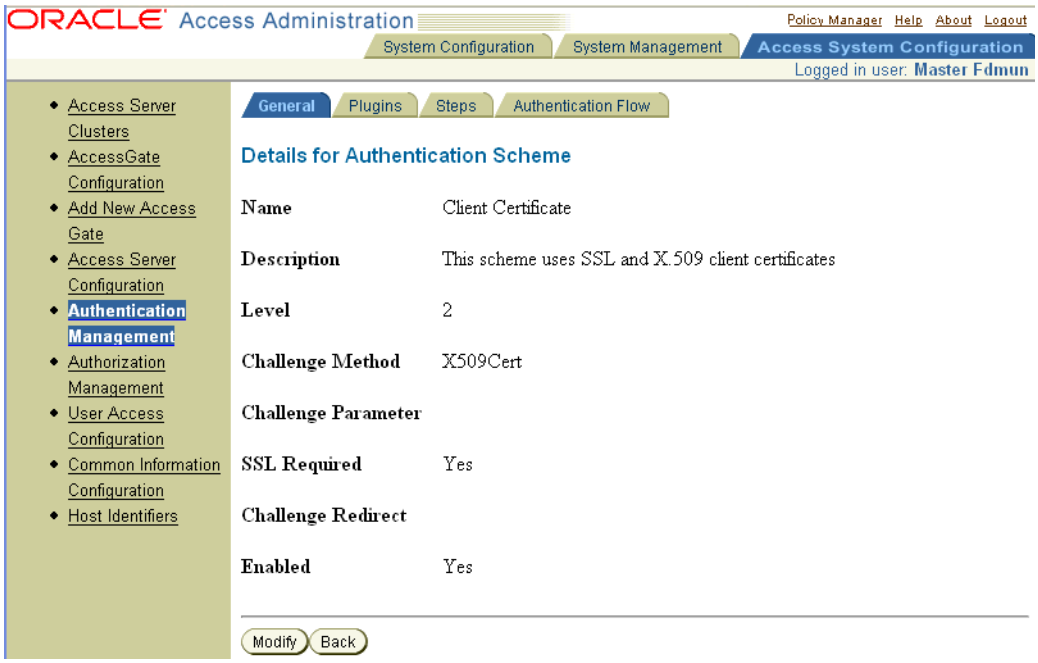
You need to modify the Client Certificate authentication scheme and add it to a policy domain to protect resources for smart card authentication.

Steps are provided in this procedure. For additional information, see the *Oracle Access Manager Access System Administration Guide*

To configure the authentication scheme for smart card

1. Navigate to the Access System Console, Access System Configuration tab, Authentication Management function.
2. Create or modify the Client Certificate authentication scheme to use the X509Cert challenge method, as shown in the example in Figure 15–2.

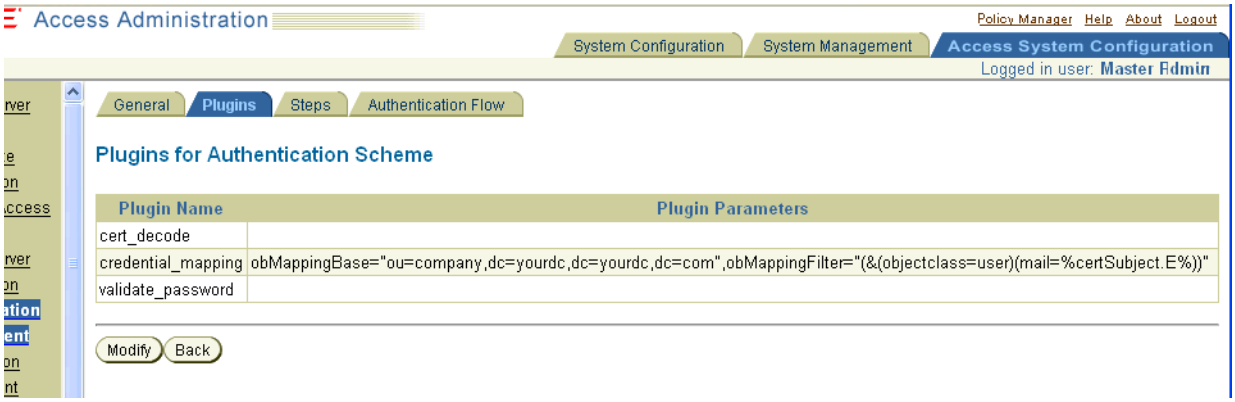
Figure 15–2 Client Certificate Authentication Scheme for Smart Card



3. Click the Plug-Ins tab and ensure that the cert_decode and credential_mapping plug-ins contain appropriate parameters and values for smart card authentication, as shown in the example in Figure 15–3.

   For more information, see "About Client Certificate Authentication Schemes" on page 15-2.

Figure 15–3 Smart Card Authentication Scheme Plug-In Parameters



   This scheme will appear in the Authentication Scheme list when you add authentication rules to the policy domain.

Next, you create a policy domain in the Policy Manager, as described in the following sections.

To protect resources

1. Navigate to the landing page for Access System administration:

   `http://hostname:port/access/oblix`

2. Select the Policy Manager application, and click Create Policy Domain in the left navigation pane.

   For example:

   ```
   Name—Your Choice.
   Description—Optional
   ```

   Note: Do not enable the policy domain until all specifications are completed.

3. Click Save.
4. Click the Resources tab, then click Add and add a resource.

   For example:

   ```
   Resource Type—Your Choice
   URL Prefix—Your Choice
   Description—Optional
   ```

5. Click Save.
6. Click Authorization rules, and configure those that apply to your policy domain and resource, then confirm or add plug-in parameters, as usual.
7. Click the Default Rules tab, click the Add button, enter the details for the authentication rule and confirm that you are using the modified Client Certificate authentication scheme.

   For example:

   ```
   Name—Your choice
   Description—Optional
   Authentication Scheme—Client Certificate
   ```

8. Add an access policy, as needed.

   Delegating Administration is done as usual. There are no special requirements. For more information, see the *Oracle Access Manager Identity and Common Administration Guide*.

9. Click the General tab and enable the policy domain, as usual.
10. Continue with "Setting Up the IIS Manager" on page 15-10.

Setting Up the IIS Manager

Next you must configure the Oracle Access Manager cert_authn.dll to "accept cookies", in the Internet Services Manager.

To configure the cert_authn.dll

1. Navigate to the Internet Services Manager by clicking Start, then Programs, then Administrative Tools, then Internet Services Manager.
2. Expand the host, double click the Default Web Site (or another Web site if you are not using the default), then navigate to and double-click the cert_authn.dll.

   For example:

   ```
   hostname > Default Web Site
   access\oblix\apps\webgate\bin\cert_authn.dll
   ```

Note: If the ISAPI WebGate installation configuration is performed manually, the following information will be presented on an HTML page:

"If you are using client certificate authentication you must enable client certificates for the WebGate and SSL must be enabled on the IIS Web server hosting the WebGate. Once this is done, do the following steps to enable client certificates for the WebGate:"

3. Select the File Security tab, then click Edit in the Secure Communications panel at the bottom of the window: File Security, Secure communications Edit.
4. In the Client Certificate Authentication subpanel, enable Accept Certificates.
5. Click OK in the Secure Communications window, and click OK in the cert_ authn.dll Properties window.

Troubleshooting

This section discusses the following troubleshooting tips for smart card authentication:

Problem Requesting X.509 Certificates

Oracle Access Manager requires X.509 certificates from Microsoft's Certification Server on Windows 2000 to be downloaded to the smart card. In this case, you need the ActivCard Gold for authentication.

Problem

You request a certificate for smart card from the following Web page:

`http://`*`hostname`*`/cersrv/certsces.asp`

You see the message "Downloading ActiveX Controls..." yet never complete the process.

Solution

1. Visit the following Web page:

   `http://www.microsoft.com/windows2000/downloads/critical/q323172/default.asp`

2. Obtain security patch Q323172 for certificate downloads with IIS.

Additional Resources

There are several sources of information that you may find useful when setting up smart card authentication for Oracle Access Manager 10*g* (10.1.4.0.1).

Active Directory Resources

For more information about setting up Active Directory, see:

- Microsoft Active Directory documentation
- *Oracle Access Manager Installation Guide* chapter on installing on Active Directory
- *Oracle Access Manager Identity and Common Administration Guide* for details on deploying with Active Directory

Smart Card Resources

For more information about setting up ActivCard utilities and the smart card, see the documentation that accompanies your ActivCard product packages, including:

- ActivCard Gold User Guide
- ActivCard: Configuring smart card logon with ActivCard CSP for Windows 2000
- ActivCard Trouble Shooting Guide

For general information about smart cards, see:

- Microsoft Step-by-Step Guide to Installing and Using a Smart Card Reader
- Microsoft Step-by-Step Guide to Mapping Certificates to User Accounts

Oracle Access Manager Policy Domain Details

For more information about setting up protecting resources with Oracle Access Manager policy domains, see the *Oracle Access Manager Access System Administration Guide.*

16

Single Sign-On for Lotus Domino

Lotus Domino is a server platform for messaging, collaboration, and applications. You can configure Domino impersonation that is similar to Windows impersonation on IIS.

This chapter discusses the following topic:

- Configuring Single Sign-On for Lotus Domino

Configuring Single Sign-On for Lotus Domino

By setting the remote_user header to the name of the authorized user using standard actions, you can configure Domino impersonation that is similar to Windows impersonation on IIS.

Domino uses its own user store. To provide single sign-on between the Access System and Domino, the Access System passes a header variable, remote_user, that contains the name of the user as it is contained in the Domino user store. The Access System looks up the user in the Domino user store, using both the long and short name stored there, and uses the preferred name defined by the Domino instance in the remote_user header.

> **Note:** On Lotus Domino v6, be sure that the Anonymous authentication radio button on the server/ports/internet ports/web page tab is disabled.

To configure single sign-on using a Lotus Domino Web server

1. Create an authorization rule, as described in the chapter on configuring authorization in the *Oracle Access Manager Access System Administration Guide.*
2. In the General screen displaying the authorization rule, click Actions.

   The Actions page appears.
3. Click Add.
4. Under Authorization Success:
   a. Type headervar in the first Type field.
   b. Type remote_user in the Name field.
   c. In the Return Attribute field, type the name of any attribute that identifies the user.
5. Click Save to save your changes (or click Cancel to exit the page without saving).

17

Integrating SharePoint Server

SharePoint Portal Server is a secure, scalable, enterprise portal server. This chapter explains how to integrate with the Microsoft SharePoint Portal Server 2003 and SharePoint Office Server 2007. It covers the following topics:

About Oracle Access Manager and the SharePoint Server

Oracle Access Manager provides identity management and security functions, including Web-based single sign-on, user self-service and self-registration, user provisioning, reporting and auditing, policy management, dynamic groups, and delegated administration. Oracle Access Manager integrates with all leading directory servers, application servers, Web servers, and enterprise applications.

SharePoint Portal Server is a secure, scalable, enterprise portal server that builds on Windows Server 2003 Microsoft Internet Information Services (IIS) and Windows SharePoint Services (WSS). SharePoint Portal Server can aggregate SharePoint sites, information, and applications into a single, easy-to-use portal. In addition to WSS functionality, SharePoint Portal Server incorporates additional features such as News and Topics as well as personal and public views for My Site, and so on.

SharePoint Office Server 2007 enhances control over content, business processes, and information sharing. Office SharePoint Office Server 2007 provides centralized access and control over documents, files, Web content, and e-mail, and enables users to submit files to portals for collaborative work.

When Oracle Access Manager is integrated with SharePoint Portal Server 2003 or SharePoint Office Server 2007, the Access System handles user authentication through an ISAPI filter for IIS and an ISAPI wildcard extension, which enables single sign-on between Oracle Access Manager and SharePoint Portal Server. WSS handles resource request authorization for all SharePoint Portal Server resources.

This integration provides single sign-on to SharePoint Portal Server 2003 and SharePoint Office Server 2007 resources and all other Oracle Access Manager-protected resources.

About Windows Impersonation

This integration relies on Windows impersonation, which enables a trusted user in the Windows server domain to assume the identity of any user requesting a target resource in SharePoint Portal Server 2003 or SharePoint Office Server 2007. This trusted impersonator maintains the identity context of the user while accessing the resource on behalf of the user.

Impersonation is transparent to the user. Access appears to take place as if the SharePoint resource were a resource within the Access System domain.

Supported Platforms and Requirements

Successful integration with SharePoint Portal Server 2003 or SharePoint Office Server 2007 requires both Oracle Access Manager and Microsoft components, which must be installed and configured to support impersonation as well as integration.

This section contains the following topics:

Supported Versions and Platforms

Any references to specific versions and platforms in this chapter are for demonstration purposes.

You can find support and certification information at the following URL:

`http://www.oracle.com/technology/documentation/`

You must register with OTN to view this information.

Also, you can see the supported versions and platforms for this integration on Metalink, as follows.

To view information on Metalink

1. In your browser, enter the following URL:

   `https://metalink.oracle.com`

2. Log in to MetaLink.
3. Click the **Certify** tab.
4. Click **View Certifications by Product**.
5. Select the **Application Server** option and click **Submit**.
6. Choose **Oracle Identity Management** and click **Submit**.
7. Click **Oracle Identity Management Certification Information 10g (10.1.4.0.1) (html)** to display the Oracle Identity Management page.
8. Click the link for **Section 6, Oracle Access Manager Certification** to display the certification matrix.

Required Microsoft Components

Table 17–1 lists the Microsoft components required to integrate with SharePoint Portal Server 2003 and SharePoint Office Server 2007.

Note: Be sure to install SP1 or later to have all critical updates.

Table 17–1 Microsoft Requirements

Component	Description
Operating System	Windows Server for the SharePoint Server host. **Note**: Any of the five editions is acceptable. The Active Directory Domain Controller must reside on a Windows Server 2003 computer. This computer does not have to be the SharePoint Portal Server host.
Extended Services	■ Internet Information Services (IIS) 6.0. You must install IIS on the host computer for SharePoint Server after installing Windows Server 2003 on that computer. ■ Windows SharePoint Services (WSS) 2.0. These services install automatically when you install the SharePoint Server. See SharePoint Server in this table.
Directory Service	Active Directory. You install this after installing Windows Server 2003. You can connect a SharePoint Server directly to the Active Directory Domain Controller or to a different instance of Active Directory, as follows: ■ The Active Directory Domain Controller can reside on the same computer as your SharePoint Server installation. If it does, SharePoint Server requires an instance of SQL Server (not Desktop SQL) installed on a computer in the Active Directory domain. ■ Alternatively, you can connect SharePoint Server to an Active Directory Domain Controller residing on a different Windows Server 2003 machine. ■ Another option is to connect SharePoint Server to a non-domain controller instance of Active Directory, which can reside on the machine hosting SharePoint Server or on any other machine in your Active Directory domain. **Note**: For all scenarios mentioned here, SharePoint Server can use either Desktop SQL or an instance of SQL Server installed on a machine within the Active Directory domain.
SharePoint Server	SharePoint Server. After installing Active Directory, you install SharePoint Portal Server 2003 or SharePoint Office Server 2007 on a computer where Windows Server 2003 and IIS are already installed.
Security Service	Kerberos Key Distribution Center (KDC) installs automatically as part of Windows Server 2003.

Required Oracle Access Manager Components

The components in Table 17–2 are required to integrate with SharePoint Server and SharePoint Portal Server.

Table 17–2 Component Requirements

Item	Requirement/Description
WebGate	The ISAPI version WebGate that ships with Oracle Access Manager must reside on the same machine as SPPS. Within the context of the SPPS integration, this WebGate is an ISAPI filter that intercepts HTTP requests for Web resources and works with the Access Server to authenticate the user who made the request. If authentication is successful, the WebGate creates an ObSSOCookie and sends it to the user's browser, thus facilitating single sign-on. The WebGate also sets impersonate as a HeaderVar action for this user session.
IISImpersonationExtension. dll	This dll is an IIS wildcard extension that checks whether the Authorization Success Action HeaderVar has been set to impersonate, and if it has been, the dll creates a Kerberos U4S2Self ticket so that the special trusted user in the SPPS Active Directory can impersonate the user who originally made the request. When you run the ISAPI WebGate installation wizard, IISImpersonationExtension.dll installs automatically within the WebGate directory structure, but you still need to configure the dll manually to enable impersonation and SPPS integration.
Oracle Access ManagerDirectory	Oracle Access Manager can be connected to any supported directory service including, but not limited to, LDAP and Active Directory. It can even connect to the same instance of Active Directory used by SPPS. In any case, the directory does not have to reside on the same machine as SPPS and the WebGate protecting it.
Other Components	The SPPS integration also requires installation of the other standard Oracle Access Manager system components such as the Access Server with which the WebGate protecting your SPPS installation is configured to interoperate. For details see the *Oracle Access Manager Installation Guide*. Except for the WebGate protecting SPPS, your components do not need to reside on the machine hosting SPPS. **Note** that if you install either Policy Manager or WebPass (or both) on the same IIS virtual server as SPPS, you must exclude the URL paths to those components through SharePoint Managed Paths. (See "To define managed paths in SharePoint" on page 17-9 for details). You may find it easier to install Policy Manager and WebPass on a machine other than the one on which SPPS resides or, at the very least, on an IIS virtual server other than the one on which SPPS has been installed.

Request Processing by the SharePoint Portal Server Integration

Oracle Access Manager uses the Windows impersonation feature to facilitate user access to SharePoint Portal Server resources.

Process overview: Request processing with the SharePoint Portal Server integration

1. The user requests access to an SharePoint Portal Server resource.

2. The WebGate ISAPI filter protecting SharePoint Portal Server intercepts the request, determines whether the target resource is protected, and if it is, challenges the user for authentication credentials.
3. If the user supplies credentials and the Access Server validates them, the WebGate sets an ObSSOCookie in the user's browser, thus enabling single sign-on.

   The WebGate also sets an HTTP header variable called "impersonate", whose value is set to the authenticated user's LDAP uid (or samaccountname, if the user account exists in Active Directory, or userPrincipalName, if the user account exists in a multi-domain Active Directory forest).

   Note: At this point, IIS considers the user to be anonymous, since the impersonation has not yet been set.

4. The Oracle Access Manager ISAPI wildcard extension IISImpersonationExtension.dll checks for the Authorization Success Action header variable named impersonate.

   When such a header variable exists, the wildcard extension obtains a Kerberos ticket for the user. This Service for User to Self (S4U2Self) impersonation token enables the designated trusted user to assume the identity of the requesting user and obtain access to the target resource through IIS and SharePoint Portal Server.

Integrating with SharePoint Portal Server 2003

Any references to specific versions and platforms in this chapter are made for demonstration purposes. For complete support information, see the Certify tab at `https://metalink.oracle.com`.

There are several phases to integrate with the SharePoint Portal Server.

Task overview: Integrating with SharePoint Portal Server

1. Install the Microsoft components, as described in "Installing Microsoft Components" on page 17-5.
2. Install Oracle Access Manager, as described in "Installing Oracle Access Manager Components" on page 17-8.
3. Set up impersonation, as described in "Setting Up Impersonation" on page 17-13.
4. Complete the integration, as described in "Completing the SharePoint Server Integration" on page 17-21.

Installing Microsoft Components

Except where noted, all Microsoft SharePoint Portal Server-related components must be installed on the same host machine, including the following software:

- Windows Server 2003
- Microsoft IIS v6 Web Server
- SharePoint Portal Server (and underlying WSS)

The following Microsoft components can be installed on machines other than the one hosting the main SharePoint Portal Server installation:

- Active Directory (see Table 17–1 on page 17-3 for installation location details).

- SQL Server must be installed on a machine in the Active Directory domain only if the Active Directory Domain Controller is installed on the same machine as SharePoint Portal Server. See Table 17–1 on page 17-3 for installation location details.
- Additional SharePoint Portal Server front end servers.
- One or more back end servers containing SharePoint Portal Server resources such as Web pages, documents, or applications.

> **Note:** All of the machines hosting the preceding SharePoint Portal Server-related components must be in the same Active Server domain as the SharePoint Portal Server server you are installing.

The following task overview includes references to documentation that provides procedures and steps you need to complete when installing the Microsoft components for this integration.

Task overview: Installing Microsoft Components

1. Install Windows Server 2003 on the machine that will host your SharePoint Portal Server installation, as described in the appropriate Microsoft documentation.
2. Install IIS on the machine that will host your SharePoint Portal Server installation, as described in the appropriate Microsoft documentation.
3. Install Active Directory, as described in the appropriate Microsoft documentation and the *Oracle Access Manager Installation Guide*; also, see Table 17–1 for installation location considerations.
4. If SharePoint Portal Server and Active Directory Domain Controller reside on the same machine, you must also install Microsoft SQL Server on that machine as well, as described in Table 17–1.
5. Install SharePoint Services and SharePoint Portal Server on the root virtual server (which uses port 80, by default) of your IIS installation or on some other IIS virtual server.
6. Create and set up your portal.
7. After you install SharePoint Portal Server, stop and test the installation to ensure it operates correctly before you integrate with Oracle Access Manager.

Task Overview: Creating and setting up a Sharepoint portal

1. Create a portal.

   See "To create a portal" on page 17-7 for details.
2. Upload a test document.

   See "To upload a document to the portal" on page 17-7 for details.
3. Create audiences.

   See "To create audiences" on page 17-7 for details.
4. Edit audiences (if necessary).

   See "To edit audiences" on page 17-8 for details.
5. Compile audiences.

   See "To compile audiences" on page 17-8 for details.

To create a portal

1. In the Portal Site and Virtual Server Configuration section of the SharePoint Portal Server Central Administration page for server on which you wish to create a portal, click Create a portal site.
2. In the Portal Creation Options section, click Create a portal.
3. In the Site Name section, in the Name box, type a name for the portal site. This name will appear at the top of most pages for the portal site.
4. In the Virtual server list within the Site URL section, click the existing virtual server on the server that will host the portal site.
5. In the URL box, type the URL through which users connect to the portal site. By default, this URL is http://*server_name*/. If you choose a virtual server that has a port number other than 80, the port number appears as part of the URL, that is, http://*server_name:port_number*/.

   Note: Make sure to specify the load-balanced URL, not the local server URL.

6. In the Account name box of the Owner section, type the account name of the portal site owner in the format *Domain\user_name*. The portal site owner manages content and user access.
7. In the E-mail address box, type the e-mail address for the portal site owner, then click OK.
8. On the Create Portal Confirmation for *server_name* page, click OK to begin creating the portal site.
9. The Operation Status page displays as the portal is created. Following successful portal site creation, the Operation Successful page appears. At this point, you can begin detailed configuration of the portal site.

To upload a document to the portal

1. Using your Web browser, navigate to the home page for the portal.
2. Select Upload Document from the Actions list.
3. On the Upload Document page, click Browse, navigate to the document you wish to add, then click Open.

   To add multiple documents simultaneously, click Upload Multiple Files.

   To replace a file of the same name within the library, select the checkbox titled "Overwrite existing file(s)?"
4. Click Save, then click Close.

To create audiences

1. Audiences, which are based on jobs or tasks within an organization, match specified users to target content while preventing all other users from viewing that content. On the Managing Audiences page for the site you wish to configure, click Create audience.
2. On the Create Audience page, type a name and description for the audience.
3. Click either "Satisfy all rules" or "Satisfy any of the rules," then click OK.

4. After the Add Audience Rule page appears, add whatever rules you wish to govern access to the site content. (You can also add rules through the View Audience Properties page.) For details, consult the Microsoft SharePoint Portal Server documentation on Adding and Editing Audience Rules.
5. Compile the audience so that the content is targeted to that audience. See "To compile audiences" on page 17-8.

To edit audiences

1. On the View Audience Properties page for the site you are configuring, click View Audience Properties, then click Edit audience.
2. On the Edit Audience page, change the name or description of the audience, as necessary.
3. Click either "Satisfy all rules" or "Satisfy any of the rules," then click OK.
4. When the View Audience Properties page reappears, Add, Delete, or Edit the audience rules, as necessary.
5. Review the statistics for the audience, checking the number of current members and the most recent time of compilation. When you are satisfied with all the settings for the audience and the rules associated with that audience, compile the audience so that your changes take effect. See "To compile audiences" on page 17-8.

To compile audiences

1. Any changes you make to an audience or the rules associated with them do not take effect until you compile the audience. Navigate to the Manage Audiences page and check the compilation status and most recent compilation time for the audience you wish to compile. (You can also view the number of incomplete audiences on this page).
2. Either start a compilation or set a compilation schedule.

Installing Oracle Access Manager Components

The ISAPI Webgate for SharePoint Portal Server must be installed on the same machine as SharePoint Portal Server. The rest of your installation can reside on the same machine or any other machine.

If you choose to install on a different machine (which can be a Solaris, Linux, or Windows machine), it can be set up for Active Directory (if the host machine runs Windows Server 2003) or some other directory service, such as NetScape Directory Server (if the machine runs Solaris or Linux, for example).

If both Oracle Access Manager and SharePoint Portal Server are set up for different instances of Active Directory, both instances must belong to the same Active Directory domain.

To install Oracle Access Manager components for SharePoint Portal Server integration

1. On either the same machine that hosts SharePoint Portal Server (or on a different machine), install an Identity Server and a WebPass, then set up the Identity System as described in the *Oracle Access Manager Installation Guide* and see Table 17–2 for WebPass installation considerations.

2. On either the same machine that hosts SharePoint Portal Server (or a different machine), install Policy Manager and one or more instances of the Access Server, as described in the *Oracle Access Manager Installation Guide* and Table 17–2.
3. On the machine hosting the SharePoint Portal Server instance you are trying to integrate, install an ISAPI WebGate.

   The IISImpersonationExtension.dll will be installed as part of the package in the following directory:

   ```
   WebGate_install_dir\access\Oblix\apps\webgate\bin\
   ```

   Where *WebGate_install_dir* is the directory where you installed the WebGate.
4. If you installed Policy Manager or WebPass on the same IIS virtual server as SharePoint Portal Server, complete activities in "Defining Managed Paths in SharePoint" on page 17-9.

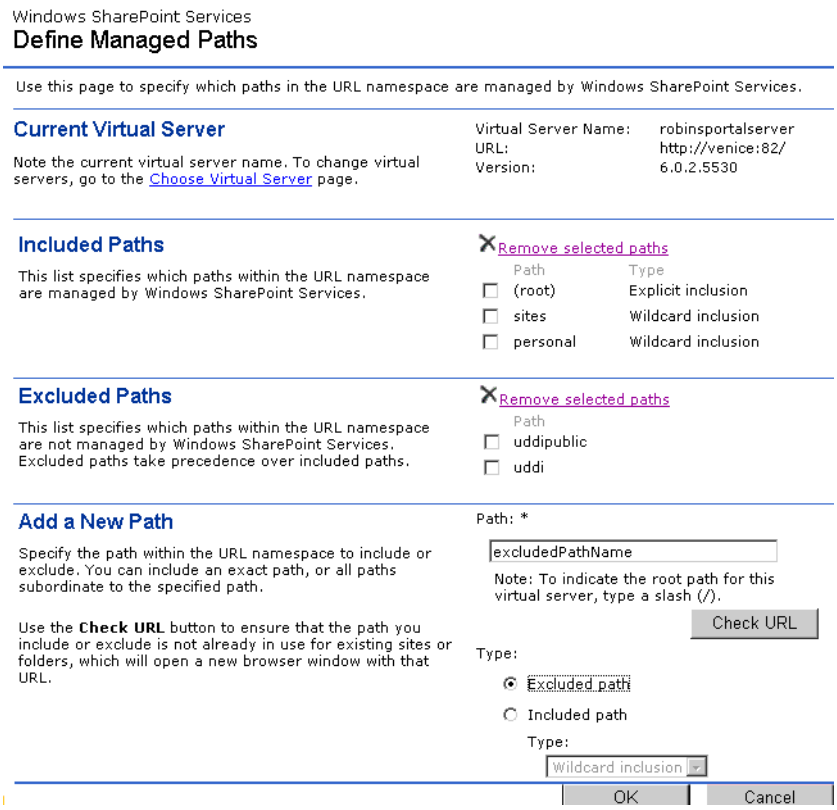
Defining Managed Paths in SharePoint

You complete the following procedure only if the Policy Manager or WebPass resides on the same IIS virtual server as SharePoint Portal Server *and* listens to the same port as that IIS virtual server. For instance, the default virtual IIS server uses port 80, as do many Policy Manager and WebPass installations; therefore, you need to change the port used by one application or exclude the path used by the Oracle Access Manager component through the Define Managed Paths feature in SharePoint.

To define managed paths in SharePoint

1. Select Start, Administrative Tools, SharePoint Central Administration.
2. In the Virtual Server Configuration section, click Configure virtual server settings.
3. In the Virtual Server list, click Default Web Site or the name of the virtual server on which both SharePoint Portal Server and the Oracle Access Manager components are installed.
4. In the Virtual Server Management section, select Define Managed Paths.
5. In the Add a Path section, type the path to Policy Manager or WebPass, then click the button marked Excluded path.
6. Click OK to add the path to the list of excluded paths.

Figure 17–1 Defining Managed Paths in SharePoint



Integrating with SharePoint Office Server 2007

Integrating with SharePoint Office Server 2007 is very similar to integrating with SharePoint Portal Server 2003. Many of the steps for performing the two integrations are identical.

The following paragraphs describe how to create and set up a Web site using SharePoint Office Server 2007.

Note: The SharePoint Portal Server 2003 portal creation object model is deprecated in SharePoint Office Server 2007. In SharePoint Office Server 2007, portal sites use the same provisioning process as Windows SharePoint Services sites. You must update any scripts that create portals sites to use the Microsoft Windows SharePoint Services 3.0 site creation APIs. If you require a new vServer (Web application), use the Windows SharePoint Services CreateWebApplication API before creating the site. See the following URL for details:

`http://msdn2.microsoft.com/en-us/library/ms545807.aspx`

Task overview: Integrating with SharePoint Portal Server

1. Install the Microsoft components, as described in "Installing Microsoft Components" on page 17-5.
2. Create a new Web application or site application in SharePoint Office Server 2007.

See "To create a new Web application in SharePoint Office Server 2007" on page 17-11 and "To create a new site collection for SharePoint Office Server 2007" on page 17-13 for details.

3. Install Oracle Access Manager, as described in "Installing Oracle Access Manager Components" on page 17-8.
4. Set up impersonation, as described in "Setting Up Impersonation" on page 17-13.
5. Complete the integration, as described in "Completing the SharePoint Server Integration" on page 17-21.

To create a new Web application in SharePoint Office Server 2007

1. From the Windows desktop, click Start, then All Programs, then Microsoft Office Server, then click SharePoint 3.0 Central Administration.
2. From the Central Administration home page, click Application Managment.
3. From the Application Managemetn page, in the SharePoint Web Application Management section, click Create or Extend Web Application.
4. From the Create or Extend Web Application page, in the Adding a SharePoint Web Application section, click Create a New Web Application.
5. Configure the following on the Create New Web Application page:

Table 17–3 Create Web Application Options for SharePoint Office Server 2007

Section	What You Configure in This Section
IIS Web Site	In this section you configure the following settings for your new Web application, as follows: ■ To choose an existing Web site, click Use an Existing Web Site, and from the drop-down menu select the Web site where you want to install the new Web application. ■ To create a new site, click Create a New IIS Web Site, and enter a name in the Description text field. ■ In the Port field, enter the port number you want to use to access the Web application. For a new Web site, this field contains a default port number. For an exiting site, this field contains the currently configured port number. ■ In the optional Host Header field, enter the URL for accessing the Web application. ■ In the Path field, enter the path to the directory that contains the site on the server. For a new Web site, this field contains a default path. For an exiting site, this field contains the current path.

Table 17–3 (Cont.) Create Web Application Options for SharePoint Office Server 2007

Section	What You Configure in This Section
Security Configuration	In this section you configure authentication and encryption for your Web application, as follows: ■ In the Authentication Provider section, select Negotiate (Kerberos) or NTLM, as appropriate. ■ In the Allow Anonymous section, choose Yes or No. A value of Yes allows anonymous access to the Web site by using a computer-specific anonymous access account. The account name is IUSR_*computername*. ■ In the Secure Sockets Layer (SSL) section, choose Yes or No. If you choose to enable SSL for the Web site, you must configure SSL by requesting and installing a certificate.
Load Balanced URL	Enter the URL for the domain name for all sites that users will access in this Web application. This URL domain will be used in all links shown on pages in the Web application. By default, the box is populated with the current server name and port. The Zone field is automatically set to Default for a new Web application and cannot be changed from this page.
Application Pool	In the Application Pool section, choose whether to use an existing application pool or create a new application pool for this Web application, as follows: ■ To use an existing application pool, select Use Existing Application Pool, then select the application pool you wish to use from the drop-down menu. ■ To create a new application pool, select Create a New Application Pool, and in the Application Pool Name field, type the name of the new application pool, or keep the default name. In the section Select a Security Account for This Application Pool, select Predefined to use an existing application pool security account, then select the security account from the drop-down menu. To use a security account that is not currently being used for an existing application pool, select Configurable, enter the user name of the account you want to use in the User Name field, and enter the password for the account in the Password field.
Reset Internet Information Services	In this section, choose whether to allow SharePoint Office Server 2007 to restart IIS on other farm servers. The local server must be restarted manually for the process to finish. If you do not select this option and you have more than one server in the farm, you must wait until the IIS Web site is created on all servers and then run `iisreset/noforce` on each Web server. The new IIS site is not usable until this action is completed. This choice is unavailable if your farm only contains a single server.
Database Name and Authentication	In this section, choose the database server, database name, and authentication method for your new Web application. In the Database Name field, enter the name of the database or use the default entry. In the Database Authentication field, choose whether to use Windows authentication (recommended) or SQL authentication, as follows: ■ If you want to use Windows authentication, leave this option selected. ■ If you want to use SQL authentication, select SQL authentication. In the Account field, type the name of the account that you want the Web application to use to authenticate to the SQL Server database, then type the password in the Password field.

6. Click OK to create the new Web application, or click Cancel to cancel the process and return to the Application Management page.

To create a new site collection for SharePoint Office Server 2007

1. On the SharePoint Central Administration home page, click the Application Management tab on the top link bar.
2. On the Application Management page, in the SharePoint Site Management section, click Create Site Collection.
3. On the Create Site Collection page, in the Web Application section, either select a Web application to host the site collection from the Web Application drop-down list, or create a new Web application to host the site collection, as follows:

Table 17–4 Create a Web Application to Host a Site Collection for SharePoint Office Server 2007

Section	What You Configure in This Section
Title and Description	Enter a title and description for the site collection
Web Site Address	Select a URL type, and specify a URL for the site collection.
Template	Select a template from the tabbed template control.
Primary Site Collection Administrator	Enter the user account name for the user you want to be the primary administrator for the site collection. You can also browse for the user account by clicking the book icon to the right of the text box. You can verify the user account by clicking the check names icon to the right of the text box.
Secondary Site Collection Administrator (optional)	Enter the user account for the user that you want to be the secondary administrator for the site collection. You can also browse for the user account by clicking the book icon to the right of the text box. You can verify the user account by clicking the Check Names icon to the right of the text box.

Setting Up Impersonation

Setting up impersonation, whether for SharePoint Server integration or for use by some other application, is described in the following sections:

Task overview: Setting up impersonation

1. Create a trusted user account for only impersonation in the Active Directory connected to SharePoint Server, as described in "Creating a Trusted User Accounts" on page 17-14.
2. Give the trusted user the special right to act as part of the operating system., as described in "Assigning Rights to the Trusted User" on page 17-14.
3. Bind the trusted user to the WebGate by supplying the authentication credentials for the trusted user, as described in "Binding the Trusted User to Your WebGate" on page 17-15.
4. Add a header variable named impersonate to Authorization Success Action in the policy domain for impersonation, as described in "Adding an Impersonation Action to a Policy Domain" on page 17-16.
5. Configure IIS by adding the IISImpersonationExtension.dll to your IIS configuration, as described in "Adding an Impersonation dll to IIS" on page 17-17.

6. Test impersonation, as described in "Testing Impersonation" on page 17-19.

Creating a Trusted User Accounts

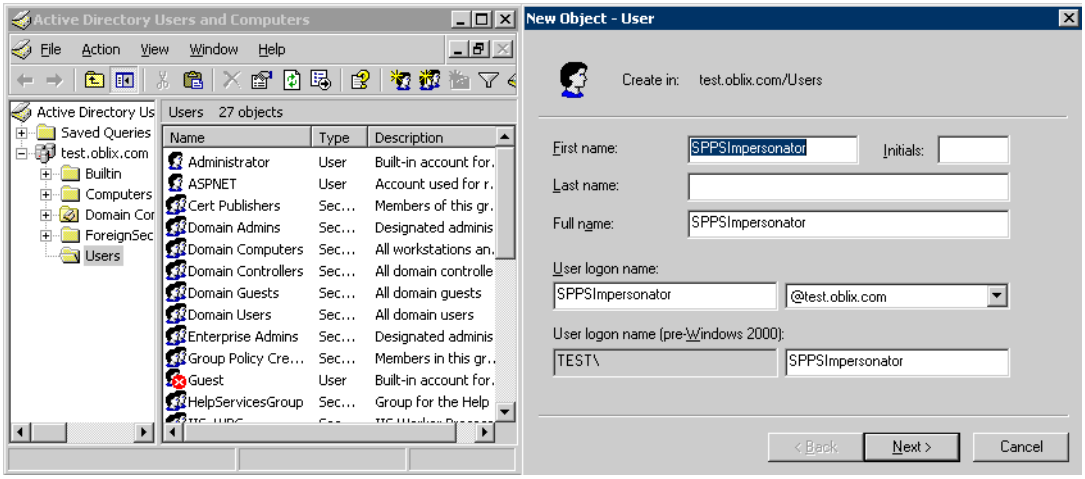
This special user should not be used for anything other than impersonation.

To create a trusted user account

1. On the Windows 2003 machine hosting your SharePoint Portal Server installation, select Start, Programs, Manage Your Server, Domain Controller (Active Directory), Manage Users and Computers in Active Directory.
2. In the Active Directory Users and Computers window, right-click Users on the tree in the left pane, then select New, User.
3. In the First name field of the pane entitled New Object - User, enter an easy-to-remember name such as *SPPSImpersonator*.
4. Copy this same string to the User logon name field, then click Next.
5. In succeeding panels, you will be asked to choose a password and then retype it to confirm.

Note: Oracle recommends that you chose a very complex password, because your trusted user is being given very powerful permissions. Also, be sure to check the box marked Password Never Expires. Since the impersonation extension should be the only entity that ever sees the trusted user account, it would be very difficult for an outside agency to discover that the password has expired.

Figure 17–2 Setting up a Trusted User Account for Windows Impersonation



Assigning Rights to the Trusted User

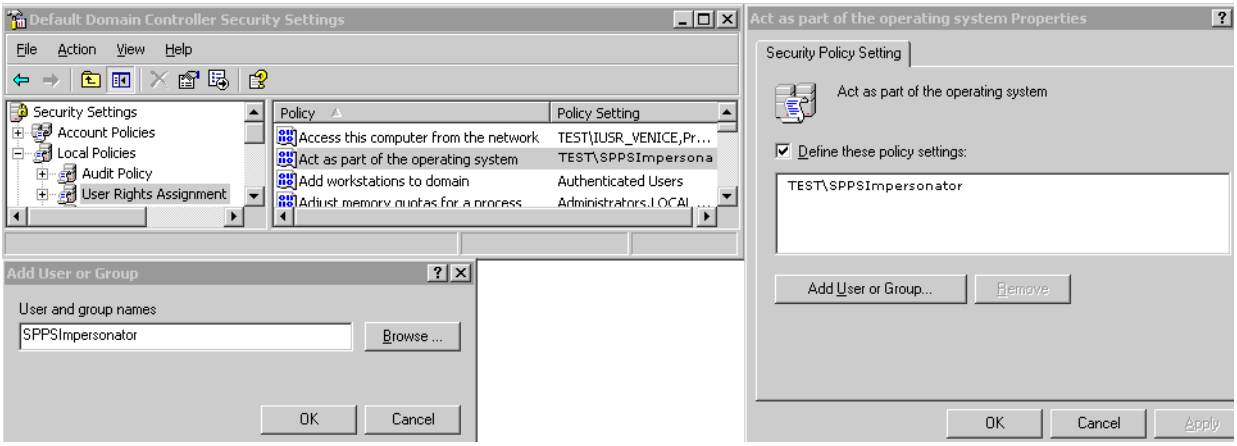
You need to give the trusted user the right to act as part of the operating system.

To give appropriate rights to the trusted user

1. Select Control Panel, Administrative Tools, Domain Controller Security Policy.
2. On the tree in the left pane, click the plus icon (+) next to Local Policies.

3. Click User Rights Assignment on the tree in the left pane.
4. Double-click "Act as part of the operating system" in the right pane.
5. Click Add User or Group.
6. In the Add User or Group panel, type the User logon name of the trusted user (SPPSImpersonator in our example) in the User and group names text entry box, then click OK to register the change.

Figure 17–3 Configuring Rights for the Trusted User in Windows Impersonation



Binding the Trusted User to Your WebGate

You need to bind the trusted user to the WebGate by supplying the authentication credentials for the trusted user, as follows.

To bind your trusted user to your WebGate

1. Point your browser to your Access System Console. For example:

   `http://hostname.domain.com:port/access/oblix`

   where *hostname* is the DNS name of the machine hosting your Policy Manager, *domain* is the name of the server domain to which the machine belongs, and port is the number of the *port* to which Policy Manager listens.

2. Navigate to Access System Console, Access System Configuration, AccessGate Configuration.
3. Select the name of the Webgate you want to modify.

   The Details for AccessGate page appears with a summary of the configuration information for this WebGate. At the bottom of this Web page are fields for Impersonation Username and Impersonation Password.

4. Click the Modify button at the bottom of the Details for AccessGate page.
5. After the Modify AccessGate page appears, scroll to the bottom and enter the username and password for the trusted user account you created through the task on page 17-14.

   For example:

6. Click the Save button to commit the changes and return to the Details page.

   A bind has been created for the WebGate and the trusted user. The WebGate is now ready to provide impersonation on demand. The demand is created by an Authorization Success Action in a policy domain created for impersonation.

Adding an Impersonation Action to a Policy Domain

You must create or configure a policy domain to protect your SharePoint resources. You do this by adding an Authorization Success Action with a return type of "headerVar," the "name" parameter set to "Impersonate", and the "return attribute" parameter set to "samaccountname" for a single-domain Active Directory installation or "userPrincipalName" for a multi-domain Active Directory forest.

You must also choose an easy-to-remember name for the domain, such as *Impersonation Policy Domain*.

For details on creating a policy domain, see the *Oracle Access Manager Access System Administration Guide.*

To add an impersonation action to your policy domain

1. Navigate to the Access System Console and log in, for example:

   `http://hostname.domain.com:port/access/oblix`

   Where *hostname* is the DNS name of the machine hosting your WebPass and Policy Manager, *domain* is the name of the server domain to which the machine belongs, and port is the number of the *port* to which Policy Manager listens.

2. Navigate to the Authorization Definitions page of the policy domain you want to change:

   Policy Manager, My Policy Domains, *PolicyName*, Authorization Rules

   Where *PolicyName* refers to the policy domain you created specifically for impersonation (*ImpersonationPolicyDomain* in our example).

   Currently defined authorization rules are listed. If none are listed, click the Add button and complete the form to create one.

3. Click the link to the rule to which you want to add the impersonation action to expand the description.

4. Click the Actions tab, directly under the Authorization Rules tab.

   The Authorization Success page appears, with a separate section for Authorization Success and Authorization Failure. If no actions are identified, you must add them. If actions are provided, you can modify them.

   You need to add a header variable named "impersonate" to the Authorization Success Action in the policy domain for impersonation.

5. On the Authorization Success page, click the Add or Modify button.

6. In the Authorization Success area, fill in the Return details.

   For example:

   `Type: HeaderVar`

```
Name: IMPERSONATE
Return attribute: uid or samaccountname (Active Directory username, the Windows
domain user for the desired folder)
```

Where "HeaderVar" is the return Type, "IMPERSONATE" is Name of the header variable for impersonation, and the Return value of *uid* or *samaccountname* is based on the directory being used.

7. Save the rule, which is used for the second WebGate request (for authorization).

   The following is a sample screen shot.

Adding an Impersonation dll to IIS

You are ready to configure IIS by adding the IISImpersonationExtension.dll to your IIS configuration. See "To add the impersonation dll to your IIS configuration" on page 17-17 for details.

If you have multiple Web sites, where some are integrated with SharePoint Portal Server while others are not, you may want to enable impersonation for the Web sites that are not integrated with the SharePoint Portal Server. To do this, you must install Impersonation.dll at any level of the Web site tree and add a wildcard extension for Web sites. See "To add a wildcard extension for Web sites" on page 17-18 for details.

To add the impersonation dll to your IIS configuration

1. Select Start, Administrative Tools, Internet Information Services (IIS) Manager.
2. Click the plus icon (+) to the left of the local computer icon on the tree in the left pane.
3. Click Web Service Extensions on the tree in the left pane.
4. Double-click Oblix WebGate in the right panel to open the Properties panel.
5. Click the Required Files tab.
6. Click Add.
7. In the Path to file text box, type the full path to IISImpersonationExtension.dll.

   By default, the path is:

   ```
   WebGate_install_dir\access\oblix\apps\webgate\bin\
   IISImpersonationExtension.dll
   ```

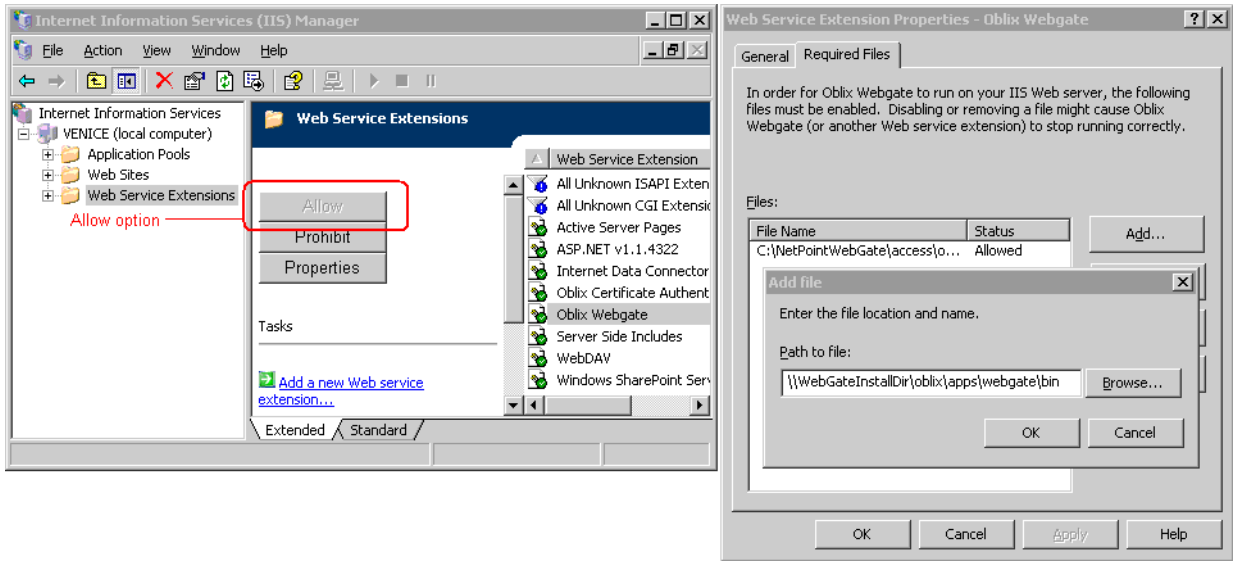
   Where *WebGate_install_dir* is the directory of your WebGate installation.

Note: If any spaces exist in the path (for example, C:\Program Files\Oracle\...) surround the entire string with double quotes (" ").

8. Click OK.
9. Click the General tab on the Web Services Extension Properties panel, then verify that the box "Do not check the file location" is not checked.
10. Verify that the Allow button to the left of the Oblix WebGate icon is greyed out, which indicates that the dll is allowed to run as a Web service extension.

Note: If Allow is not greyed out, click it so that it becomes greyed out. When Allow is greyed out, this indicates that the highlighted file is permitted to run on the IIS virtual server.

Figure 17–4 Configuring IIS Security Settings



To add a wildcard extension for Web sites

1. If Oracle Access Manager is not integrated with SharePoint Server, configure a wildcard extension for Web sites by navigating as follows:

   Click Start, then click Administrative Tools, then click Internet Information Services (IIS) Manager.
2. Click the plus icon (+) to the left of the local computer icon on the tree in the left pane.
3. Click Web Sites in the tree in the left pane.
4. Right-click the icon representing your Web site, then click Properties in the menu.
5. Click the Home Directory tab.
6. Click the Configuration button.
7. In the list box for Wildcard application maps, click the entry for IISImpersonationExtension.dll to highlight it, then click Edit.

8. Ensure that the box is unchecked.

Testing Impersonation

You can test Impersonation in the following ways:

- Outside the SharePoint Server context or test single sign-on, as described in "Creating an IIS Virtual Site Not Protected by SharePoint Server" on page 17-19
- Using the Event Viewer, as described in "Testing Impersonation Using the Event Viewer" on page 17-19
- Using a Web page, as described in "Testing Impersonation using a Web Page" on page 17-20
- Using negative testing as described in "Negative Testing for Impersonation" on page 17-21

Creating an IIS Virtual Site Not Protected by SharePoint Server

To test the impersonation feature outside the SharePoint Server context or to test single sign-on, you will need a target Web page on an IIS virtual Web site that is not protected by SharePoint Server. You create such a virtual Web site by completing the following task.

To create an IIS virtual site not protected by SharePoint Server

1. Select Start, Administrative Tools, Internet Information Services (IIS) Manager.
2. Click the plus icon (+) to the left of the local computer icon on the tree in the left pane.
3. Right-click Web Sites on the tree in the left pane, then navigate to New, Web Site on the menu.
4. Respond to the prompts by the Web site creation wizard.
5. After you create the virtual site, you must protect it with a policy domain, as described in the *Oracle Access Manager Access System Administration Guide.*

Testing Impersonation Using the Event Viewer

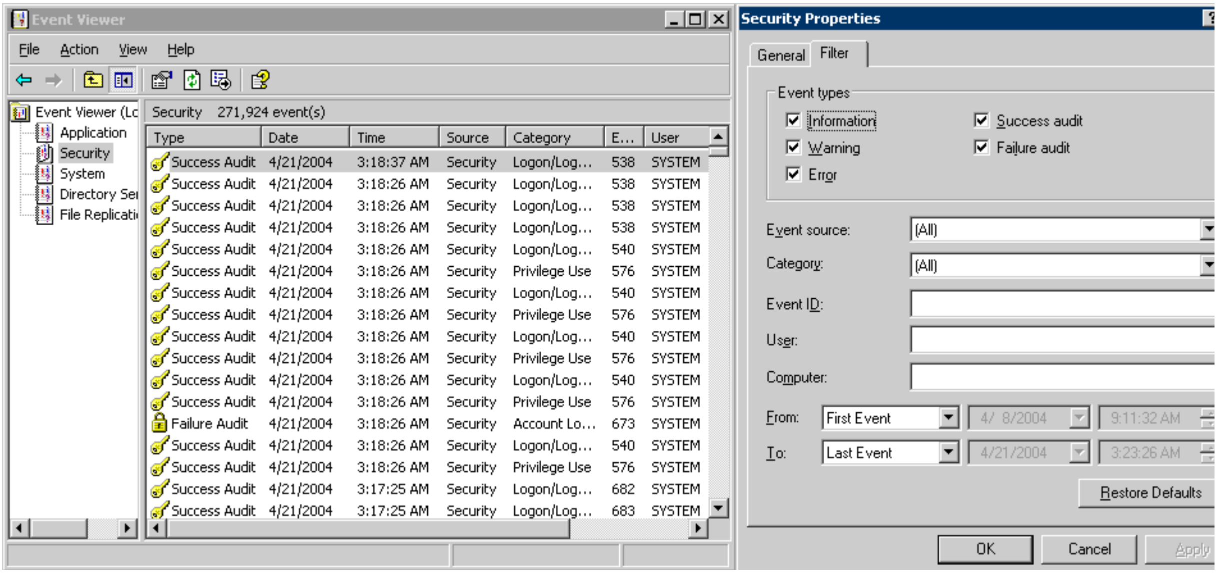
When you complete impersonation testing using the Windows 2003 Event Viewer, you must configure the event viewer before conducting the actual test.

To test impersonation through the Event Viewer

1. Select Start Menu, Event Viewer.
2. In the left pane, right-click Security, then click Properties.
3. Click the Filter tab on the Security property sheet.
4. Verify that all Event Types are checked, and the Event Source and Category lists are set to All, then click OK to dismiss the property sheet.

   Your Event Viewer is now configured to display information about the HeaderVar associated with a resource request.

Figure 17–5 Verifying Event Viewer Settings



5. Create a new IIS virtual server (virtual site).
6. Place a target Web page anywhere in the tree on the virtual site.
7. Point your browser at the Web page.

If impersonation is working correctly, the Event Viewer will report the success of the access attempt.

Testing Impersonation using a Web Page

You can also test impersonation using a dynamic test page, such as an .asp page or a perl script, that can return and display information about the request.

To test impersonation through a Web page that displays server variables

1. Create an .asp page or perl script that will display the parameters AUTH_USER and IMPERSONATE, which can resemble the sample page presented in the following listing:

Example 17–1 Sample .ASP Page Code

```
<TABLE border=1>
       <TR>
             <TD>Variable</TD>
             <TD>  </TD>
             <TD>Value</TD></TR>
       <%for each servervar in request.servervariables%>
       <TR>
             <TD><%=servervar%></TD>
             <TD>  </TD>
             <TD><%=request.servervariables(servervar)%> </TD>
       </TR>
```

2. Create an IIS virtual site, or use the one you created for the previous task.
3. Place an .asp page or perl script (such as the sample in the preceding listing) anywhere in the tree of the new virtual site.

4. Point your browser at the page, which should appear, with both AUTH_USER and IMPERSONATE set to the name of the user making the request.

Negative Testing for Impersonation

To conduct negative testing for impersonation, you need to unbind the trusted user from the WebGate, as explained in the following procedure.

To unbind the trusted user from your WebGate

1. Log in to the Access System Console at a URL similar to the following:

   `http://hostname.domain.com:port/access/oblix`

   Where *hostname* is the DNS name of the computer hosting your Access Manager, *domain* is the name of the server domain to which the machine belongs, and *port* is the number of the port to which Access Manager listens.

2. Select Access System Console, then click Access System Configuration, then click AccessGate Configuration.

3. Select the name of the Webgate that you want to modify.

   The Details for AccessGate page appears with a summary of the configuration information for this WebGate. At the bottom of this Web page are fields for Impersonation Username and Impersonation Password.

4. Click the Modify button at the bottom of the Details for AccessGate page.

   The Modify AccessGate page appears.

5. Remove the credentials for the trusted user.

6. Click Save.

   You return to the Details page.

7. Restart the IIS server.

8. Point your browser at the sample .ASP code page that you created in "To test impersonation through a Web page that displays server variables" on page 17-20.

   An error message page should appear. Values for AUTH_USER and IMPERSONATE are necessary for impersonation credentials to be bound to a WebGate.

Completing the SharePoint Server Integration

You need to complete several procedures to set up a Oracle Access Manager/SharePoint Server integration.

Task overview: Setting up the SharePoint Server integration

1. Set up IIS security, as described in "Configuring IIS Security" on page 17-22.

2. Configure the wildcard extension for each SharePoint Server virtual server for which you wish to enable integration, as described in "Configuring the Wildcard Extension" on page 17-23.

3. Edit the web.config file, as described in "Editing web.config" on page 17-24.

4. Test the integration, as described in "Testing Your Integration" on page 17-26.

Configuring IIS Security

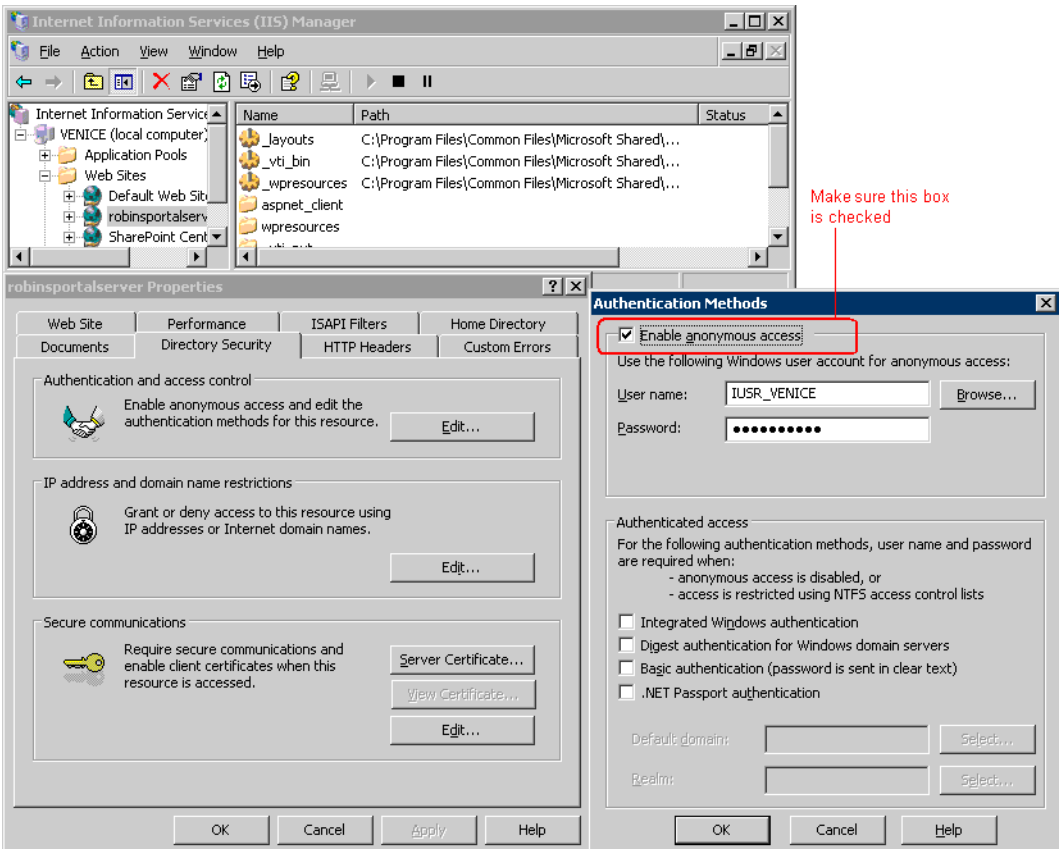
Be sure to configure IIS Security before you continue.

To configure IIS Security for the SharePoint Server integration

1. Select Start, Administrative Tools, Internet Information Services (IIS) Manager.
2. Click the plus icon (+) to the left of the local computer icon on the tree in the left pane.
3. Click Web Sites on the tree in the left pane.
4. Right-click the icon on the tree in the left pane that represents the SharePoint Server server you are protecting with your WebGate, then select Properties from the menu.
5. In the property sheet for the SharePoint Server server, click the Directory Security tab.
6. In the Authentication and access control section of the Directory Security tab, click Edit.
7. In the Authentication Methods panel, click the box labelled Enable anonymous access so that a check appears, then click OK to complete the task.

Note: Enable anonymous access does not enable anonymous users to access the SharePoint Server. Rather, this setting configures IIS to relinquish access control to the Access System.

Figure 17–6 Configuring IIS Security



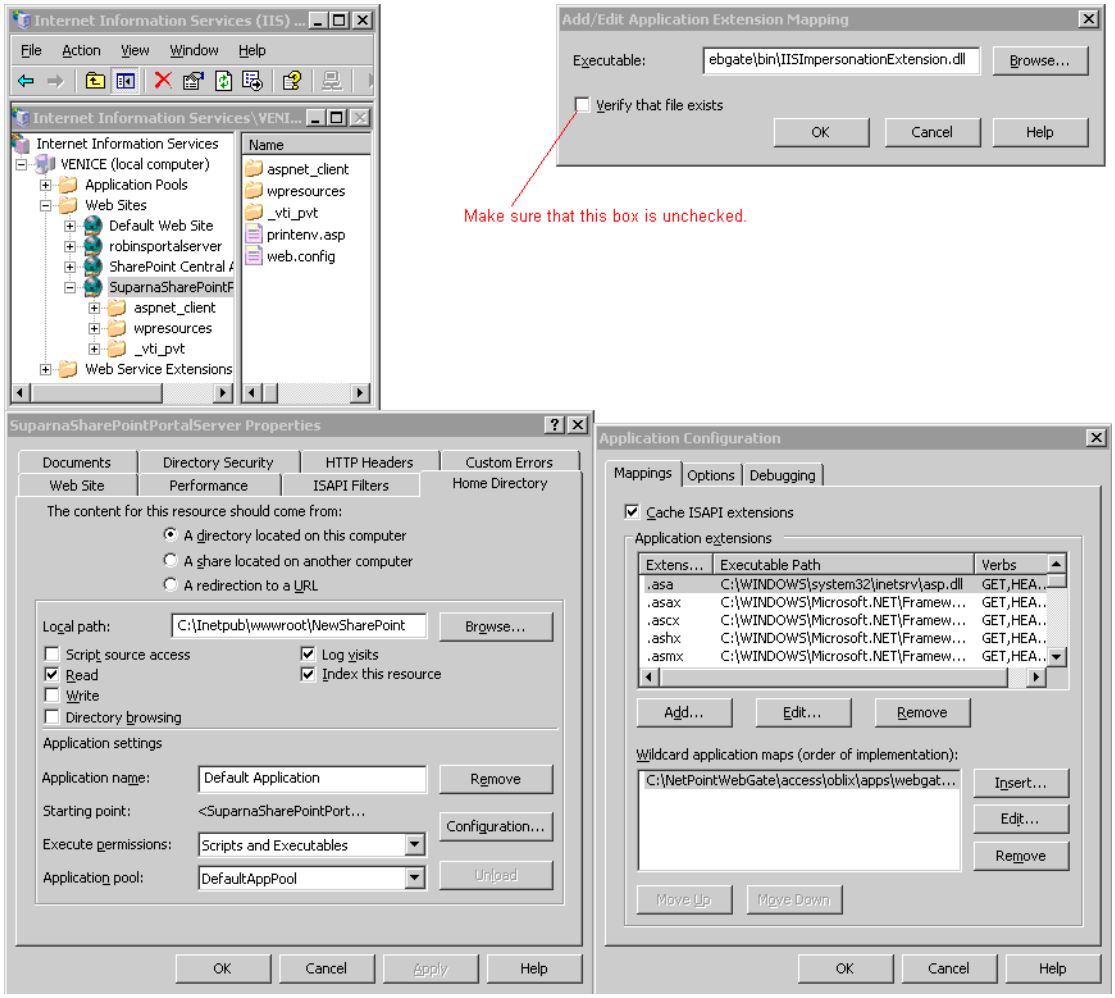
Configuring the Wildcard Extension

You are ready to configure the wildcard extension for each SharePoint Portal Server virtual server for which you wish to enable integration.

To configure the wildcard extension for SharePoint Portal Server virtual servers

1. Select Start, Administrative Tools, Internet Information Services (IIS) Manager.
2. Click the plus icon (+) to the left of the local computer icon on the tree in the left pane.
3. Click Web Sites on the tree in the left pane.
4. Right-click the icon representing your SharePoint Portal Server server, then click Properties on the menu.
5. Click the Home Directory tab.
6. Click the Configuration button.
7. In the list box for Wildcard application maps, click the entry for IISImpersonationExtension.dll to highlight it, then click Edit.
8. Ensure that the box is unchecked.
9. Verify that the file exists, then click OK three times to close the Add/Edit panel, the Application Configuration panel and the property sheet for your portal server.

Figure 17–7 Configuring the Wildcard Extension



Editing web.config

Add the following line to the web.config file.

```
<add key = "SPS-EnforceIISAnonymousSetting" value="false" />
```

To edit web.config for the SharePoint Portal Server integration

1. Open Windows Explorer and navigate to the document root of your IIS Web site.
2. Use any text editor to open the XML file web.config.
3. Locate the appSettings markers at the end of the file, or create them if they do not exist:

```
<Configuration>
// [Various configuration settings]
<appSettings>
// [Insert "<add key . . .>" here.]
</appSettings >
</Configuration>
```

Important: The appSettings markers are case sensitive and must appear as appSettings.

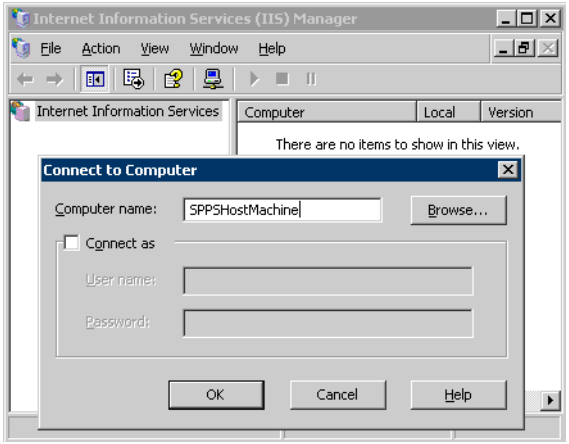
4. Add the following line where indicated in the previous listing:

```
<add key = "SPS-EnforceIISAnonymousSetting" value="false" />
```

5. Save web.config.
6. Restart IIS so that the new setting will take effect by completing the following steps:
   a. Select Start Menu, Internet Information Services (IIS) Manager.
   b. In the tree in the left pane, locate the name of the local computer hosting your SharePoint Portal Server installation and write it down; you will need the name of this computer to restart IIS.
   c. Right-click the local computer icon and select Disconnect from the menu.
   d. After the warning asks if you really want to disconnect, click Yes to confirm the action.

      The local computer icon disappears from the tree in the left pane, indicating that IIS has been shut down on that machine.
   e. In the tree in the left pane, right-click the Internet Information Services icon and click Connect on the menu.
   f. In the Connect to computer panel, type the name of the computer hosting your SharePoint Portal Server installation, then click OK to restart IIS.

Figure 17–8 Restarting IIS after Editing web.config



Synchronizing User Profiles Between Directories

You need to synchronize user profiles between the SharePoint Portal Server directory and the Oracle Access Manager directory:

- **Uploading user data**—If your Oracle Access Manager installation is configured for any directory server other than SharePoint Active Directory, you must load the user profiles that reside on the other directory server to SharePoint Active Directory.

- **Importing user profiles in SharePoint Portal Server**—After uploading the user profiles, you need to import the profiles from Active Directory to SharePoint Portal Server.

To configure importing user profiles in SharePoint Portal Server

1. Go to Site Settings.
2. Under User Profile, Audiences, and Personal Sites, click Manage Profile Database.
3. To configure user profile imports from Active Directory, click Configure Profile Import.

Testing Your Integration

After you complete the tasks to enable integration, you should test to verify that integration is working.

This section contains the following topics:

Testing the SharePoint Portal Server Integration

You want to verify that a user can access SharePoint Portal Server resources through Oracle Access Manager authentication and SharePoint Portal Server authorization.

To test your SharePoint Portal Server integration

1. Navigate to any SharePoint Portal Server Web page using your browser.

   The Access System challenges you for credentials.
2. Log in by supplying the necessary credentials, then verify that the page you requested is visible.
3. **Optional**: Check the Event Viewer to confirm that the access request was successful.

Testing Single Sign-On for the SharePoint Portal Server Integration

You should also test single sign-on by demonstrating that a user who has just supplied credentials and accessed an SharePoint Portal Server resource can (before the ObSSOCookie expires) access a non-SharePoint Portal Server resource without having to supply credentials a second time. For example, use a resource defined in the Policy Manager.

When single sign-on is working, you should be granted access to the page without having to supply credentials a second time.

To test single sign-on for your SharePoint Portal Server integration

1. Create and protect a new virtual site with a policy domain (or use one you have already created.
2. Place a Web page anywhere in the tree of this virtual site.
3. Using a browser, navigate to the page in the new virtual site.

   If you have already passed authentication, you should be granted access to the page without having to supply credentials a second time.

18

Integrating With ASP.NET

Oracle Access Manager supports the ASP.NET component of the Microsoft .NET Framework, which developers can use to build, deploy, and run Web applications and distributed applications.

The Security Connector for ASP.NET supports and enhances native .NET role-based security. This chapter explains how to use the Security Connector for ASP.NET to instantiate a new OblixPrincipal object and populate it with roles (Oracle Access Manager authorization rules) and the native WindowsPrincipal object.

This chapter includes the following topics:

About ASP.NET

ASP.NET is a set of technologies in the Microsoft .NET Framework that enables the building of Web applications and XML Web services using compilation and caching technologies available in the .NET Framework. Characteristics of ASP.NET pages:

- Run on a server and generate markup that is sent to a browser, for example, HTML, WML, or XML
- Use a compiled, event-driven programming model that enables the separation of application logic and the user interface
- Contain server-side logic, rather than client-side logic, written in Visual Basic .NET, C# .NET, or any .NET-compatible language

Developers can use the .NET Framework class library, which is an object-oriented collection of reusable types, to create ASP.NET applications. Web applications and XML Web services benefit from features of the common language runtime (CLR).

Security Principals and Security Identifiers (SIDs)

Both ASP.NET and Microsoft Internet Information Services (IIS) provide security models that allow you to authenticate users appropriately and obtain the correct security context within your application.

The user's (or potentially an application's or computer's) identity is referred to as a security principal. The client must provide credentials to allow the server to verify the identity of the principal. After the identity is known, the application can authorize the principal to access protected resources.

WIndows provides a WindowsPrincipal object that defines a user identity and the user's role identity. The role identity is the role or roles defined in Windows for the user identity. Microsoft .NET technology provides an interface to create a Principal object using only Windows-specific roles. ASP.NET applications can call the Windows IPrincipal.IsInRole method to find out if the identity is in a specific role, for example, the admin role or users role.

Security within the ASP.NET Framework revolves around security identifiers (SIDs). SIDs are equal to Oracle Access Manager single sign-on tokens and represent a unique user within the Windows operating system. ASP.NET wraps each SID into a series of managed objects that allow a developer to impersonate that user.

The main object that wraps the SID is the Identity object (IIdentity). This object enables a developer to discover how that identity was established by calling methods to obtain the following:

- The authentication method
- The name of the identity
- The authentication status (authenticated or not)

For more information, consult the Microsoft ASP.NET documentation.

IPrincipal.IsInRole Method Syntax

A principal object represents the security context of the user on whose behalf the code is running, including that user's identity (IIdentity) and any roles to which the user belongs. The .NET Framework class library IPrincipal interface defines the basic functionality of a principal object.

Note: All principal objects are required to implement the IPrincipal interface.

During the authorization process, the public IPrincipal.IsInRole method determines whether the current principal belongs to the specified role.

The following IPrincipal.IsInRole method syntax is based on .NET Framework version 1.1 and is intended only as an example:

```
[Visual Basic]
Function IsInRole( _
    ByVal role As String _
) As Boolean
[C#]
bool IsInRole(
     string role
);
[C++]
```

```
bool IsInRole(
     string* role
);
[JScript]
function IsInRole(
     role : String
) : Boolean;
```

Parameters

`role`

The name of the role for which to check membership.

Return value

`true`—Returns true if the current principal is a member of the specified role.

`false`—Returns false otherwise.

Supported Versions and Platforms

You can find support and certification information at the following URL:

http://www.oracle.com/technology/documentation/

You must register with OTN to view this information.

Also, you can see the supported versions and platforms for this integration on Metalink, as follows.

To view information on Metalink

1. In your browser, enter the following URL:

   https://metalink.oracle.com

2. Log in to MetaLink.
3. Click the **Certify** tab.
4. Click **View Certifications by Product**.
5. Select the **Application Server** option and click **Submit**.
6. Choose **Oracle Identity Management** and click **Submit**.
7. Click **Oracle Identity Management Certification Information 10g (10.1.4.0.1) (html)** to display the Oracle Identity Management page.
8. Click the link for **Section 6, Oracle Access Manager Certification** to display the certification matrix.

Requirements

This operates on all Windows platforms that support the .NET framework.

Older WebGates are compatible with more current Access Servers. However, older WebGates use a different encryption scheme for the shared secret, as discussed in the *Oracle Access Manager Access System Administration Guide*.

For more information and the most current syntax, see the following Web site:

http://msdn.microsoft.com/developercenters/

About the Security Connector for ASP.NET

The Security Connector for ASP.NET:

- Provides a dynamically-loaded native library assembly to enhance ASP.NET behavior to take advantage of Oracle Access Manager features
- Extends pre-defined Microsoft roles to include dynamic groups and any attribute values defined in an Oracle Access Manager user's profile

For example, Microsoft provides pre-defined roles within a Windows domain. However, Microsoft roles do not include the flexibility of Oracle Access Manager's dynamic groups, timing, and other conditions that can alter a user's access rights.

You can customize your ASP.NET application or Web service to use the Oracle Access Manager assembly during the authorization process. This converts Oracle Access Manager authorization actions into roles using a header variable that maps to roles that are meaningful to the .NET environment.

Note: Administrators must plan and coordinate the roles that will be used with the application developer or deployer. Oracle Access Manager does not know what the .NET roles are, and has no way to discover what the roles are. The Oracle Access Manager role and the .NET role are related only through an Oracle Access Manager role string. There is no referential integrity supplied or implied.

Oracle Access Manager Components and Requirements

The Security Connector for ASP.NET library assembly, OBPrincipalHTTPModule.dll, is installed with the WebGate in the same directory as the webgate.dll. For example:

WebGate_install_dir\access\oblix\apps\webgate\bin\ObPrincipalHTTPModule.dll

Because more than one application may share the OBPrincipalHTTPModule assembly, it is included in the global assembly cache (GAC).

The new Oracle Access Manager library assembly runs as an ASP.NET HttpModule. Therefore, you must include details about this assembly in the Web.config file on the same machine and in the same directory as the ASP.NET application.

The ObPrincipalHTTPModule assembly includes a new OblixPrincipal object class and the OblixHttpModule. With this assembly, the ASP.NET application can define and pass an Oracle Access Manager role. Without this assembly, the application can pass only Windows roles.

The OblixHttpModule

The OblixHttpModule recognizes Oracle Access Manager roles, and the roles supported with the WindowsPrincipal object. In fact, Oracle Access Manager recognizes any principal object in the .NET framework class library.

The OblixHttpModule must be specified as an action type in the Oracle Access Manager authorization rule that protects the ASP.NET application. See "Setting up the Oracle Access Manager Role Action" on page 18-8 for details.

During authorization, the OblixHTTPModule:

- Instantiates the new OblixPrincipal object before control is given to the ASP.NET application using the WindowsPrincipal object (or any other type of principal object in the .NET framework class library) that was extracted from the request.

- Adds Oracle Access Manager role data (the authorization rule action) to the OblixPrincipal object.
- Associates the OblixPrincipal object with the ASP.NET HTTP request.

> **Note:** Whenever the application requests the principal object, it receives the OblixPrincipal object, which encapsulates all other principal objects.

The OblixPrincipal Object

A role is the name of a membership category, for example, admin or user. The OblixPrincipal object represents the security context of the user on whose behalf the code is running. This includes the user's identity (IIdentity) and any roles to which they belong as derived from the Windows IPrincipal interface. The iPrincipal.IsInRole method checks both Oracle Access Manager roles and the iPrincipal interface.

Authorization with the Security Connector for ASP.NET

The following high-level overview introduces authorization using the Security Connector for ASP.NET. For a more detailed example, see "Oracle Access Manager Role-Based Authorization" on page 18-9.

Process overview: Authorization with the Security Connector for ASP.NET

1. After the user is authenticated, the WebGate begins the authorization process with the Access Server, as usual.
2. The IIS Web server creates the WindowsPrincipal object based on Windows Impersonation.
3. The OblixHttpModule receives the request, instantiates a new OblixPrincipal object using the WindowsPrincipal object, and adds Oracle Access Manager role data based on the Oracle Access Manager authorization rule action.
4. The OblixPrincipal object initializes itself, then recognizes and stores Oracle Access Manager role data in memory.
5. The OblixHttpModule associates the OblixPrincipal object with the request and returns control to the IIS Web server.
6. The ASP application extracts the OblixPrincipal object for the request and calls the IPrincipal.IsInRole method.
7. The OblixPrincipal object calls the WindowsPrincipal object's IsInRole method from the .NET framework class library to determine whether the current principal belongs to the specified .NET role, checks the Oracle Access Manager role, and returns the answer to the ASP.NET application.
8. The IPrincipal.IsInRole method returns true or false, depending on the current user's identity and the Oracle Access Manager authorization rule. For a syntax example, see "IPrincipal.IsInRole Method Syntax" on page 18-2.

   If the answer is false, the Principal object looks in the Oracle Access Manager role list for the requested role and returns the answer to the ASP.NET application. If the answer is true, the ASP.NET application completes processing and access to the resource is granted.

Using the Security Connector for ASP.NET

The following task overview explains how to use the Security Connector for ASP.NET.

Task overview: Setting up the Security Connector for ASP.NET

1. Install a WebGate, as described in "Setting Up Your Environment" on page 18-6.
2. Set up the application, as described in "Setting Up the ASP.NET Application for the Security Connector" on page 18-7.
3. Configure the Oracle Access Manager role action, as described in "Setting up the Oracle Access Manager Role Action" on page 18-8.

For a process overview, see "Oracle Access Manager Role-Based Authorization" on page 18-9.

Setting Up Your Environment

Before you can use the Security Connector for ASP.NET, you must set up the WebGate on a machine hosting the IIS Web server and the .NET framework with ASP.NET.

You are given the option to include the .NET framework and ASP.NET during IIS Web server installation. This automatically configures the IIS metabase. The ASP.NET application security configuration and the IIS security configuration are independent. Each may be use separately or together.

IIS maintains security related configuration settings in the IIS metabase. ASP.NET maintains security (and other) configuration settings in XML configuration files. For more information, see your Microsoft documentation.

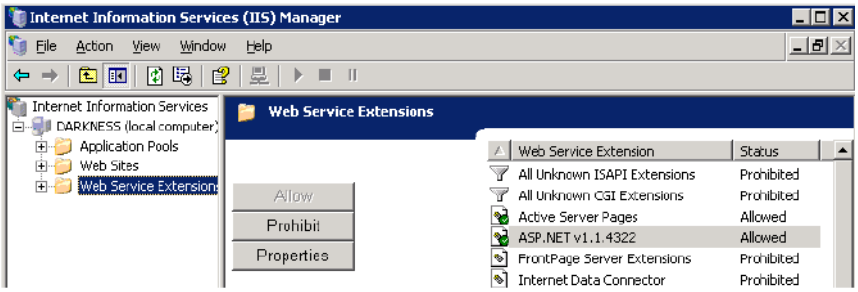
To set up your environment

1. Install the IIS Web server and the .NET Framework with ASP.NET.

   If you are using IIS v6.x, be sure to enable ASP.NET applications. The actions in the next step occur automatically if you install the .NET framework after installing the IIS Web server. If this reflects your installation, skip the next step.

2. Register ASP.NET and allow the ASP.NET Web Service Extension on the machine that will host the WebGate, if needed.

   For example, if you configure aspnet.regiis.exe, you would go to IIS, then *local_host*, then to Web Service Extensions, then to ASP.NET v1.1.4322, then to Properties, then to Allowed, as illustrated in the following screen shot:



3. Install the WebGate on a machine that hosts the IIS Web server, the .NET Framework, and ASP.NET.

The WebGate installation will end before completion if the .NET framework is not included on the WebGate host. To share the ObPrincipalHTTPModule assembly among all applications, the assembly is installed as part of the global assembly cache.

Setting Up the ASP.NET Application for the Security Connector

When you create the ASP.NET application or Web service using Visual Studio.NET, a generic Web.config file is created automatically. You can modify this file to customize your application to use the ObPrincipalHTTPModule.dll assembly during authorization.

With impersonation enabled, ASP.NET applications can run with the identity of the client on whose behalf they are operating. ASP.NET will receive the token to impersonate from IIS.

Note: If you do not enable impersonation in the application and in Oracle Access Manager, Oracle Access Manager roles will not be returned, which encapsulate all roles. However, the permissions may not be all you need.

To set up the ASP.NET application

1. Use Visual Studio .NET to write your ASP.NET application or Web service, as described in the Microsoft documentation.
2. Include the following details in the Web.config file under <System.Web> to use the OblixHttpModule and OBPrincipalHTTPModule.dll.

   Be sure to include your own PublicKeyToken. For example:

```
-->
<httpModules>
     <add type="Oblix.Agents.OblixHttpModule,
ObPrincipalHttpModule, Version=7.0.0.0, Culture=neutral,
PublicKeyToken="xxxxxxxxnnnnxxxx" name="OblixHttpModule"/>
</httpModules>
<compilation
     defaultLanguage="c#"
     debug="true"
/>
```

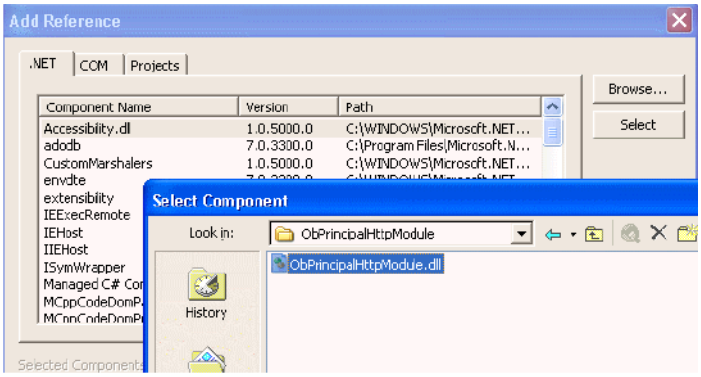
   Note: The value of Culture= is case-sensitive; "neutral" must be lowercase.

   ---

3. Reference the ObPrincipalHttpModule assembly in your application.

   Right-click the project in Visual Studio, select Add Reference, click the Project tab, then browse for and select the global assembly cache.

   Note: You may also use the /r option if the application is built from the command line.



You can either reference the iPrincipal object in the application, as described in the following steps, or point to the ACLs in the Web.config file.

4. Add the iPrincipal references for the required .NET assemblies to the application (or see the sample web.config file).

   For example:

   ```
   using System.Security.Principal;
   using System.Web.Security;
   ```

5. Add a method to the application that calls the IPrincipal.IsInRole() function with the appropriate parameters for your application.

   For example, if the return value of the authorization rule action is Manager, the method would be the following.

   ```
   Context.User.IsInRole("Manager"); // Context - HTTPContext
   object associated with the page
          // user - Principal object
   ```

   The application is now set up to use the Security Connector. Next you must set up the Oracle Access Manager role action.

Setting up the Oracle Access Manager Role Action

Actions can pass information about users to other applications in the same, or different, Oracle Access Manager policy domain. Authorization actions occur when a user requests access to a resource (that is, when the user requests the resource's URL).

Before passing roles to the OblixHttpModule, you need to create a role action in the authorization rule for the policy domain that protects the ASP.NET application. This action relies on the OblixHttpModule. Aware Oracle Access Manager clients can use the role action to define roles separate from header variables.

The role is contained in a Principal object. You may have as many roles as you choose. Each role value will be added to the OblixPrincipal object. When calling the IPrincipal.IsInRole method from the .NET Framework class library with any of the Role values, true is returned.

To set up Oracle Access Manager Role Actions

1. Create a policy domain in the Policy Manager to protect the ASP.NET applications and include an impersonation action.

For more information on policy domain configuration and single sign-on configuration, see the *Oracle Access Manager Access System Administration Guide*.

2. Select the authorization rule and click the Actions tab to define an action for this authorization rule to pass roles to the OblixHttpModule:
   - **Type**: Specify OblixHttpModule. Only a type of OblixHttpModule will be forwarded to the ASP.NET HttpModule.
   - **Name**: Supply a role or a name.

     You can have as many roles as you choose. Each role value is added to the OblixPrincipal object. When calling the IPrincipal.IsInRole method with any of these values, true is returned.

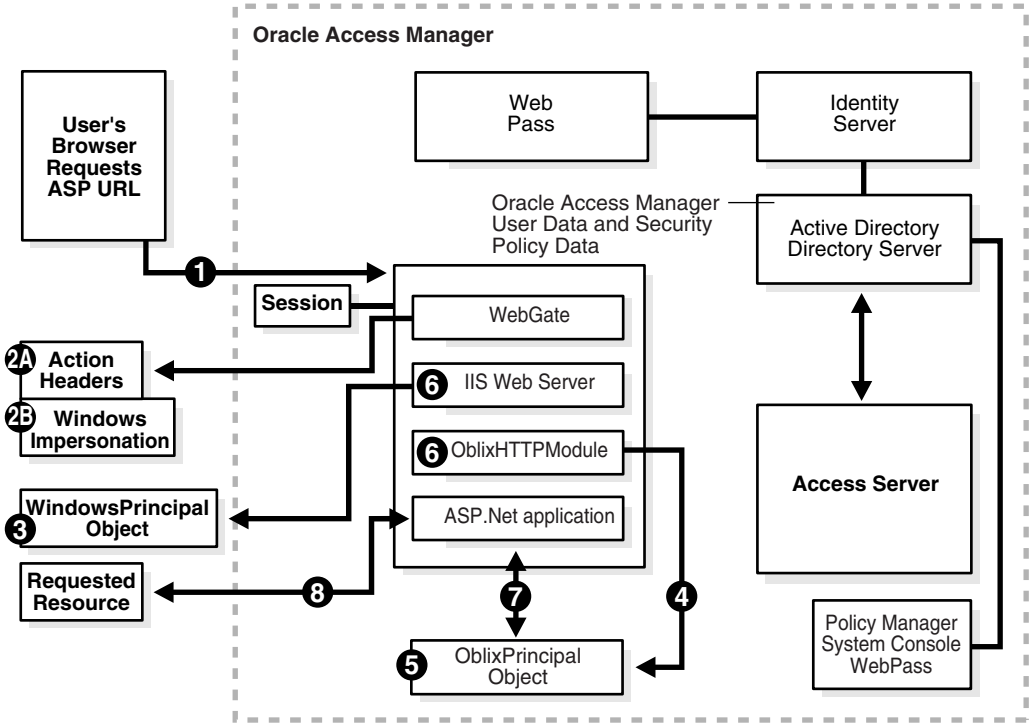
     Only one name may be set. Name exists in the event that IIS does not provide an Identity name, for example, None authentication in the Web.config file. If Windows authentication is set and the WebGate is configured for impersonation, this action is ignored.
   - **Return Value**: This can be any static or dynamic value, like any other action.
3. Save the rule and restart the Access Server to have your changes take effect.

   Your environment is set up, the ASP.Net application is complete, and the Oracle Access Manager policy domain protects the application with the new authorization rule.

Oracle Access Manager Role-Based Authorization

The following process occurs during authentication and role-based authorization with the Security Connector for ASP.NET. Figure 18–1 illustrates the sequence and is followed by a detailed description.

Figure 18–1 Security Connector for ASP.NET Authorization Flow



Process overview: Events during authentication and authorization

1. The Web server receives the user's ASP URL request. The WebGate intercepts the request and communicates with the Access Server to determine:
   - If the resource is protected
   - How the resource is protected
   - If the user is authenticated
   - If user access is authorized

   Authentication is performed between the Access Server and directory server, as usual.
2. When the user is authenticated, the WebGate begins the authorization process with the Access Server and:
   a. Sets action headers for roles
   b. Performs Windows Impersonation
   c. Returns control to the IIS Web server
3. The IIS Web server creates the WindowsPrincipal object based on Windows Impersonation.
4. The OblixHttpModule:
   a. Receives the request
   b. Instantiates a new OblixPrincipal object using the WindowsPrincipal object that was extracted from the request
   c. Adds Oracle Access Manager role data (the authorization rule action)

5. The OblixPrincipal object initializes itself and stores Oracle Access Manager role data in memory.
6. The OblixHttpModule associates the OblixPrincipal object with the request and returns control to the IIS Web server.
7. The ASP application extracts the OblixPrincipal object for the request and calls the IPrincipal.InIsRole method. The OblixPrincipal object does the following:
   a. Calls the WindowsPrincipal object's IsInRole method
   b. Checks the Oracle Access Manager role
   c. Returns the answer to the ASP.NET application

      If the answer is false, the Principal object looks in the Oracle Access Manager role list for the requested role and returns the answer to the ASP.NET application.
8. The ASP.NET application completes processing and access to the resource is granted.

19

Integrating Authorization Manager Services

Oracle Access Manager 10*g* (10.1.4.0.1) provides an authorization plug-in that uses the Microsoft Windows Server 2003 Authorization Manager (AzMan) services to make authorization decisions for Access Server clients, including WebGates and callers of the Access Manager API.

This chapter explains how to configure a Oracle Access Manager policy domain for the 10*g* (10.1.4.0.1) AzMan Plug-in, and includes the following topics:

About Oracle Access Manager and the AzMan Plug-In

Authorization is the process that determines what access a user is permitted to have, and what a user is permitted to do, after they have been authenticated. Oracle Access Manager extends its access policies through authorization plug-ins.

An Oracle Access Manager authorization plug-in is a component that consists of a set of functions that reside in a dynamically-loaded native library to change or enhance Oracle Access Manager behavior. The AzMan Plug-in enables Access System authorization rules to use the facilities of the Microsoft Authorization Manager on Windows Server 2003.

When using the AzMan Plug-in:

- WebGates can control access to Web content based on Authorization Manager policies.
- Applications using the Access Manager API can use Authorization Manager policies through the ObUserSession isAuthorized() call.
- WebGates and Access Manager API clients can be on any Oracle Access Manager-supported platform, which means that:

  a. If your environment is primarily Microsoft, you can use the Authorization Manager to define policy for Windows-based applications and Oracle Access

Manager can enforce those policies in the parts of the protected applications, such as ASP URLs, for instance.

In this case, you can define application roles in the Authorization Manager and Oracle Access Manager can use these roles when enforcing Web access control.

b. If your environment includes non-Windows applications, these applications can also use Authorization Manager policies.

Non-Windows applications can use the Access Manager SDK for the Authorization Plug-in API to get authorization decisions from the Authorization Manager through the AzMan Plug-in.

The AzMan Plug-in is executed by a Access Server during the evaluation of access policies. The Authorization Plug-in API enables the Access Server to use the plug-in to make outbound calls to external business logic. The external business logic determines whether a user is authorized to access a resource. The external business logic also determines whether to pass authorization actions during the evaluation of access policies.

The Authorization Plug-in API:

- Provides the interface that the AzMan Plug-in implements and the Access Server calls
- Provides the callback functions that the plug-in uses to retrieve additional data from the Access Server
- Defines data structures that pass information between the AzMan Plug-in and the Access Server

  For example, the ObUserSession.isAuthorized() method in the Access Manager SDK for the Authorization Plug-in API can evaluate AzMan policies for a user and, optionally, for a set of parameters.

For more information, see the following discussions:

- "Authorization with the AzMan Plug-In" on page 19-3
- "Oracle Access Manager Authorization Rules and Schemes" on page 19-6
- "Oracle Access Manager Components and Requirements" on page 19-5
- "About the Windows Authorization Manager" on page 19-8.

Supported Versions and Platforms

Any references to specific versions and platforms in this chapter are for demonstration purposes.

You can find support and certification information at the following URL:

`http://www.oracle.com/technology/documentation/`

You must register with OTN to view this information.

Also, you can see the supported versions and platforms for this integration on Metalink, as follows.

To view information on Metalink

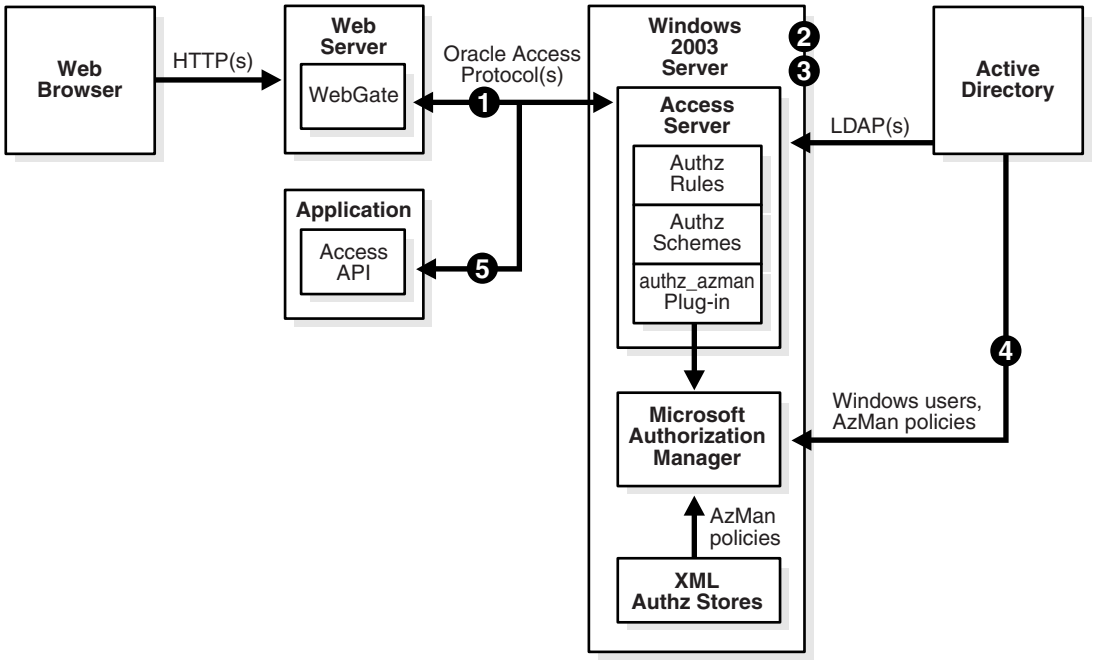
1. In your browser, enter the following URL:

   `https://metalink.oracle.com`

2. Log in to MetaLink.
3. Click the **Certify** tab.
4. Click **View Certifications by Product**.
5. Select the **Application Server** option and click **Submit**.
6. Choose **Oracle Identity Management** and click **Submit**.
7. Click **Oracle Identity Management Certification Information 10g (10.1.4.0.1) (html)** to display the Oracle Identity Management page.
8. Click the link for **Section 6, Oracle Access Manager Certification** to display the certification matrix.

Authorization with the AzMan Plug-In

The following figure introduces authorization with the AzMan Plug-in.



Process overview: WebGate operation with the AzMan Plug-in

1. The WebGate sends an ISAuthorized() request for the authenticated user and the URL to the Access Server.
2. The Access Server determines the URL is protected by a policy with an authorization rule that specifies an authorization scheme for the AzMan Plug-in.

   If the authorization scheme requires request context values (configured as RA_ user parameters in the authorization scheme) that are not available in the IsAuthorized request, the Access Server returns a NeedMoreData response to the WebGate.

   When the WebGate receives the NeedMoreData status, the WebGate:

   a. Gets the request context data indicated in the status
   b. Resends the IsAuthorized request with the data

c. Continues processing with the beginning of step 2

3. If this is the first time the plug-in has been invoked, the Access Server loads the authz_azman library and executes the ObAzPluginInit() function in the library, which:
   a. Creates the authz_azman_log.txt file in the *install_dir*/oblix/engine directory
   b. Reads the authz_azman_msg.lst file in the default language directory of the installation
   c. Checks that the Authorization API version of the Access Server is compatible with the plug-in
   d. Initializes the COM interface
   e. Creates a mutex to protect a global list of open application stores
4. The Access Server executes the ObAzPluginFn() function in the plug-in, which:
   a. Gets its configuration parameters from the various plug-in data blocks
   b. Searches the list of open application stores for a store matching the AzStore parameter
   c. If no open store is found, the plug-in opens the store and puts it in the list
   d. Creates an application object for the AzApplication parameter
   e. Initializes a client context for the browser user
   f. If an AzRole is specified, the plug-in sets the client context to the role
   g. Converts the AzOperation parameter values to an array of IDs
   h. If the AzRuleParameters is specified, the plug-in retrieves the corresponding parameter values from the plug-in data blocks and sets up arrays with the parameters and their values
   i. Calls the AzMan AccessCheck() method for the client context, the scope (if specified), the operation ID array, and the rule parameter and value arrays (if present)
   j. Interprets the result of the access check:
      – If the result is access allowed and AzContinueOnAllow=yes, the plug-in returns ObAzPluginStatusContinue, which instructs the Access Server to continue processing subsequent authorization rules (possibly invoking other plug-ins).
      – If the result is access allowed and AzContinueOnAllow=no, or is omitted, the plug-in returns ObAzPluginStatusAccessAllowed this causes the Access Server to immediately return allowed.
      – If the result is access denied, return ObAzPluginStatusAccessDenied, this causes the Access Server to immediately return denied.
5. The WebGate gets the IsAuthorized result from the Access Server and blocks or allows access to the requested URL.

Note: For more information, see "Example 3: Authorization Process Flow" on page 19-19.

Process overview: Access Manager API operation with the AzMan Plug-In

An application using the Access Manager API executes the ObUserSession IsAuthorized() method for an authenticated user, a resource, and optional set of parameters that may include Authorization Manager configuration parameters. The following process overview the operation of the Access Manager API and the Authorization Manager Plug-in.

1. The Access Manager API sends an IsAuthorized request with the authenticated user, resource, and parameters to the Access Server.
2. The Access Server determines the resource is protected by a policy with an authorization rule that specifies an authorization scheme for the AzMan plug-in.
3. If the authorization scheme requires request context values (configured as RA_ user parameters) that are not available in the IsAuthorized request, the Access Server returns a NeedMoreData status to the Access Manager API.

> **Note:** This is not as likely to happen as with WebGate, since the application can include the required parameters in the isAuthorized() call.

4. If Access Manager API gets the NeedMoreData status, it gets the request context data indicated in the status from the resource (for example, a query string) and resends the IsAuthorized request with the data.
5. Processing then continues with step 2 in "Process overview: WebGate operation with the AzMan Plug-in" on page 19-3.
6. Steps 3 and 4 in this process are the same as in "Process overview: WebGate operation with the AzMan Plug-in" on page 19-3. The parameters from the isAuthorized() call are in the request context data block.
7. The Access Manager API client in the application gets the IsAuthorized result from the Access Server and returns the result through the isAuthorized() call. The application then takes appropriate action.

For more information, see "Using the AzMan Plug-In with the Access Manager API" on page 19-25.

Oracle Access Manager Components and Requirements

The plug-in is included during Access Server installation in the following library:

`AccessServer_install_dir\access\oblix\lib\authz_azman.dll`

The AzMan Plug-in must be installed on each application server you want to protect. This enables you to complete authorization for resources protected by Oracle Access Manager using policies and roles defined outside the Access System policies, within the Authorization Manager in Active Directory, through the Authorization Plug-in API. For more information about the Authorization Plug-in API, see *Oracle Access Manager Developer Guide*.

> **Note:** Oracle Access Manager does not provide or allow administration of Authorization Manager policies through the Policy Manager.

Oracle Access Manager provides the custom AzMan Plug-in but not a custom authorization scheme, because external programs and calls to external business logic are unique from business to business.

A Master Access Administrator uses the Access System Console to define a custom authorization scheme that includes the full path to the shared library for the AzMan Plug-in.

A Delegated Access Administrator uses the Policy Manager to define a policy domain using the custom authorization scheme and plug-in parameters for the AzMan Plug-in as a basis for the authorization rules or action in a policy domain, and to protect resources.

For more information, see "Configuring the AzMan Plug-In" on page 19-21.

Oracle Access Manager Authorization Rules and Schemes

This discussion explains authorization rules and schemes and AzMan Plug-in parameters for use with the Authorization Manager.

A Oracle Access Manager authorization rule allows or denies users the right to access the resources within the policy domain (or a subset of resources, if a policy applies). Authorization rules can be combined into expressions. For more information about chaining authorization rules, see the *Oracle Access Manager Access System Administration Guide*.

Note: You define and enable authorization rules through the Policy Manager. When using the AzMan Plug-in, you need to create a custom authorization scheme that consists of a name, a description, a shared library path for the installed plug-in (without a platform-specific extension such as .dll), and a set of required and optional parameters. The purpose of the authorization scheme parameters is shown in Table 19–1.

Table 19–1 Purpose of Authorization Scheme Parameters

Scheme Parameters	Description
User Parameters: User	User profile attribute values passed into the plug-in in the RequesterInfo data structure. For more information, see on page 19-8.
User Parameters: Request Context	Request data (HTTP headers and cookies, Access Manager API parameters) passed into the plug-in in the RequestContext data structure. ■ Introduced in NP 6.1.1 and defined as user parameters with the prefix RA_. If not available in the request, the access check will fail. ■ For more information, see Table 19–3.
Required Parameters	Name-value pairs passed into the plug-in in the Context data structure. These must be specified in either the authorization scheme or the authorization rule.
Optional Parameters	Name-value pairs passed into the plug-in in the Context data structure. These may be specified in either the authorization scheme or the authorization rule or may be omitted.

The AzMan Plug-in uses optional plug-in parameters to specify the input to the AccessCheck() method, IAzClientContext::AccessCheck(), discussed in "Using the AzMan Plug-In with the Access Manager API" on page 19-25. If a plug-in parameter is not specified, the plug-in will check the User Parameters, Request Context data (see Table 19–1) for the omitted values. In this case, callers of the Access Manager API can supply these parameters in the ObResourceRequest constructor or the ObUserSession isAuthorized() call. AccessCheck() can return a value indicating that access is allowed or denied. The plug-in can take a different action based on the AzContinueOnAllow configuration parameter in Table 19–2. For details about the Access Manager API, see the *Oracle Access Manager Developer Guide*.

The plug-in parameters shown in Table 19–2 are specific to the AzMan Plug-in. You use them to specify input to the Authorization Manager.

Table 19–2 AzMan Plug-in Parameters

Parameters	Description
AzStore	URL (msldap:// or msxml://) identifying the authorization store with the relevant policies.
AzApplication	Name of the application in the store containing the policies to be used. This must be specified.
AzObject	Name of the object to be identified in the AzMan audit log. If not specified, the Oracle Access Manager resource URL will be used.
AzScope	Name of the scope in the application containing the policies to be used. If not specified, no scope will be used and the default application policies will be applied.
AzOperations	Space-separated list of operation names to be used in the access check. Operation names with embedded spaces must be enclosed in quotation marks such as "an operation". If not specified, the Oracle Access Manager resource request operation name will be used.
AzRuleParameters	Space-separated list of names of parameters to be passed to AzMan authorization rules. Parameter names with embedded spaces must be enclosed in quotation marks such as "a parameter name".
AzContinueOnAllow	▪ If AzContinueOnAllow=yes, the plug-in will return a continue status to the Access Server, executes subsequent authorization plug-ins, if any. ▪ If AzContinueOnAllow=no, or is omitted (the default), the plug-in will return an allow status and the Access Server will immediately return an allowed status for the policy evaluation.
AzLogLevel	▪ If high, all authorization requests with their parameters and result (allow, deny, continue) will be logged. ▪ Otherwise only errors are logged in: AccessServer_install_dir\oblix\engine\authz_azman_log.txt

The rule parameters (specified with AzRuleParameters in Table 19–2 are values from either the user's profile (User Parameters) or the values from the request (Request Context parameters). Rule parameters are passed to the Authorization Manager for possible use within authorization rules/scripts.

Table 19–3 shows the user parameters for IAzClientContext.

Table 19–3 User Parameters

User Parameters	Description
samacctuser	Username to construct the IAzClientContext object.

Table 19–4 shows the parameters for collecting authorization data from an external application that are configured as RA_user parameters in the authorization scheme.

Table 19–4 Request Context Parameters

Request Context Parameters	Description
AzRole	Value is used as the role in the access check.
rule parameters	Post data, query data, and all other types of data appropriate for context-specific requests can be used in an authorization decision. For post data, postgate.dll must be installed. See the *Oracle Access Manager Installation Guide* for details.

Table 19–5 summarizes what occurs when the Access Server evaluates a policy or policy domain that contains an authorization rule with a custom authorization scheme.

Table 19–5 Summary of Evaluation

The Access Server	The Plug-In
Executes the plug-in	Extracts the parameter values from the passed data
Collects relevant parameter values for the plug-in and the target user, resource, and request.	Performs its designed tasks
Adds these values to the appropriate data structures and executes the main plug-in function	Returns a result with optional actions to the Access Server, which may include continue, allow, deny, or abort.
The Access Server interprets the result and either continues processing authorization rules or stops and returns its result to the access client.	

For more information about authorization, see the *Oracle Access Manager Access System Administration Guide*. For details about the Authorization API, see the *Oracle Access Manager Developer Guide*.

About the Windows Authorization Manager

The Windows Server 2003 Authorization Manager is a role-based access control interface characterized by using collections of settings based on an object's role within an organization. The Authorization Manager provides a GUI tool to define access policy for applications and an API for applications to request access decisions using the policy. You can use role-based administration to manage users, computers, and other file-system and directory-service objects.

The Authorization Manager provides two modes of operation:

- **Developer Mode**: Enables you to create, deploy, and maintain applications with unrestricted access to all Authorization Manager features.

  You run Authorization Manager in developer mode only until the authorization store is created and configured. After you initially set up an application in developer mode, you can work in administrator mode.

- **Administrator Mode**: The default mode, enables you to deploy and maintain applications and have access to all Authorization Manager features. However, you cannot create new applications or define operations.

  Before you can use administrator mode, you must provide an application that supports roles, includes all of the necessary operation and task definitions, includes its own authorization store, and is ready for use in the Authorization Manager.

Authorization Stores

An authorization policy store contains information about the security policy of an application or group of applications. The information includes the applications, operations, tasks, users, and groups of users associated with the store.

The authorization policy store must be located on a trusted system to afford administrators on that system access to the store. The Authorization Manager supports storing authorization policy either in the Active Directory directory service or in an XML file:

- Active Directory objects are identified by an LDAP DN in a URL.

  For example:

  msldap:// (for example, msldap://CN=MyAzStore, CN=Program Data, DC=authmanager, DC=com)

or

- XML files are identified by a path in a URL.

  For example:

  msxml://C:\MyStore.xml

> **Note:** Active Directory stores allow the delegation of administrative control. However, XML stores do not.

By default, the group "Domain Admins" is listed within the Security tab when you create the Active Directory authorization store. To run the Authorization Manager policy through Oracle Access Manager, the Access Server user (for example, Administrator) should also be listed in the Users and Groups list within the Security tab. However, similar settings are not required for the XML store.

For more information about Authorization Stores, see your Microsoft documentation.

Applications and Scopes

An application is a program that is designed to perform specific functions directly for the user or for another application.

An authorization store can contain policies for resources for multiple applications. Alternatively, an application's resources and associated policies may be subdivided

into scopes. For example, if you do not want to apply Authorization Manager groups, role assignments, role definitions, or task definitions to an entire application, you can create them at the scope level.

A scope can be one of the following:

- Folder
- Active Directory container
- File-masked collection of files, for example *.doc
- URL
- Any grouping of resources meaningful to the application

You can use scopes in Active Directory authorization stores to delegate control. For more information, see your Microsoft documentation.

Operations and Tasks

In the Authorization Manager, an operation is a small computer-level action or method of an application. Operations are grouped together as tasks. An operation is defined by:

- Name
- Description
- Operation number

> **Note:** Operations can be defined at the application level but not the scope or store levels.

A task is a high-level action that users of an application need to complete. Tasks are composed of the lower-level operations required to perform the task. Users of an application request permission to complete tasks. A task is defined by:

- Name
- Description
- Set of other tasks and operations
- Authorization rule (optional)

For more information, see your Microsoft documentation.

Roles

A role is a set of permissions that a user must have to perform the application's tasks. A role is defined by a:

- Name
- Description
- Set of tasks, operations, and other roles that are granted by the role
- Authorization rules that can test arbitrary conditions

Permissions are assigned or denied by the object's owner. The Authorization Manager is capable of implementing multiple configuration and permission changes at once

and provides advantages over other management tools, such as the access control list (ACL) and Delegation of Control Wizard.

Authorization roles are based on a user's job function. You can use authorization roles to authorize access, delegate administrative privileges, or manage interaction with computer-based resources.

The Authorization Manager enables administrators to implement this role-based administration through applications. Applications using this role-based access are constructed to use logical roles that relate to the tasks performed by the application. The settings that authorize users for specific roles are made automatically through the use of scripts, called authorization rules, that enable you to control the mapping between access control and the structure of your organization.

For more information, see your Microsoft documentation.

Groups

A group defines a set of principals (users and computers) to which roles can be assigned. A group can be defined using:

- Windows users and groups
- LDAP queries
- Other groups

A group specifies principals that are either:

- Explicitly included (members)
- Explicitly excluded (non-members)

> **Note:** Circular group membership, for example, group A contains group B and group B contains group A, is detected and prohibited.

Groups can be defined at the store, application, and scope levels. Assigning a group to a role grants the role's permissions to the users defined in the group. A role definition can also contain authorization rules that can test arbitrary conditions.

For more information, see your Microsoft documentation.

Rules

In the Authorization Manager, authorization rules are either VBScript or JScript scripts that can be used in role and task definitions. An authorization rule can determine whether the role or task is allowed. With authorization rules, you can base authorization decisions on any conditions that a script can test, including privileges and permissions, time of day, billable expense limits, account balances, and other criteria.

A rule associated with an object can regulate which users gain access and in what manner. Named parameter values can be passed from the application to the Authorization Manager for use within the scripts.

You can write your scripts in a text editor (for example, Notepad), in an integrated development environment like Visual Studio .NET, or in another application of your choice.

For more information, see your Microsoft documentation.

Auditing

The Authorization Manager provides runtime auditing that records application-access checks using policies in an authorization store. The runtime audit log contains the relevant client contexts with the access checks. The Authorization Manager also provides authorization store-change auditing to record modifications to policies in authorization stores.

Runtime auditing can be applied at the authorization store and application levels for all stores, and at the scope level for Active Directory stores. Store-change auditing can be applied at the store, application, and scope levels for Active Directory stores, but only to the store level for XML stores.

For more information, see your Microsoft documentation.

Authorization Manager (AzMan) API

An application program interface (API) includes the formal requests and means of communicating with other programs used by an application program. Windows Server 2003 provides Component Object Model Component Services (COM+) interfaces to manage and use Authorization Manager policies.

COM+ provides an infrastructure that enables clients and objects to work together. This binary standard enables interoperability between software components in a networked environment regardless of the language in which they were developed.

A COM client (software that uses and controls objects) does not know the internal workings of the objects (software that knows how to perform a specific task) the client is using. Clients and objects must communicate about and agree on the functionality that an object will supply to the client. This agreement is implemented in software by a COM interface.

For example, in the Authorization Manager API:

- The state of a particular user (client) is represented by an IAzClientContext interface.
- The object is created from one of the following:
  - IAzApplication::InitializeClientContextFromToken(): needs the user's token.
  - IAzApplication::InitializeClientContextFromName(): needs the user name and domain name.
  - IAzApplication::InitializeClientContextFromStringSid(): needs the string representation of the SID.
- The AccessCheck method of the IAzClientContext interface method invokes the Authorization Manager to determine if the user represented by the IAzClientContext object is allowed to perform a specified application operation.

For more information, see your Microsoft documentation.

Examples

Topics in this section walk you through an application created with the Authorization Manager and the configuration details for the AzMan Plug-in.

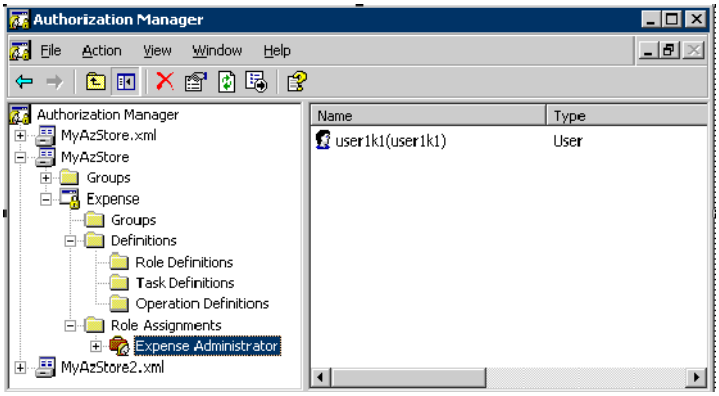
- Example 1: An Expense Application
- Example 2: Oracle Access Manager Configuration
- Example 3: Authorization Process Flow

Example 1: An Expense Application

In this example, a financial role is defined that includes the right to authorize expenditures and audit account transactions. The Authorization Manager enables you to implement this type of role-based administration through an application that you create.

You set up the authorization store, design your application using the Authorization Manager, define tasks, operations, roles, and make role assignments. Figure 19–1 shows the main Authorization Manager window and the hierarchy of the authorization store, MyAzStore.

Figure 19–1 Authorization Store Hierarchy in the Authorization Manager



In Figure 19–1 you can see the folders for Groups, Definitions, and Role Assignments for the application. Beneath the Definitions folder are the Role, Task, and Operation Definitions folders. In the right-hand panel, you can see the user assigned to the Expense Administrator role, user1k1.

The financial application, named "Expense", may have the operations shown in Table 19–6:

Table 19–6 Expense Application Operations

Name
RetrieveForm
EnqueRequest
DequeRequest
UseFormControl
MarkFormApproved
SendApprovalNotify

The Expense application may include a task, "Submit Expense", which consists of the operations in Table 19–6 and another task, Approve Expense, as shown in Table 19–7

Table 19–7 Expense Application Submit Expense Task Definition

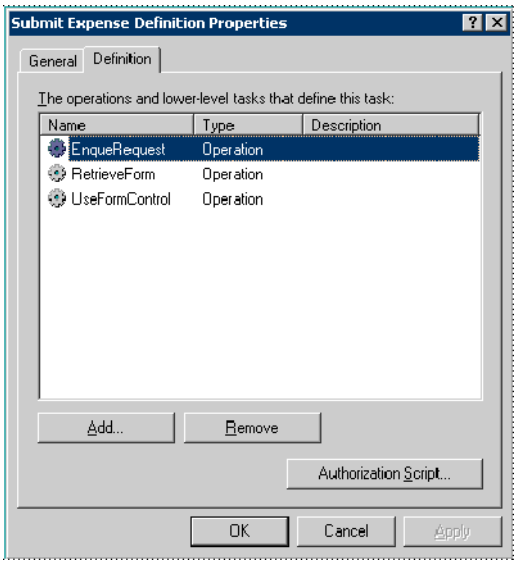
Name
RetrieveForm
EnqueRequest

Table 19–7 (Cont.) Expense Application Submit Expense Task Definition

Name
UseFormControl
Approve Expense

Figure 19–2 shows the Submit Expense task, as it appears in the Authorization Manager.

Figure 19–2 Submit Expense task in the Authorization Manager



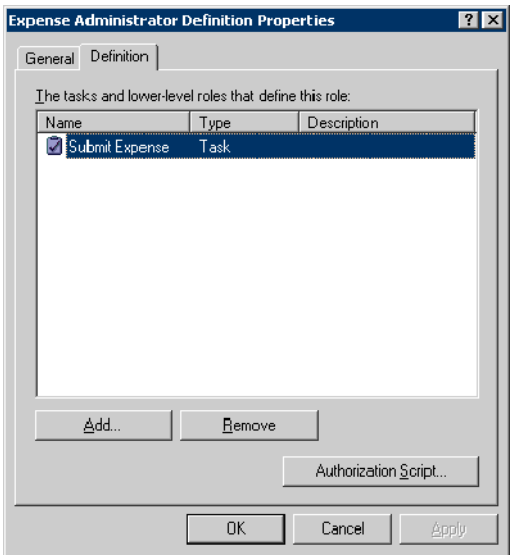
The Expense application includes a role, Expense Administrator, that consists of the tasks in Table 19–8. A user who is assigned the Expense Administrator role is authorized to perform the operations (See Table 19–7) to complete the Submit Expense task, among others identified as follows.

Table 19–8 Tasks for the Expense Administrator Role

Tasks
Submit Expense
Approve Expense
Nested role Expense Admin

Figure 19–3 shows the Expense Administrator role-definition properties in the Authorization Manager. The Submit Expense task is identified; other tasks will be added.

Figure 19–3 Expense Administrator Role-Definition Properties



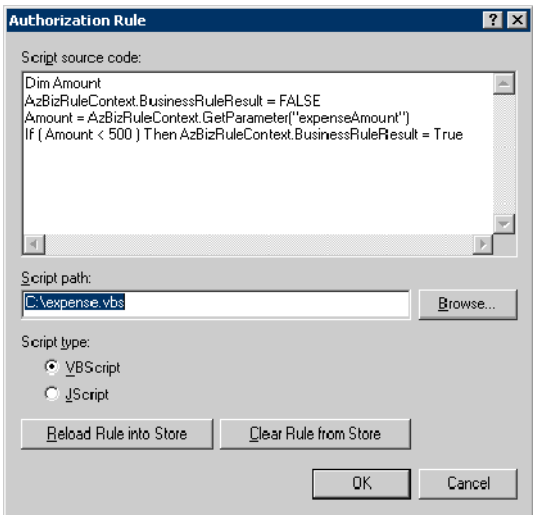
The Expense application may include a group of "Approvers", to which the Expense Administrator role can be assigned. Members of the Approvers group are given permission to perform the tasks in Table 19–9.

Table 19–9 Approvers Group Tasks

Tasks
Submit Expense
Approve Expense tasks
Any tasks assigned to the nested Expense Admin role

An authorization rule for the Expense application is a script that tests the user's expense amount (a parameter from the application) against the user's expense limit, which could either be another application parameter or could be determined by the script itself. The rule for this Expense application is shown in Figure 19–4.

Figure 19–4 Authorization Rule for the Expense Application



Example 2: Oracle Access Manager Configuration

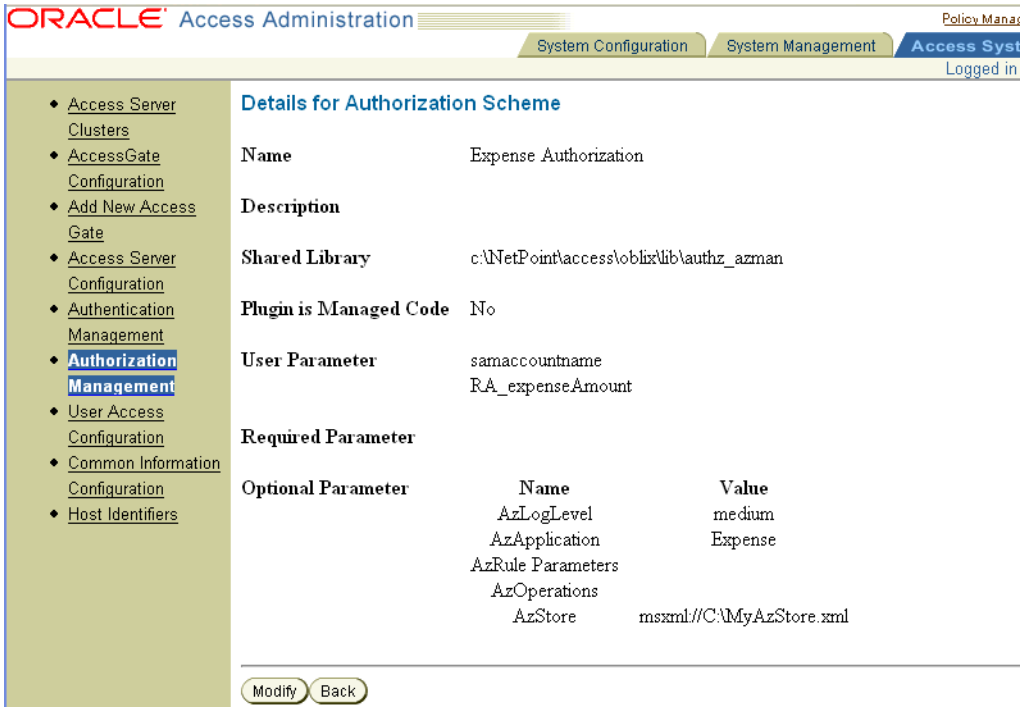
Continuing with the Expense application example described and shown in "Example 1: An Expense Application" on page 19-13, the following information explores the Oracle Access Manager policy domain for this application. Included is a custom authorization scheme for the AzMan Plug-in.

The Expense application has been implemented to use Web forms to input the expense data. An XML file is used, rather than storing Authorization Manager policies in the Active Directory. Both methods are valid.

Authorization Scheme

Figure 19–5 shows a custom authorization scheme for the Expense application. Not all of the allowable AzMan Plug-in parameters are used. Your scheme may be different.

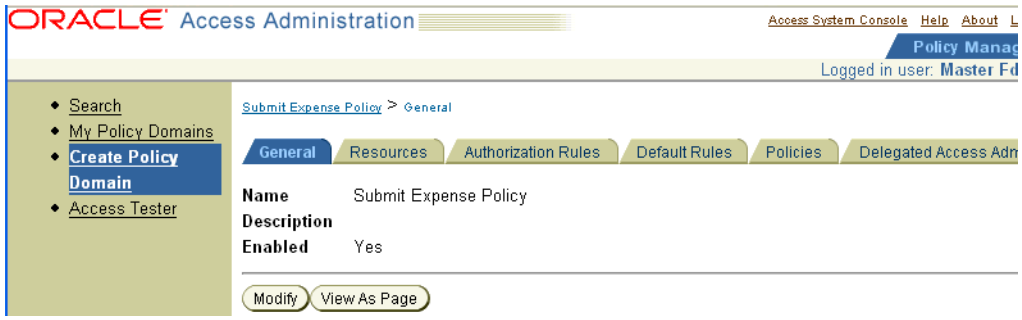
Figure 19–5 Oracle Access Manager Authorization Scheme



Policy Domain

Figure 19–6 shows the Submit Expense policy domain, enabled in the Policy Manager.

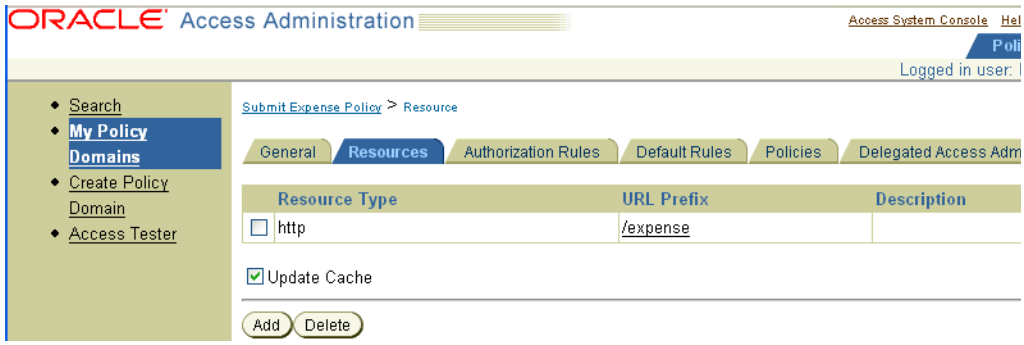
Figure 19–6 Submit Expense Policy Domain in the Policy Manager



Resources

Within the policy domain, resources have been added and protected, as shown in Figure 19–7 for /expense/submit.asp.

Figure 19–7 Resource Types in the Oracle Access Manager Policy Domain



Authorization Rules

The policy domain authorization rule is as follows. Notice that it specifies the custom authorization scheme defined in the Access System Console earlier.

Name: Expense Authn

Description:

Authorization Scheme: Expense Authorization

There are no timing conditions for this specific authorization rule, though your rule may include these.

The plug-in parameters for this policy domain are shown next:

Profile attributes that are passed to the plug-In:

- samaccountname
- RA_expenseAmount

Optional parameters and values:

- Name: AzLogLevel

  Value: medium
- Name: AzApplication

  Value: Expense
- Name: AzRule Parameters

  Value: No Value Specified
- Name: AzOperations

  Value: *No Value Specified*
- Name: AzStore

  Value: msxml

The authorization rule uses the custom Expense Authorization scheme and passes the User Parameters and AzStore and AzApplication parameters as specified in the scheme. The rule adds an AzOperations value for the EnqueRequest operation and an AzRuleParameters value for the expenseAmount variable.

There are no actions associated with this particular rule; however, your application may have specific actions.

Default Rules

The default authentication rule for the policy domain is as follows. There are no authorization expressions or audit rules for this policy domain. Your environment may be different.

Name**:** Basic over LDAP

Description:

Authentication Scheme: Basic over LDAP

Access Policy

Next you see the access policy for /expense/submit.asp:

Name: AzMan Policy

Description:

Resource Type: http

Resource Operation(s) GET

POST

Resource ell

URL Pattern /expense/submit.asp

There are no authentication rules, authorization expressions, or audit rules defined for this policy.

Delegated Access Administrators

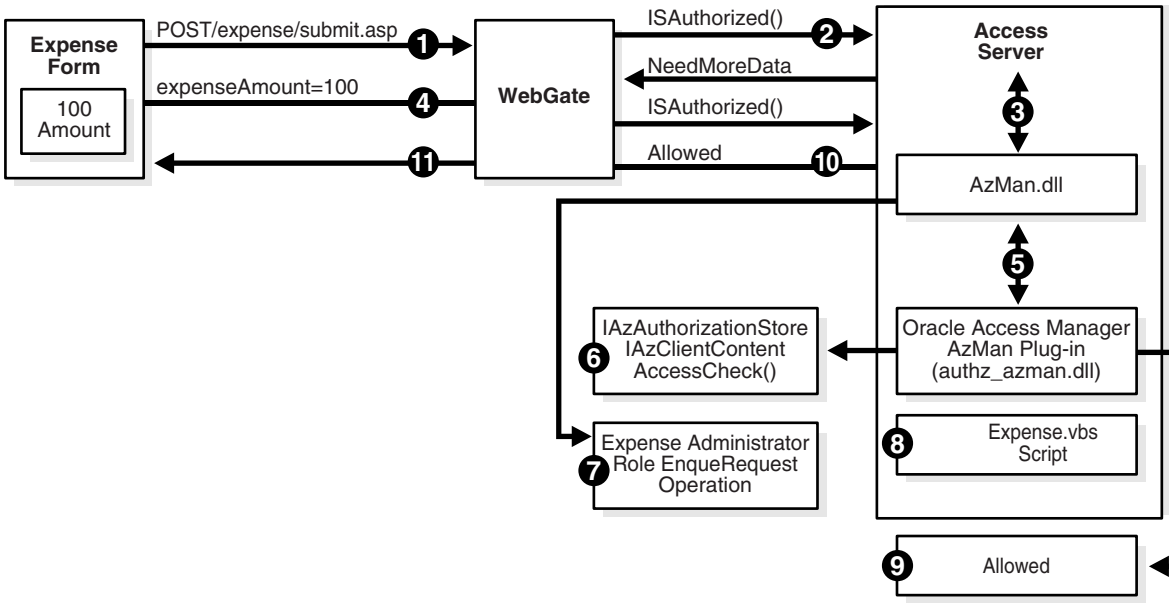
Delegated Access Administrators are defined for this policy domain, but are not shown here.

The authorization flow using the example that was implemented in the previous paragraphs is described next.

Example 3: Authorization Process Flow

The following scenario walks you through the process flow for the Oracle Access Manager and AzMan plug-in authorization process. This process flow remains the same no matter where the Authorization Manager resides. In the following scenario:

- An Expense application was implemented to use Web forms to input expense data, as explained under "Example 1: An Expense Application" on page 19-13.
- An XML file is used for Authorization Manager policies, rather than storing these policies in the Active Directory. Both methods are valid, as described in "Authorization Stores" on page 19-9.
- The resource is protected by an Oracle Access Manager policy with an authorization rule based on a custom authorization scheme that passes parameters to the AzMan Plug-in, as described in "Using the AzMan Plug-In with the Access Manager API" on page 19-25



Process overview: AzMan authorization after a user is authenticated

1. The user (user1k1) submits an expense form to /expense/submit.asp using the Authorization Manager Submit Expense task.
2. The WebGate intercepts the POST request to /expense/submit.asp and sends an ISAuthorized() request for user1k1, and the URL, to the Access Server.
3. The Access Server passes the parameter to the AzMan.dll, which applies the Submit Expense Policy, determines the expenseAmount variable is needed, and returns a NeedMoreData response to the WebGate.
4. The WebGate retrieves expenseAmount=100 from the POST data and re-sends the ISAuthorized() request with the data to the Access Server.
5. The Access Server:
   a. Applies the Submit Expense Policy again
   b. Executes the AzMan Plug-in for the Expense Authorization Scheme
   c. Passes the expenseAmount variable in the RequestContext data and the samaccountname for the user in the Requestor data
6. The AzMan Plug-in:
   a. Constructs an IAzAuthorizationStore object, which in this case is for msxml://C:\MyAzStore.xml
   b. Constructs an IAzApplication object for the Expense application
   c. Constructs an IAzClientContext object for user1k1
   d. Calls AccessCheck() for the client context with the following:

```
bstrObjectName = /expense/submit.asp
varScopeNames = {}
varOperations = {EnqueRequest}
varParameterNames = {expenseAmount}
varParameterValues = {100}
varInterfaceNames = {}
varInterfaceFlags = {}
```

```
varInterface = {}
```

7. The Authorization Manager runtime:
   a. Determines that user1k1 is assigned to the Expense Administrator role

      The Expense Administrator role can perform the Submit Expense and Approve Expense tasks and includes the EnqueRequest operation.

   b. Determines that user1k1 is allowed to perform the EnqueRequest operation
8. The Authorization Manager executes the expense.vbs script with expenseAmount=100, the script tests expenseAmount < 500 and returns a BusinessRuleResult of TRUE.
9. The AzMan Plug-in receives the allowed result and returns Allowed.
10. The Access Server returns an allowed response to WebGate.
11. WebGate allows the POST request processing to proceed.

Configuring the AzMan Plug-In

The following information is provided to guide you during the configuration needed to use the AzMan Plug-in. Some information is tailored for the Expense example discussed earlier. Your specifications may be different. Sample screens are presented in "Example 2: Oracle Access Manager Configuration" on page 19-16.

Task overview: Configuring the AzMan Plug-in

1. Prepare your environment, as described in "Preparing Your Environment" on page 19-21.
2. Configure authorization schemes, as described in "Creating an Authorization Scheme for the AzMan Plug-In" on page 19-22.
3. Protect resources, as described in "Protecting Resources" on page 19-23.
4. Configure authorization rules and policies, as described in "Defining Authorization Rules and Policies" on page 19-23.
5. Use the AzMan plug-in, as described in "Using the AzMan Plug-In with the Access Manager API" on page 19-25.

For a process overview, see "Using the AzMan Plug-In with the Access Manager API" on page 19-25.

Preparing Your Environment

The following procedures must be completed before you begin.

Task overview: Preparing your environment

1. Install and set up Windows Server 2003 on the machine that will host the Access Server, as described in your Microsoft documentation.
2. Install and set up Oracle Access Manager, as described in the *Oracle Access Manager Installation Guide*.

   The AzMan Plug-in is included with the Access Server, as discussed under "Oracle Access Manager Components and Requirements" on page 19-5.
3. Set up the AzMan authorization store, azman.msc, as described in your Microsoft documentation.

Note: By default, the group "Domain Admins" is listed within the Security tab when you create the Active Directory authorization store. The Access Server user (for example, Administrator) should also be listed in the Users and Groups list within the Security tab. Similar settings are not required for the XML store.

4. Design your application using the Authorization Manager to specify operation, task, and role definitions and role assignments, as described in your Microsoft documentation.

Creating an Authorization Scheme for the AzMan Plug-In

The following steps presume that you have already defined an authentication scheme for this policy domain. The authorization scheme that you create can be included with any policy domain or policy and must include an authorization rule.

When you create a custom authorization scheme be sure to enter the full path to the shared authz_azman library (without the extension). You must also specify the user profile attribute values to be passed to the plug-in with the RequesterInfo data structure (the username is used to construct the IAzClientContext object). Also, specify the AzMan Plug-in parameters needed for your own application. For more information about policy domains and authorization rules, see the *Oracle Access Manager Access System Administration Guide*.

To create a custom authorization scheme

1. Navigate to the Access System Console:

   http://*hostname:port*/access/oblix

2. Within the Access System Console, click Access System Configuration, Authorization Management.

3. Click the Add button to begin a custom authorization scheme.

4. Enter the information for your custom authorization scheme.

   For example:

   Name: Name of this custom authorization scheme

   Description: Optional description.

   Shared Library: Full path to the authz_azman library (without the extension)
   c:*coreid*\access\oblix\lib\authz_azman

   User Parameter: samaccountname
   RA_expenseAmount
   (The RA_expenseAmount is a reverse action user parameter that is needed only if the authorization rule expects parameters).

   Optional Parameters:

   zAzApplication Expense
   AzRuleParameters
   AzOperations
   AzStore msxml://C:\MyAzStore.xml

5. Save the scheme, as usual.

Protecting Resources

You need to create a policy domain and add resources to protect. For general information about policy domains, see the *Oracle Access Manager Access System Administration Guide*.

To create a policy domain and add a resource

1. Click the Policy Manager link or navigate to the Access System administration URL and select the Policy Manager application:

   http://*hostname:port*/access/oblix

2. Click Create Policy Domain in the left navigation pane and create a policy domain.

   For example:

   Name: Submit Expense
   Description: Optional

   > **Note:** Do not enable the policy domain until you have finished all specifications for it, as described next.

3. Click Save.

4. Add a resource to protect with this policy domain: Policy Manager, My Policy Domains, *link*, Resources, Add

   For example:

   Resource Type: http
   URL Prefix: /expense
   Description: Optional

5. Click Save.

Defining Authorization Rules and Policies

You need to add the custom authorization scheme you created earlier to an authorization rule. The following steps presume that you have already defined your authentication rule for this policy domain.

To add the authorization scheme to the authorization rule

1. Navigate to Authorization Rules page: Policy Manager, My Policy Domains, *link,* Authorization Rules.

2. Click the Add button to display the Create Authorization Expression page.

3. Select Custom Authorization Scheme from the list, then click Add.

   For example:

   Authorization Scheme: Custom Authorization Scheme

   A new page appears where you enter details for this rule.

4. Enter the details for this authorization rule, and confirm that the authorization scheme you created earlier is selected in the Authorization Scheme list.

   For example:

Name: Expense Authn
Description: Optional
Authorization Scheme: Expense Authorization

5. Save the rule, as usual.
6. Return and click Plug-in Parameters; confirm the profile attributes to be passed to the plug-in from the authorization scheme match those you specified in your custom authorization scheme.

   For example:
   Profile Attributes Passed to Plug-In: samaccountname
   RA_expenseAmount

Optional Parameters:	Name	Value
	AzLogLevel	medium
	AzApplication	Expense
	AzRuleParameters	No Value
	AzOperations	No Value
	AzStore	msxml

7. Modify and save, if needed.
8. Add timing considerations and actions, as needed for your application.

To add default rules and the authentication rule

1. Click the Default Rules link.
2. Click the Add button on the Default Rules page to add an authentication rule, which includes an authentication scheme.
3. Enter the details and save as usual.

   For example:

   Name: AzMan Basic Over LDAP
   Description: Optional
   Authentication Scheme: Basic Over LDAP

To add access policies

1. Click the Policies link to add an access policy for the application.
2. Click the Add button on the Policies page.
3. Fill in the requested information for your application and policy domain.

   For example:

   Name: SubExp Access Policy
   Description: Refines control of the resource
   Resource type: http
   Resource Operations: GET POST
   Resource: All
   URL Prefix: /expense
   URL Pattern: /expense/submit.asp

   This policy contains no query string or query string variables.

4. Save the policy, as usual.

Delegating Administration is done as usual. There are no special requirements for the application in this example. For more information, see the *Oracle Access Manager Identity and Common Administration Guide*.

5. Click the General tab and enable the policy domain, as usual.

You request the resource as usual and Oracle Access Manager will complete the authorization process as described under "Example 3: Authorization Process Flow" on page 19-19.

Using the AzMan Plug-In with the Access Manager API

The following example is provided as a guide if you want to use the Access Manager API with the AzMan Plug-in. For general information about the Access Manager API, see the*Oracle Access Manager Developer Guide*.

```
// Set up the Expense resource.

ObResourceRequest rr = new ObRequestRequest("http", "/expense/submit.asp",
"POST");

// Authenticate DOMAIM\jsmith.

Hashtable creds = new Hashtable();
creds.put("username", "user1k1");
creds.put("password", "oblix");
ObUserSession user = new ObUserSession(rr, creds);

// Check if administrator is authorized to submit an expense form with
expenseAmount=100.
// This uses the AzStore and AzApplication parameters defined by the Expense
// Authorization scheme and the AzOperations and AzRuleParameter expenseAmount
// defined by the Submit Expense Authorization Rule.
//
// Equivalent access_test_cplus command:
// user1k1 oblix GET http://dotnet/expense/submit.asp dotnet expenseAmount=100

Hashtable parameters = new Hashtable(); parameters.put("expenseAmount", "100"); if
(user.isAuthorized(rr, parameters)) {
     // authorized
}
else {
    // not authorized.
}

// Check if administrator is authorized to perform the UseFormControl operation in
// the Expense application. This uses the AzStore and AzApplication parameters
// defined by the Expense Authorization scheme but overrides the AzOperations
// parameter in the Submit Expense Authorization Rule.
//
// Equivalent access_test_cplus command:
// user1k1 oblix GET http://dotnet/expense/submit.asp dotnet
// expenseAmount=100&AzOperations=UseFormControl

parameters.put("AzOperations", "UseFormControl"); if (user.isAuthorized(rr,
parameters)) {
     // authorized
}
else {
     // not authorized.
```

```
}

// Check if the Expense Administrator role is authorized to perform the
// UseFormControl operation in the Expense application. Note that user1k1 must
// have this role.
//
// Equivalent access_test_cplus command:
// user1k1 oblix GET http://dotnet/expense/submit.asp dotnet
// expenseAmount=100&AzRole=Expense+Administrator&AzOperations=UseFormControl
//
// Note that access_test_cplus does not actually convert + to blank, but it
should.

parameters.put("AzRole", "Expense Administrator");
if (user.isAuthorized(rr, parameters)) {
// authorized
}
else {
// not authorized.
}
```

Troubleshooting

An "Insufficient access right" error may appear in the log file when the Access Server user (for example, Administrator) does not appear in the Security tab of the Active Directory authorization store.

By default, the group "Domain Admins" is listed within the Security tab when you create the Active Directory authorization store. To run the Authorization Manager policy through Oracle Access Manager, the Access Server user (for example, Administrator) should also be listed in the Users and Groups list in within the Security tab. However, similar settings are not required for the XML store.

20

Enabling Impersonation with the Access System

In a Windows environment, after a user authenticates, the authenticating application can impersonate that user's impersonation. Impersonation is implemented on a thread-by-thread basis. The primary purpose of impersonation is to trigger access checks against a client's identity.

This chapter discusses enabling impersonation in the Access System to override impersonation enabled with IIS. See the following topics:

> **Note:** "Integrating SharePoint Server" on page 17-1 provides a detailed example of how to integrate with the SharePoint Portal Server as well as the extra measures you may have to take to get impersonation running in different contexts

About Windows Impersonation

When running in a client's security context, a service can to an extent become a client. After the user authenticates, the service can take on that user's identity through impersonation. One of the service's threads uses an access token, known as an impersonation token, to obtain access to objects the client can access. The access token is a protected object that represents the client's credentials.

The impersonation token identifies the client, the client's groups, and the client's privileges. The information in the token is used during access checks when the thread requests access to resources on the client's behalf. When the server is impersonating the client, any operations performed by the server are performed using the client's credentials.

Impersonation ensures that the server can or cannot do exactly what the client can or cannot do. Access to resources can be restricted or expanded, depending on what the client has permission to do. Impersonation requires the participation of both the client and the server. The client must indicate its willingness to let the server use its identity, and the server must explicitly assume the client's identity programmatically.

When impersonation concludes, the thread uses the primary token to operate using the service's own security context rather than the client's. The primary token describes the security context of the user account associated with the process (the person who started the application).

Services run under their own accounts and act as users in their own right. For example, system services that are installed with the operating system run under the Local System account. You can configure other services to run under the Local System account, or separate accounts on the local system or in Active Directory.

The IIS Web server provides impersonation capabilities. However, the Access System overrides IIS authentication, authorization, and impersonation functions. For more information, see:

- "About Impersonation and the Access System" on page 20-2
- "Windows Impersonation Background" on page 20-18

Single sign-on for Authenticated Oracle Access Manager Users into Exchange: This is also supported using the Windows Impersonation feature. OWA provides Web access to Exchange mail services and may be configured on either of the following:

- An IIS Web server that does not reside on the same server as the Exchange server, which is also known as a front-end server
- An IIS Web server running on the Exchange server, which is also known as the back-end server

In a front-end server configuration, the front-end OWA server authenticates the user, determines the back-end Exchange server that hosts the user's mailbox, then proxies the request to the appropriate back-end Exchange server. No additional credential information is passed. No delegation is performed. Setting up Impersonation on the back-end Exchange server ensures that the Exchange server does not need to request credentials before granting access.

For more information, see "Setting Up Impersonation for OWA" on page 20-13.

About Impersonation and the Access System

You can enable support for Windows impersonation to provide additional access control for protected applications. You bind a trusted user to a WebGate and protect the application with a policy domain that includes an impersonation action in the authorization rule. During the authorization process, the protected application creates an impersonation token.

Table 20–1 identifies Access System support for Windows impersonation.

Table 20–1 Support for Windows Impersonation

Access System Version 6.5 and Higher Supports	Previous Versions Supported
Microsoft Kerberos Service-for-User-to-Self (S4U2Self) extension	User name and password required. LOGON_USER, LOGON_PASSWORD (in authorization rule, action)

Table 20–1 (Cont.) Support for Windows Impersonation

Access System Version 6.5 and Higher Supports	Previous Versions Supported
The Impersonate HeaderVar action type is as an authorization rule action in the Access System	User name (LOGON_USER) used in proper header variables.
No password needed	Password (LOGON_PASSWORD) stored in a directory in clear text or in a separate database, not set as a header variable.
REMOTE_USER may be set to any value in Authorization Rule, Action (type HTTP).	No change

For more information, see "The Kerberos Protocol" on page 20-20 and "The S4U2Self Extension" on page 20-20. Also, see the following:

- "Enabling Impersonation With a Header Variable" on page 20-3 provides prerequisites and details about implementing impersonation using header variables.
- "Enabling Impersonation with a User Name and Password" on page 20-12 explains how to implement impersonation using features available *before* version 6.5.

Enabling Impersonation With a Header Variable

Enabling impersonation with a header variable involves the following procedures.

Task overview: Enabling impersonation with a header variable includes

1. Reviewing all Requirements
2. Creating an Impersonator as a Trusted User
3. Assigning Rights to the Trusted User
4. Binding the Trusted User to Your WebGate
5. Adding an Impersonation Action to a Policy Domain
6. Adding an Impersonation DLL to IIS
7. Testing Impersonation

> **Note:** The example in this chapter illustrates setting up the impersonation feature for the Access System to Microsoft SharePoint Portal Server integration. The principles are the same regardless of your application.

See also "Setting Up Impersonation for OWA" on page 20-13.

Requirements

Prepare the environment and confirm that it is operating properly before implementing impersonation with the Access System.

Table 20–2 identifies the platform requirements for version 6.5 and later when you enable impersonation using a header variable.

Table 20–2 Version 6.5 and Later Requirements for Impersonation with a Header Variable

Item	Platform
WebGate (and Impersonation dll)	Microsoft IIS 6.x and Windows Server 2003 Note: Other Access System components have no specific requirements.
Impersonation dll	*WebGate_install_dir*\access\oblix\apps\webgate\bin ▪ Must be installed as an IIS wildcard extension. ▪ May be installed at any level of the Web site tree. For details, see "Wildcard Extension" on page 20-20.
Kerberos Key Distribution Center (KDC) and Active Directory	Windows Server 2003
Client and Server machines	▪ Both must be in the same Windows Server 2003 domain with a trust relationship. ▪ A bidirectional trust path is required because the service, acting on the client's behalf, must request tickets from the client's domain.
Security context	Must have *Act as operating system* privileges. Note: IWAM_Machine is not recommended because it is the account used by the Microsoft Transaction Server (MTS) and various IIS entities to provide programmatic and transactional functions.
Mutual authentication is required	Mutual authentication is supported remotely.

Creating an Impersonator as a Trusted User

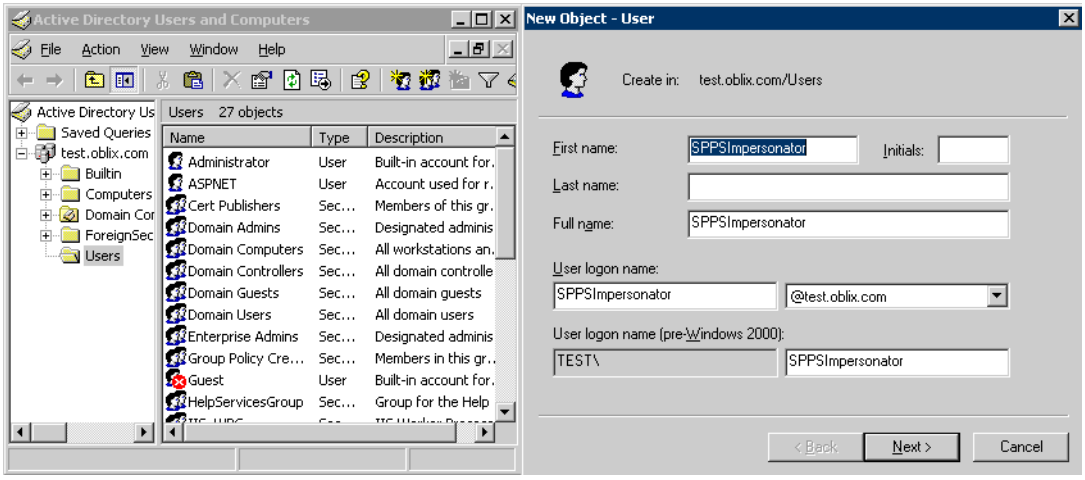
Whether you enable impersonation using a HeaderVar or user profile attribute, the return value must be a trusted user in Active Directory. This special user should not be used for anything other than impersonation.

To create a trusted user account

1. On the Windows 2003 machine hosting your SPPS installation, select Start; Programs; Manage Your Server; Domain Controller (Active Directory); Manage Users and Computers in Active Directory.
2. In the Active Directory Users and Computers window, right-click Users on the tree in the left pane, then select New; User.
3. In the First name field of the pane entitled New Object - User, enter an easy-to-remember name such as *SPPSImpersonator*.
4. Copy this same string to the User logon name field, then click Next.
5. In succeeding panels, you are asked to choose a password and then retype it to confirm.

Note: Oracle recommends that you choose a very complex password, because your trusted user is being given very powerful permissions. Also, be sure to check the box marked Password Never Expires. Since the impersonation extension should be the only entity that ever sees the trusted user account, it would be very difficult for an outside agency to discover that the password has expired.

Figure 20–1 Setting up a Trusted User Account for Windows Impersonation



Assigning Rights to the Trusted User

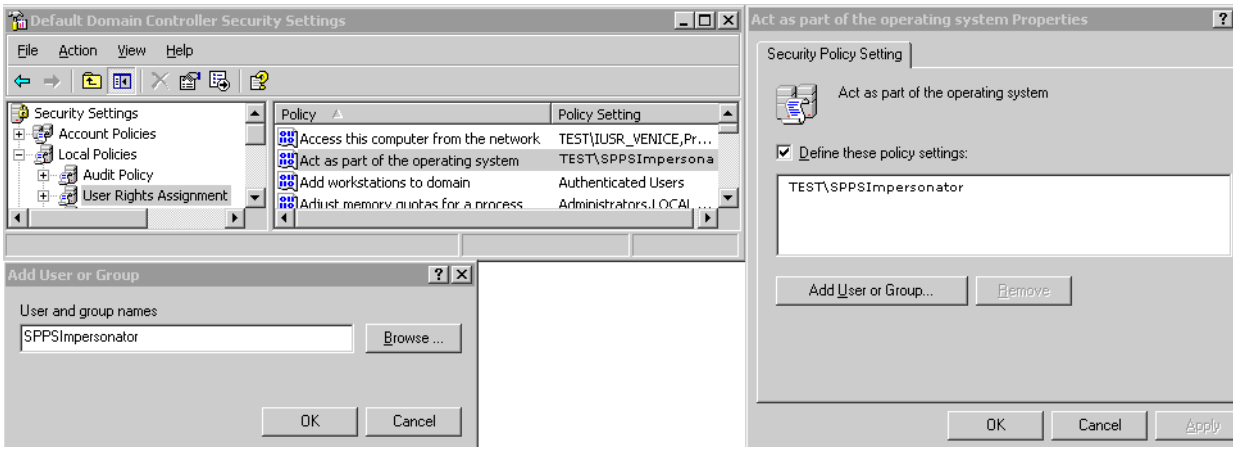
You need to give the trusted user the right to act as part of the operating system

To give appropriate rights to the trusted user

1. From the desktop, click the Start menu, then click Control Panel, and open Administrative Tools.
2. Select Domain Controller Security Policy or Local Security Policy (depending on if the computer is a domain controller).

   You must modify the group policy object that applies to the computer where the WebGate is installed.
3. On the tree in the left pane, click the plus icon (+) next to Local Policies.
4. Click User Rights Assignment on the tree in the left pane.
5. Double-click "Act as part of the operating system" in the right pane.
6. Click Add User or Group.
7. In the Add User or Group panel, type the User logon name of the trusted user (SPPSImpersonator in our example) in the User and group names text entry box, then click OK to register the change.

Figure 20–2 Configuring Rights for the Trusted User in Windows Impersonation



To configure Active Directory settings for the trusted user

1. In Active Directory, assign the truster user (in this example, SPPSImpersonator) the Allowed to Authenticate permission for all user objects that the account will impersonate.
2. If the option Do Not Require Kerboeros Preauthentication is set on any user account that the SPSSImpersonate account will be impersonating, remove this option from the account.
3. Assign the following Property Right to the trusted user account (in this example, SPPSImpersonate): Read Remote Access Information.

Binding the Trusted User to Your WebGate

You need to bind the trusted user to the WebGate by supplying the authentication credentials for the trusted user, as described in the following paragraphs.

To bind your trusted user to your WebGate

- Point your browser to your Access System Console.

  For example:

  ```
  http://hostname.domain.com:port/access/oblix
  ```

  where *hostname* is the DNS name of the machine hosting your Policy Manager, domain is the name of the server *domain* to which the machine belongs, and port is the number of the *port* to which Policy Manager listens.

- From the Access System Console, click Access System Configuration, then click AccessGate Configuration.
- Select the name of the WebGate you want to modify.

  The Details for AccessGate page appears with a summary of the configuration information for this WebGate. At the bottom of this Web page are fields for Impersonation Username and Impersonation Password.

- Click the Modify button at the bottom of the page.
- In the Modify AccessGate page, scroll to the bottom and enter the user name and password for the trusted user account you created through the task in "Assigning Rights to the Trusted User" on page 20-5.

For example:

- Click the Save button to commit the changes and return to the Details page.

  A bind has been created for the WebGate and the trusted user. The WebGate is now ready to provide impersonation on demand. The demand is created by an Authorization Success Action in a policy domain created for impersonation.

Adding an Impersonation Action to a Policy Domain

You must create or configure a policy domain to protect your SharePoint resources. You do this by adding an Authorization Success Action with a return type of headerVar, the name parameter set to the name of the trusted user (SPPSImpersonator in our example), and the return attribute parameter set to samaccountname for a single-domain Active Directory installation or userPrincipalName for a multi-domain Active Directory forest.

You must also choose an easy-to-remember name for the domain, such as *ImpersonationPolicyDomain*.

For details on creating a policy domain, see the chapter on protecting resources with policy domains in the *Oracle Access Manager Access System Administration Guide*.

To add an impersonation action to your policy domain

1. Point your browser to the Access System Console. For example:

   ```
   http://hostname.domain.com:port/access/oblix
   ```

   where *hostname* is the DNS name of the machine hosting your Policy Manager, *domain* is the name of the server domain to which the machine belongs, and port is the number of the *port* to which Policy Manager listens.

2. Navigate to the Authorization Definitions page of the policy domain you want to change:

   In the Policy Manager, click My Policy Domains, then click PolicyName, then click Authorization Rules

   where *PolicyName* refers to the policy domain you created specifically for impersonation (*ImpersonationPolicyDomain* in our example).

   Note: Currently defined authorization rules are listed. If none are listed, click the Add button and complete the form to create one.

3. Click the link to the rule to which you want to add the impersonation action. The description will expand.

4. Click the Actions link, which appears directly under the Authorization Rules tab.

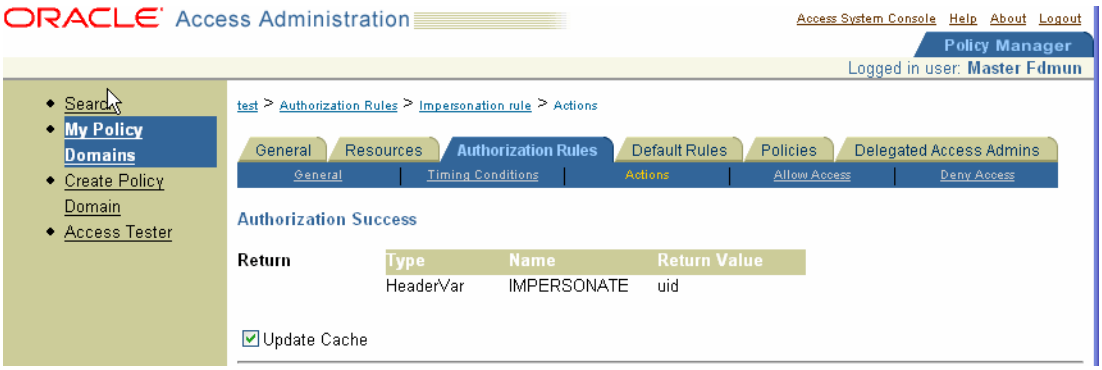
   The Authorization Success page appears. If no actions are identified, you must add them. If actions are provided, you can modify them.

   You need to add a header variable named impersonate to Authorization Success Action in the policy domain for impersonation.

5. On the Authorization Success page appears, click Add or Modify.
6. Complete the form using headerVar as the Return Type, the User log on name of the trusted user you have bound to the WebGate, and the appropriate return value for your environment. For example:

   Type: HeaderVar
   Name: IMPERSONATE
   Return value: uid or samAccountName (Active Directory username, the Windows domain user for the desired folder)

   Your completed form may look something like the following:



7. Save the rule.

This rule is used for the second WebGate request (for authorization).

Adding an Impersonation DLL to IIS

You are ready to configure IIS by adding the IISImpersonationExtension.dll to your IIS configuration.

To add the impersonation DLL to your IIS configuration

1. Select Start; Administrative Tools; Internet Information Services (IIS) Manager.
2. Click the plus icon (+) to the left of the local computer icon on the tree in the left pane.
3. Click Web Service Extensions on the tree in the left pane.
4. Double-click WebGate in the right panel to open the Properties panel.
5. Click the Required Files tab.
6. Click Add.
7. In the Path to file text box, type the full path to IISImpersonationExtension.dll.

   By default, the path is:

   WebGate_install_dir \access\oblix\apps\webgate\bin\IISImpersonation\Extension.dll

   where *WebGate_install_dir* is the root directory of your WebGate installation.

Note: If any spaces exist in the path (for example, C:\Program Files\Oracle\...) surround the entire string with double quotes (" ").

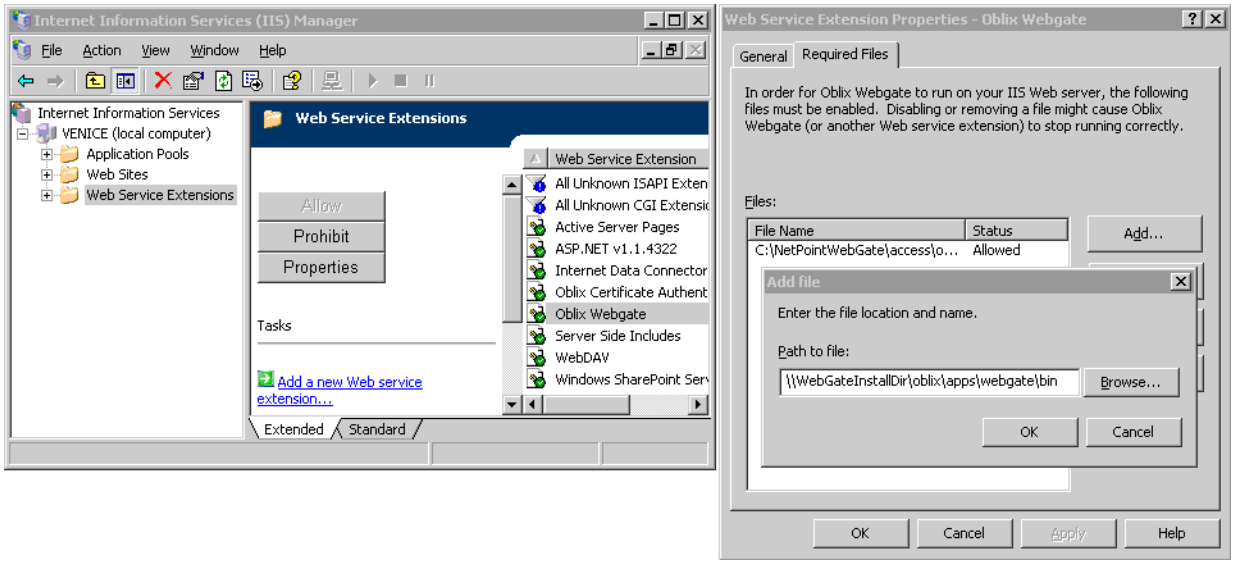
---

8. Click OK.
9. Add the IISImpersonationExtension.dll to the Wildcard Application Maps for the WebSite or IIS Virtual directory.

   To do this, from the Internet Services Manager, go to WebSite/Virtual Directory Properties, then to the Home Directory, then to Configuration, then to Mappings and add IISImpersonationExtension.dll in Wildcard Applications maps.
10. Verify that the Allow button to the left of the WebGate icon is greyed out, which indicates that the dll is allowed to run as a Web service extension.

Note: If Allow is not greyed out, click it so that it becomes greyed out. When Allow is greyed out, this indicates that the highlighted file is permitted to run on the IIS virtual server.

Figure 20–3 Configuring IIS Security Settings



Extending Impersonation to Resources Beyond the WebGate's Host Computer

In addition to configuring impersonation for resources on the computer that is protected by a WebGate, you can extend impersonation to other resources on the network. This is known as assigning a Delegate impersonation level to the client.

Note: More information on delegation is provided at the following URL:

`http://www.microsoft.com/technet/prodtechnol/windows2000serv/reskit/distrib/dsce_ctl_awdg.mspx?mfr=true`

To extend impersonation to resources beyond the computer protected by a WebGate

1. In Active Directory Users and Computers, right-click the trusted user account that performs impersonation.
2. Click Properties, then click Accounts.

3. In the Account Options dialog box, de-select the option "Account is sensitive and cannot be delegated" if it is selected.
4. In Active Directory Users and Computers, right-click the user account for the OblixService.
5. Click Properties, then click Accounts.
6. In the Account Options dialog box, select the option "Account is trusted for delegation."

Testing Impersonation

You can test Impersonation in the following two ways:

- Testing Impersonation Using the Event Viewer
- Testing Impersonation using a Web Page

Creating an IIS Virtual Site Not Protected by SPPS

To test the impersonation feature outside the SPPS context or to test single sign-on, you will need a target Web page on an IIS virtual Web site that is not protected by SPPS. You create such a virtual Web site by completing the following task.

To create an IIS virtual site not protected by SPPS

1. Click the Start menu, then click Administrative Tools, then click Internet Information Services (IIS) Manager.
2. Click the plus icon (+) to the left of the local computer icon on the tree in the left pane.
3. Right-click Web Sites on the tree in the left pane, then select New, then select Web Site on the menu.
4. Respond to the prompts by the Web site creation wizard.
5. After you create the virtual site, you must protect it with a policy domain, as described elsewhere in this guide.

Testing Impersonation Using the Event Viewer

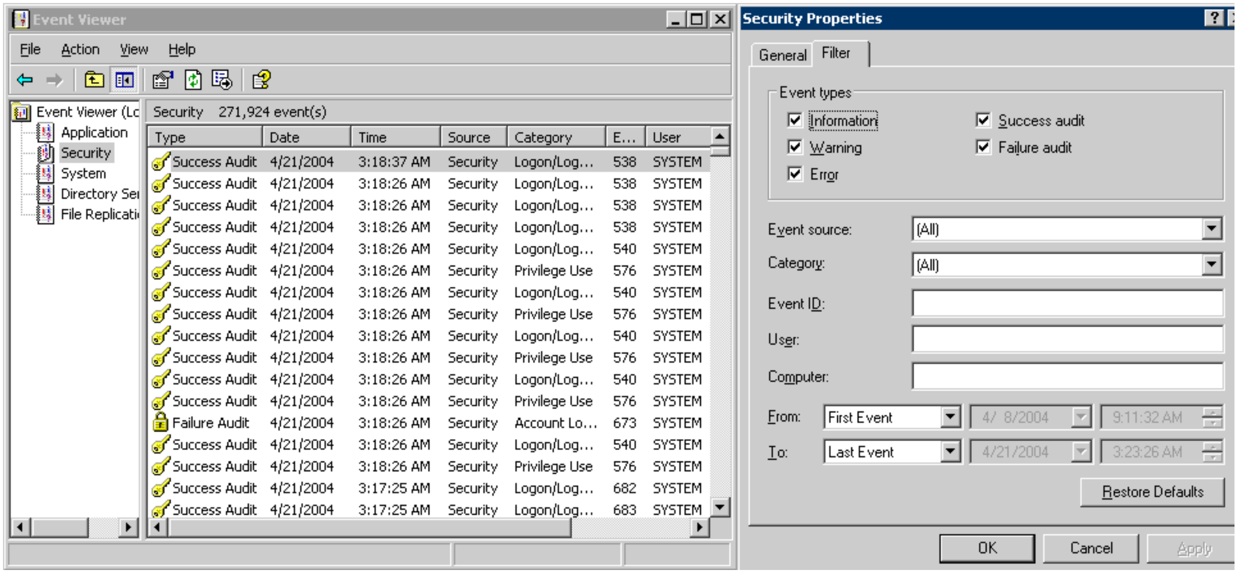
When you complete impersonation testing using the Windows 2003 Event Viewer, you must configure the event viewer before conducting the actual test.

To test impersonation through the Event Viewer

1. Select Start Menu; Event Viewer.
2. In the left pane, right-click Security, then click Properties.
3. Click the Filter tab on the Security property sheet.
4. Verify that all Event Types are checked, and the Event Source and Category lists are set to All, then click OK to dismiss the property sheet.

   Your Event Viewer is now configured to display information about the headerVar associated with a resource request.

Figure 20–4 Verifying Event Viewer Settings



5. Create a new IIS virtual server (virtual site).
6. Place a target Web page anywhere in the tree on the virtual site.
7. Point your browser at the Web page

   If impersonation is working correctly, the Event Viewer will report the success of the access attempt.

Testing Impersonation using a Web Page

You can also test impersonation using a dynamic test page, such as an .asp page or a perl script, that can return and display information about the request.

To test impersonation through a Web page that displays server variables

1. Create an .asp page or perl script that will display the parameters AUTH_USER and IMPERSONATE. It can resemble the sample page presented in the following listing:

Example 20–1 Sample .ASP Page Code

```
<TABLE border=1>
<TR>
<TD>Variable</TD>
<TD>  </TD>
<TD>Value</TD></TR>
<%for each servervar in request.servervariables%>
<TR>
<TD><%=servervar%></TD>
<TD>  </TD>
<TD><%=request.servervariables(servervar)%> </TD>
</TR>
```

2. Create an IIS virtual site, or use the one you created for the previous task.
3. Place an .asp page or perl script (such as the sample in the preceding listing) anywhere in the tree of the new virtual site.

4. Point your browser at the page. The page should display, with both AUTH_USER and IMPERSONATE set to the name of the user making the request.

Setting Up Impersonation with Integrations

The *Oracle Access Manager Integration Guide* provides a detailed example of how to integrate the Access System with the SharePoint Portal Server (SPPS) and the extra measures you may have to take to get impersonation running in different contexts:

- Configuring IIS Security
- Configuring the Wildcard Extension
- Editing web.config (this is not needed with the integration between the Access System and the Microsoft Content Management Server)

Enabling Impersonation with a User Name and Password

The method to enable impersonation before version 6.5 remains valid and may also be used with version 6.5 and later, as described in the following paragraphs.

The Access System provides an API that tells IIS which user to impersonate. To use this API, you must provide the user name and password to IIS. The user name is used in the proper header variables. This causes IIS to change the owner of the thread for downstream applications.

To have IIS log in as the user, you set the following two success actions in the authorization policy:

- **LOGON_USER**: The NT user name of the user you want to impersonate
- **LOGON_PASSWORD**: The NT password of the user.

The LOGON_PASSWORD is not set as a header variable. This prevents downstream applications from learning the password. This variable is only used to impersonate the user. The following are methods for providing the Windows NT or Active Directory (AD) password:

- Store the NT or AD password in clear text in the directory, then configure the Access System security policy to set the proper header variable with the password value.
- Store the password in a separate database. This requires an authorization plug-in to be written to access the password and set the appropriate header variable. The authorization plug-in supplies the action with the password. The store would have to be kept synchronized using the Identity System's PPP mechanisms.
- Create a static header variable that impersonates the user for a particular role (for instance, manager) that provides the proper security settings. This provides a more granular option if you do not require the actual individual to be impersonated.

The Access System supports additional IIS header variables for integration with Microsoft environments and Windows Impersonation, as shown in Table 20–3.

Table 20–3 Support for Additional IIS Header Variables

REMOTE_USER	AUTH_USER	AUTH_ PASSWORD	AUTH_TYPE

These are special case headers that show downstream applications that the user is logged in. If you set the REMOTE_USER header by creating a REMOTE_USER http header action, the Access System will set the AUTH_USER and AUTH_PASSWORD headers. You set REMOTE_USER in the same place as LOGON_USER and LOGON_Password, as a success action in the authorization policy. Setting this action accomplishes the following for each of the variables:

- The REMOTE_USER will contain the static or attribute value
- The AUTH_USER will have the same value
- The AUTH_PASSWORD header will always contain HiddenByOblixNetpoint so the password remains hidden
- The AUTH_TYPE header will contain Basic

For more information:

- See "Windows Impersonation Background" on page 20-18 for an introduction to access tokens, security IDs, access control lists, wildcard extensions, and Kerberos.
- See the Microsoft documentation for details about single sign-on integration through Windows Impersonation.

Setting Up Impersonation for OWA

In a distributed Exchange/OWA single sign-on environment, each server needs the Access System to impersonate the current user. When you enable Impersonation, you need to include additional HTTP Headers in "Authorization Success" for your impersonation policy domain:

The following solution has been tested in both standalone and distributed OWA environments.

Task overview: Setting up impersonation for OWA

1. Install Oracle Access Manager, including a WebGate on the OWA front-end server and on all Exchange back-end servers, as described in the *Oracle Access Manager Installation Guide*.
2. Disable IP Checking for the WebGates on the back-end server using the AccessGate (because the request comes from the front-end server, not from the user's browser).
3. Create a trusted user account for only impersonation in the Active Directory, as described in "Creating a Trusted User Account for OWA" on page 20-14.
4. Give the trusted user the special right to act as part of the operating system, as described in "Assigning Rights to the OWA Trusted User" on page 20-14.
5. Bind the trusted user to the WebGate by supplying the authentication credentials for the trusted user, as described in "Binding the Trusted OWA User to Your WebGate" on page 20-14.
6. Add a header variable named impersonate to Authorization Success Action in the policy domain for impersonation, as described in "Adding an Impersonation Action to a Policy Domain" on page 20-15.
7. Configure IIS by adding IISImpersonationExtension.dll to your IIS configuration, as described in "Adding an Impersonation dll to IIS" on page 20-16.
8. Test Impersonation, as described in "Testing Impersonation for OWA" on page 20-17.

Creating a Trusted User Account for OWA

This special user should not be used for anything other than impersonation.

Oracle recommends that you chose a very complex password, because your trusted user is being given very powerful permissions. Also, be sure to check the box marked Password Never Expires. Since the impersonation extension should be the only entity that ever sees the trusted user account, it would be very difficult for an outside agency to discover that the password has expired.

To create a trusted user account for OWA

1. On the Windows 2003 machine, select Start; Programs; Manage Your Server; Domain Controller (Active Directory); Manage Users and Computers in Active Directory.
2. In the Active Directory Users and Computers window, right-click Users on the tree in the left pane, then select New; User.
3. In the First name field of the pane entitled New Object - User, enter an easy-to-remember name such as OWAImpersonator.
4. Copy this same string to the User logon name field, then click Next.
5. In succeeding panels, you will be asked to choose a password and then retype it to confirm.
6. Proceed to "Assigning Rights to the OWA Trusted User" on page 20-14.

Assigning Rights to the OWA Trusted User

You need to give the trusted user the right to act as part of the operating system.

To give appropriate rights to the trusted user

1. Select Control Panel; Administrative Tools; Domain Controller Security Policy.
2. On the tree in the left pane, click the plus icon (+) next to Local Policies.
3. Click User Rights Assignment on the tree in the left pane.
4. Double-click "Act as part of the operating system" in the right pane.
5. Click Add User or Group.
6. In the Add User or Group panel, type the User logon name of the trusted user (OWAImpersonator in our example) in the User and group names text entry box, then click OK to register the change.
7. Proceed to "Binding the Trusted OWA User to Your WebGate" on page 20-14.

Binding the Trusted OWA User to Your WebGate

You need to bind the trusted user to the WebGate by supplying the authentication credentials for the trusted user, as described in the following procedure.

To bind your trusted OWA user to your WebGate

1. Point your browser to your Access System Console. For example:

   http://hostname.*domain*.com:*port*/access/oblix

   where hostname is the DNS name of the machine hosting your Policy Manager; domain is the name of the server domain to which the machine belongs; and port is the number of the port to which Policy Manager listens.

2. Navigate to Access System Console, Access System Configuration, AccessGate Configuration.
3. Select the name of the Webgate you want to modify.

   The Details for AccessGate page appears with a summary of the configuration information for this WebGate. At the bottom of this Web page are fields for Impersonation Username and Impersonation Password.
4. Click the Modify button at the bottom of the Details for AccessGate page.

   The Modify AccessGate page appears.
5. Scroll to the bottom and enter the user name and password for the trusted user account you created (OWAImpersonator).
6. Click the Save button to commit the changes and return to the Details page.

   A bind has been created for the WebGate and the trusted user. The WebGate is now ready to provide impersonation on demand. The demand is created by an Authorization Success Action in a policy domain created for impersonation.
7. Proceed to "Adding an Impersonation Action to a Policy Domain" on page 20-15.

Adding an Impersonation Action to a Policy Domain

You must create or configure a policy domain to protect your OWA resources. This policy must set several HTTP Header variables.

> **Note:** You should choose an easy-to-remember name for the domain, such as ImpersonationPolicyDomain.

To add an impersonation action to your policy domain

1. Navigate to the Access System Console and log in. For example:

   ```
   http://hostname.domain.com:port/access/oblix
   ```

   where hostname is the DNS name of the machine hosting your WebPass and Policy Manager; domain is the name of the server domain to which the machine belongs; and port is the number of the port to which Policy Manager listens.
2. Navigate to the Authorization Definitions page of the policy domain you want to change:

   Policy Manager; My Policy Domains; PolicyName; Authorization Rules

   where PolicyName refers to the policy domain you created specifically for impersonation (ImpersonationPolicyDomain in this example).
3. Currently defined authorization rules are listed. If none are listed, click the Add button and complete the form to create one.
4. Click the link to the rule to which you want to add the impersonation action to expand the description.
5. Click the Actions tab, directly under the Authorization Rules tab.

   The Authorization Success page appears, with a separate section for Authorization Success and Authorization Failure. If no actions are identified, you must add them. If actions are provided, you can modify them.

You need to add header variables named "impersonate", "auth_type", "remote_user", and "npusername" to the Authorization Success Action in the policy domain for impersonation.

6. On the Authorization Success page, click the Add or Modify button.
7. In the Authorization Success area, fill in the Return details.

   Type: HeaderVar
   Name: IMPERSONATE
   Return value: uid (or samaccountname)

   Type: HeaderVar
   Name: AUTH_TYPE
   Return value: NTLM

   Type: HeaderVar
   Name: REMOTE_USER
   Return value: uid (or samaccountname)

   Type: HeaderVar
   Name: NPUSERNAME
   Return value: uid (or samaccountname)
8. Save the rule, which is used for the second WebGate request for authorization.
9. Proceed with an "Adding an Impersonation dll to IIS" on page 20-16.

Adding an Impersonation dll to IIS

You are ready to configure IIS by adding the IISImpersonationExtension.dll to your IIS configuration.

To add the impersonation dll to your IIS configuration

1. Select Start; Administrative Tools; Internet Information Services (IIS) Manager.
2. Click the plus icon (+) to the left of the local computer icon on the tree in the left pane.
3. Click Web Service Extensions on the tree in the left pane.
4. Double-click WebGate in the right panel to open the Properties panel.
5. Click the Required Files tab.
6. Click Add.
7. In the Path to file text box, type the full path to IISImpersonationExtension.dll.

   An example:

   ```
   WebGate_install_dir\access\oblix\apps\webgate\bin\IISImpersonation\
   Extension.dll
   ```

   where *WebGate_install_dir* is the directory of your WebGate installation.

   Note: If any spaces exist in the path (for example, C:\Program Files\Oracle\...) surround the entire string with double quotes (" ")
8. Click OK.

9. Add the IISImpersonationExtension.dll in Wildcard Application Maps for the Exchange virtual directory, as follows.

   From the Internet Services Manager, go to WebSite/Virtual Directory Properties, then to the Home Directory, then to Configuration, then to Mappings, and add IISImpersonationExtension.dll in Wildcard Applications maps.

   IISImpersonationExtension.dll should be the first entry in the wildcard application maps order.

   If you add IISImpersonationExtension.dll to the WebSite level, the wildcard application maps for the Exchange virtual directory is overridden and the existing OWA-related extensions are removed at the Exchange virtual directory level. This causes OWA to fail. To retrieve the OWA settings, see the Microsoft knowledge base article at the following URL:

   `http://support.microsoft.com/kb/298513`.

10. Verify that the Allow button to the left of the WebGate icon is greyed out, which indicates that the dll is allowed to run as a Web service extension.

    Note: If Allow is not greyed out, click it so that it becomes greyed out.

11. Proceed to "Testing Impersonation for OWA" on page 20-17.

Testing Impersonation for OWA

The following options are provided to test the Impersonation configuration for OWA.

- Testing Impersonation Using the Event Viewer
- Testing Impersonation using a Web Page

Testing Impersonation Using the Event Viewer

To test impersonation through the Event Viewer

1. Select Start Menu; Event Viewer.
2. In the left pane, right-click Security, then click Properties.
3. Click the Filter tab on the Security property sheet.
4. Verify that all Event Types are checked, and the Event Source and Category lists are set to All, then click OK to dismiss the property sheet.
5. Your Event Viewer is now configured to display information about the headerVar associated with a resource request.
6. Create a new IIS virtual server (virtual site).
7. Place a target Web page anywhere in the tree on the virtual site.
8. Point your browser at the Web page.

   If impersonation is working correctly, the Event Viewer will report the success of the access attempt.

Testing Impersonation using a Web Page

You can also test impersonation using a dynamic test page, such as an .asp that can return and display information about the request.

To test impersonation through a Web page

1. Create an .asp page or perl script that will display the parameters AUTH_USER and IMPERSONATE, which can resemble the sample page presented in the following listing:

```
<TABLE border=1>
<TR>
<TD>Variable</TD>
<TD>  </TD>
<TD>Value</TD></TR>
<%for each servervar in request.servervariables%>
<TR>
<TD><%=servervar%></TD>
<TD>  </TD>
<TD><%=request.servervariables(servervar)%> </TD>
</TR>
```

2. Create an IIS virtual site, or use the one you created for the previous task.
3. Place an .asp page or perl script (such as the sample in the preceding listing) anywhere in the tree of the new virtual site.
4. Point your browser at the page, which should appear, with both AUTH_USER and IMPERSONATE set to the name of the user making the request.

Windows Impersonation Background

The information here provides a simple overview of several Windows impersonation concepts. Topics include:

- Access Tokens
- Security IDs
- Access Control Lists and Entries
- Wildcard Extension
- The Kerberos Protocol
- The S4U2Self Extension

For more information, see your Microsoft documentation.

Access Tokens

The access token describes the security context of a process or thread and includes the identity and privileges of the user account associated with the process or thread. The access token is created when authentication is successful. For example:

- The logon process returns a security ID (SID) for the user and a list of SIDs for the user's security groups.
- The Local Security Authority (LSA) creates an access token that includes:
  - The SIDs returned by the logon process

  - A list of privileges assigned to the user and to the user's security groups by local security policy

A copy of the access token is attached to every process and thread that is executed on the user's behalf. When a thread interacts with a securable object, or tries to perform a system task that requires privileges, the operating system checks the access token associated with the thread to determine its level of authorization.

Security IDs

A security ID (SID) is a unique value of variable length used to identify a security principal or security group. SIDs are equal to Access System single sign-on tokens and represent a unique user within the Windows operating system.

The SID that identifies a particular account or group is generated by the system at the time the account or group is created. As mentioned previously, the SID for a local account or group is generated by the Local Security Authority (LSA) and stored with other account information in a secure area of the registry. The SID for a domain account or group is generated by the domain security authority and stored as an attribute of the User or Group object in Active Directory.

SIDs are unique within the scope of the account or group they identify. The SID for every local account and group is unique on the computer on which it was created. No two accounts or groups on the same machine can have the same SID. The SID for every domain account and group is unique within an enterprise. The SID for an account or group created in a domain never matches the SID for any other account or group created in the same domain.

One or more SIDs are included:

- In access tokens, where one SID identifies the user represented by the token and additional SIDs identify the security groups to which the user belongs.
- In security descriptors, where one SID in an object's security descriptor identifies the object's owner and another SID identifies the owner's primary group.
- In access control entries (ACEs), the SID identifies the user or group for whom access is allowed, denied, or audited.

Access Control Lists and Entries

An access control list (ACL) contains an ordered list of access control entries (ACEs) that define the policies used to control access to resources, such as directories and applications protected by the Access System.

All ACLs are based on your logon identity. An object's security descriptor can contain two ACLs:

- A discretionary access control list (DACL) that identifies the users and groups who are allowed or denied access
- A system access control list (SACL) that controls how access is audited

Each ACE includes:

- The type of the ACE (generic vs. object specific)
- Child-object inheritance attributes
- Access rights
- A SID that identifies a user or group

Wildcard Extension

The Web server normally runs in a security context called "IWAM_xxx" This security context does not have rights to impersonate another user. The Access System designates a special user that does have the right to impersonate another user by configuring it using the impersonation username/password on the AccessGate configuration page. That designated user must have "act as operating system" rights, as explained elsewhere.

The wildcard extension for the impersonation DLL behaves like a filter, which means that the wildcard extension is enabled for each request to the Web server. The DLL executes after WebGate, after all filters, and before any downstream applications.

The Kerberos Protocol

The Kerberos protocol defines how clients interact with a network authentication service. The client obtains a ticket from the Kerberos Key Distribution Center (KDC). The Kerberos ticket represents the client's network credentials. The ticket is presented to a server when the connection is established.

The Kerberos protocol handles all domain lookups in all trusted domains. As the client's identity, this protocol uses:

- The Active Directory domain name
- User name
- Password

The initial ticket that is obtained from the KDC when the user logs on is based on an encrypted hash of the user's password. This initial ticket is cached.

When the user tries to connect to a server, the Kerberos protocol checks the ticket cache for a valid ticket for that server. If one is not available, the initial ticket for the user is sent to the KDC along with a request for a ticket for the specified server. That session ticket is added to the cache and can be used to connect to the same server until the ticket expires.

The S4U2Self Extension

Windows Server 2003 domain controllers accept a new type of Kerberos request, the Service-for-User-to-Self (S4U2Self) extension. This extension enables the service to request a ticket from the client to itself, presenting its own credentials instead of the client's.

Negative Testing for Impersonation

To conduct negative testing for impersonation, you need to unbind the trusted user from the WebGate, as explained in the following procedure.

To unbind the trusted user from your WebGate

1. Log in to the Access System Console at a URL similar to the following:

   `http://`*`hostname.domain`*`.com:`*`port`*`/access/oblix`

   Where *hostname* is the DNS name of the computer hosting your Access Manager, *domain* is the name of the server domain to which the machine belongs, and *port* is the number of the port to which Access Manager listens.

2. Select Access System Console, then click Access System Configuration, then click AccessGate Configuration.
3. Select the name of the Webgate that you want to modify.

   The Details for AccessGate page appears with a summary of the configuration information for this WebGate. At the bottom of this Web page are fields for Impersonation Username and Impersonation Password.
4. Click the Modify button at the bottom of the Details for AccessGate page.

   The Modify AccessGate page appears.
5. Remove the credentials for the trusted user.
6. Click Save.

   You return to the Details page.
7. Restart the IIS server.
8. Point your browser at a protected code page that previously was accessible to the trusted user.

   An error message page should appear. Values for AUTH_USER and IMPERSONATE are necessary for impersonation credentials to be bound to a WebGate.

21

Integrating With the Content Management Server

The Microsoft Content Management Server (MCMS) is an enterprise Web content management system for authoring and delivery. This chapter explains how to integrate with the Microsoft Content Management Server (MCMS) 2002.

This chapter discusses the following topics:

About Oracle Access Manager and the MCMS

Oracle Access Manager provides a full range of identity management and security functions, including: Web-based single sign-on (SSO), user self-service and self-registration, user provisioning, reporting and auditing, policy management, dynamic groups, and delegated administration. Oracle Access Manager integrates with all leading directory servers, application servers, Web servers, and enterprise applications.

The Microsoft Content Management Server (MCMS) is an enterprise Web content management system for authoring and delivery. The MCMS streamlines the Web publishing process, enables you to build, deploy, and maintain content-rich Web sites, and enables users to manage their own content. The role-based distributed publishing model of the MCMS includes a multi-level approval workflow, automatic content scheduling and archiving, and content indexing. Developers can create Content Management Server–based applications using ASP.NET and the Microsoft .NET Framework.

The MCMS provides its own authentication mechanisms that leverage IIS and may require an additional login. After integrating Oracle Access Manager with MCMS, the Access System handles authentication and single-sign on with the site created using MCMS. Access System-authenticated users enjoy single sign-on access to MCMS resources and to Access System-protected resources.

The integration with MCMS requires authentication schemes based on Windows Impersonation. In addition, Oracle Access Manager supports URL-level authorization. MCMS performs application-level authorization based on the roles you set up in the MCMS.

The MCMS is often used with the Microsoft SharePoint Portal Server (SPPS) for developing and managing Web content. The Microsoft Content Management Server 2002 Connector for SharePoint Technologies enables you integrate the Content Management Server with the Microsoft Office® SharePoint Portal Server. The connector enables sharing of key publishing and search technologies. For details about integrating with the SharePoint Portal Server, see "Integrating with SharePoint Portal Server 2003" on page 17-5.

About Windows Impersonation

The MCMS integration relies on the Windows impersonation feature, which enables a trusted user in the Windows server domain to assume the identity of any user requesting an MCMS target resource. This trusted impersonator maintains the identity context of the user while accessing the resource on behalf of the user.

Impersonation is transparent to the user; access appears to take place directly, as if the MCMS resource were a resource within the Access System domain. For more information, see "Setting Up Impersonation" on page 21-6.

Support and Requirements

Any references to specific versions and platforms in this chapter are made for demonstration purposes.

Successful integration with MCMS requires both Oracle Access Manager and Microsoft components, which must be installed and configured to support impersonation as well as integration. The following topics provide requirements:

- Supported Versions and Platforms
- Required Oracle Access Manager Components
- Required Microsoft Components

Supported Versions and Platforms

Any references to specific versions and platforms in this chapter are for demonstration purposes.

To see the supported versions and platforms for this integration, refer to Metalink as follows.

To view information on Metalink

1. Go to the following URL:

   `http://metalink.oracle.com`

2. Click the Certify tab.
3. Click View Certifications by Product.
4. Select the Application Server option and click Submit.
5. Choose Oracle Application Server and click Submit.

Required Oracle Access Manager Components

The following components are required to integrate with MCMS. With the exception of a WebGate, all components may reside on different machines or the same machine as the MCMS.

- Identity Server
- WebPass
- Policy Manager
- Access Server
- WebGate installed with the MCMS on a Windows Server 2003

The ISAPI WebGate includes the IISImpersonationExtension.dll, which you need to configure manually to enable impersonation for the MCMS integration.

The Oracle Access Manager IISImpersonationExtension.dll is an IIS wildcard extension that checks whether the Authorization Success Action headerVar has been set to impersonate. If it has been, the dll creates a Kerberos U4S2Self ticket so that the special trusted user in the MCMS Active Directory can impersonate the user who originally made the request.

Required Microsoft Components

Any references to specific versions and platforms in this chapter are made for demonstration purposes. For the latest support information, see the Certify tab at https://metalink.oracle.com.

Oracle Access Manager supports the Microsoft Content Management Server with:

- Windows Server
- Microsoft IIS Web Server
- Active Directory (the domain controller must be on a Windows 2003 Server)
- MSSQL supported by the MCMS
- **Optional**: Microsoft SharePoint Portal Services

Request Processing by the Integration

Oracle Access Manager uses the Windows impersonation feature to facilitate user access to MCMS resources.

Process overview: Request processing with MCMS integration

1. The user requests access to an MCMS resource.
2. The WebGate protecting MCMS intercepts the request, determines whether the target resource is protected, and if it is, challenges the user for authentication credentials.
3. The user supplies credentials and the Access Server validates them.
4. Upon validation, the WebGate sets an ObSSOCookie in the user's browser, thus enabling single sign-on.

   The WebGate also sets an HTTP header variable called impersonate, whose value is set to the authenticated user's LDAP uid (*samaccountname*, if the user account exists in Active Directory, or *userPrincipalName*, if the user account exists in a multi-domain Active Directory forest).

> **Note:** At this point, IIS considers the user to be anonymous, since the impersonation has not yet been set.

5. The Oracle Access Manager ISAPI wildcard extension IISImpersonationExtension.dll checks for the Authorization Success Action header variable named impersonate.
6. When such a header variable exists, the wildcard extension obtains a Kerberos ticket for the user.

   This Service for User to Self (S4U2Self) impersonation token enables the designated trusted user to assume the identity of the requesting user and obtain access to the target resource through IIS and MCMS.
7. Authorization is performed by the MCMS based on the roles setup in the MCMS.
8. When authorization is successful, the user is granted access to the resource.

Integrating with the MCMS

You need to complete several procedures to integrate with the Content Management Server.

Note: The procedures in this chapter illustrate how to integrate with the MCMS using a sample Web site (the Microsoft WoodgroveNet Web site). References to specific versions and platforms are for demonstration purposes. See "Support and Requirements" on page 21-2.

Task overview: Integrating with MCMS

1. Install Oracle Access Manager, as described in "Installing Oracle Access Manager" on page 21-4.
2. Install the Microsoft components, as described in "Installing Microsoft Components" on page 21-5.
3. Integrate with the MCMS, as described in "Integrating with the MCMS" on page 21-5.
4. Configure Impersonation, as described in "Setting Up Impersonation" on page 21-6.
5. Finish the MCMS integration, as described in "Completing the MCMS Integration" on page 21-7.
6. Test the integration, as described in "Testing the MCMS Integration" on page 21-7.

Installing Oracle Access Manager

The ISAPI Webgate for MCMS must be installed on the machine that hosts the MCMS. All other Oracle Access Manager components can reside together on the machine hosting the MCMS or on any other machine.

If both Oracle Access Manager and MCMS are set up for different instances of Active Directory, both instances must belong to the same Active Directory domain.

To install Oracle Access Manager for the integration

1. Install an Identity Server and a WebPass, then set up the Identity System, as described in the *Oracle Access Manager Installation Guide*.

2. Install and set up the Policy Manager and one or more instances of the Access Server, as described in the *Oracle Access Manager Installation Guide*.
3. Install WebGates, as described in the *Oracle Access Manager Installation Guide*.

> **Note:** One WebGate must be installed on the machine hosting the MCMS, as described in "Integrating with the MCMS" on page 21-5.

Installing Microsoft Components

Except where noted, all MCMS components from Microsoft must be installed on the same host machine.

To install Microsoft components

1. On a machine Windows Server 2003 with IIS v6.0, complete the following activities to install the MCMS using instructions in your Microsoft documentation:
   a. Create Windows user accounts.
   b. Create a database in MSSQL and grant rights to the system administrator account.
   c. Create two Web sites.
   d. Install the MCMS 2002 SP1a.
   e. Configure the database with the MCMS 2002 Database Configuration Application (DCA).
   f. Configure the MCMS server using the Server Configuration Application (SCA).
   g. Update the maximum upload size settings in the web.config file.
   h. Install Site Manager.
2. On a Windows 2003 Server host, install Active Directory for the MCMS using instructions in your Active Directory documentation.
3. Ensure your MCMS installation is working properly using instructions in your Microsoft documentation as you:
   a. Download a sample WoodGroveNet Web site and install it on the MCMS site to use as a test vehicle for the procedures in this chapter.
   b. Ensure you can log in to the Site Manager, Server Configuration Application, and the sample WoodGroveNet Web site.
4. **Optional**: Install the Microsoft Content Management Server 2002 Connector for SharePoint Technologies, as described in your Microsoft documentation

   For details about integrating with the SPPS, see "About Oracle Access Manager and the SharePoint Server" on page 17-1.

Integrating with the MCMS

After installing Oracle Access Manager and the MCMS, as described earlier, you need to complete the following steps to integrate the two environments.

To integrate with the MCMS

1. On the Windows 2003 Server machine hosting the MCMS, install an ISAPI WebGate using instructions in the *Oracle Access Manager Installation Guide.*

   The IISImpersonationExtension.dll is installed automatically in:

   WebGate_install_dir\access\Oblix\apps\webgate\bin\

   Where *WebGate_install_dir* is the directory where you installed the WebGate.

2. Install a WebGate on the MCMS site, and impersonation.dll in the WoodGroveNet site.

   See "Setting Up Impersonation" on page 21-6 for details.

3. If MCMS is not installed on default site, do the following:
   - On the MCMS site, right-click, select New, then select Virtual Directory.
   - Set the virtual directory alias to "access" and click Next.
   - Enter the following path:

     WebGate_install_dir\access
   - Click Next.
   - Set the permissions, then click Next.
   - Click Finish.

Setting Up Impersonation

The integration with the MCMS requires Windows impersonation.

Note: Use the procedures in "Enabling Impersonation with the Access System" on page 20-1 and in "Setting Up Impersonation" on page 17-13 to implement impersonation in your environment. Details are not repeated in this chapter.

Task overview: Setting up impersonation for the MCMS

1. Create a trusted user account for only impersonation in the Active Directory connected to MCMS, as described in "Creating a Trusted User Accounts" on page 17-14
2. Give the trusted user the special right to act as part of the operating system, as described in "Assigning Rights to the Trusted User" on page 17-14.
3. Bind the trusted user to the WebGate by supplying the authentication credentials for the trusted user, as described in "Binding the Trusted User to Your WebGate" on page 17-15.
4. Add a header variable named impersonate to Authorization Success Action in the policy domain for impersonation, as described in "Adding an Impersonation Action to a Policy Domain" on page 17-16.
5. Configure IIS by adding IISImpersonationExtension.dll to your IIS configuration, as described in "Adding an Impersonation dll to IIS" on page 17-17.
6. Test impersonation, as described in "Testing Impersonation" on page 17-19.
7. Proceed as described in "Completing the MCMS Integration" on page 21-7.

Completing the MCMS Integration

After confirming that impersonation is set up properly, you need to perform the following steps to complete the integration, ensure that everything is working properly, and confirm that you have single-sign on access.

To complete the MCMS integration

1. Move the two ISAPI filters installed at the IIS Top Level Web site by MCMS to the two virtual Web sites created for MCMS, as follows:

   MCMS HTML Packager Filter

   C:\Program Files\Microsoft Content Management Server\Server\bin\REHTMLPackager.dll

   MCMS ISAPI Filter

   C:\Program Files\Microsoft Content Management Server\Server\bin\REAuthFilt.dll

2. Complete following steps to finish the impersonation implementation for the MCMS integration:

   a. Perform the steps in "Configuring IIS Security" on page 17-22 to set up the environment for the impersonation implementation.

   b. Perform the steps in "Configuring the Wildcard Extension" on page 17-23 for each MCMS virtual server for which you wish to enable integration.

3. Give appropriate rights to users for viewing different sections of the Web site using the Site Manager.

4. Using the Policy Manager, create Policies to protect the WoodGroveNet top-level resource.

Testing the MCMS Integration

After you complete the tasks to enable integration, it is a good idea to test the integration to verify things are working as expected.

This section contains the following topics:

Testing NetPoint/MCMS Integration

It is important to verify that a user can access MCMS resources through Access System authentication and MCMS authorization.

To test your MCMS integration using the WoodGroveNet application

1. Open the file Web.config in an editor.

   The path to this file is as follows:

   MCMS_InstallDir\Sample Data\WoodgroveNet\

2. In this file, change the authentication mode from `forms` to `Windows`.

3. Save the file Web.config and restart IIS.

4. Navigate to a WoodGroveNet Web site using your browser.

The Access System challenges you for credentials.

5. Log in by supplying the necessary credentials.
6. Confirm that you have access.
7. **Optional**: Check the Event Viewer to confirm that the access request was successful.

Testing single sign-on for the MCMS Integration

You test single sign-on by demonstrating that a user who has just supplied credentials and accessed an MCMS resource can (before the ObSSOCookie expires) access a non-MCMS resource without having to supply credentials a second time. For this test you can use an Access System-defined resource.

When single sign-on is working, you should be granted access to the page without having to supply credentials a second time.

To test single sign-on for your MCMS integration

1. Create a new resource and protect it with a policy domain (or use one you have already created.
2. Using a browser, navigate to the resource.

   If you have already passed authentication, you should be granted access to the page without having to supply credentials a second time.

Part III

Appendices

This part providessupplementary information on configuring third-party integrations.

Part III contains the following appendices:

- Appendix A, "Configuring Logout"

A

Configuring Logout

This appendix explains how to configure logout so that users can be logged out of all applications that they have accessed during a single sign-on session, including third-party applications that are integrated with Oracle Access Manager.

This appendix discusses the following topics:

About Oracle Access Manager Logout

If you use form-based authentication, you can automatically log users out of one or more applications by configuring a logout URL that removes session cookies and redirects users to a logout page. You can customize the default logout page, for example, to add a meta tag to redirect to another page after a few seconds.

Note that you must configure a logout link and URL for the Identity System applications and the Policy Manager as well as for any other protected resource. See the *Oracle Access Manager Access Administration Guide* for details.

The following methods are available for configuring logout:

- **Provide one Oracle Access Manager-provided logout function:** You can configure a single sign-on logout URL and logout page that removes the user's session cookies.

  See the *Oracle Access Manager Access Administration Guide* for details.
- Multiple **logout functions:** You can configure different logout URLs and pages for different purposes based on the Oracle Access Manager-provided default.
- **Third-party program for logging out users:** You can define your own logout functionality.

Note: If you have multi-domain single sign-on configured, note that the logout URL only logs users out from applications in one domain. To ensure that logout occurs across domains, you may need to consider setting an absolute session timeout value.

How Logout Works

The WebGate logs a user out when it receives a URL containing "logout." (including the "."), with the exceptions of logout.gif and logout.jpg, for example, logout.html or logout.pl. When the WebGate receives a URL with this string, the value of the ObSSOCookie is set to "logout."

The Access System sets an obSSOCookie for each user or application that accesses a resource protected by a WebGate. The obSSOCookie enables users to access resources that are protected by the Access System that have the same or a lower authentication level. Removing the ObSSOcookie causes the WebGate to log the user out and requires the user to re-authenticate the next time he or she requests a resource that is protected by the Access System.

Oracle provides a logout.html page. This form is located in:

PolicyManager_install_dir/access/oblix/lang/en-us/logout.html

The logout.html form also contains javascript for removing the ObTemC cookie set for the Identity System. However, this page does not by default contain the code to remove the ObSSOCookie. Calling the single sign-on logout URL usually, but does not always remove the ObSSOCookie, so you should manually add this code to logout.html.

The logout.html form also does not remove any cookies set by third-party applications. To ensure that users must re-authenticate, you may need to customize the single sign-on logout.html file to remove these cookies.

You can customize this page or create one or more new custom logout pages.

Configuring and Customizing the Logout URL and Page

You can configure one single sign-on logout URL and page that apply to all users and resources. Or, you can create different logout functions for different applications.

Task overview: Configuring and customizing logout

1. Modify the default logout.html or create a new logout page.

   Include the string "logout." (including the ".") in the file name, with the exceptions of logout.gif and logout.jpg, for example, logout.html or logout.pl.

   This page must contain Javascript code to remove session cookies and an onLoad event to run the code in the body tag, for example:

   ```
   <body onLoad="delOblixCookie";>
   ```

2. Place the page in the same relative path on all appropriate Web servers.

   For example, if the SSO Logout URL is /public/logout/logout.html, this file must be known to the Web server that contains any page with the logout link.

3. Protect the logout page with a policy that uses an Anonymous authentication scheme to ensure that anyone can access it.

   This is true for the SSO Logout URL and custom URLs. For example, if your SSO Logout URL is /public/logout/logout.html, ensure that this resource is protected at /public, /public/logout or '/public/logout/logout.html.

4. Ensure that the logout URL is recognized by Oracle Access Manager.

If you configured multiple logout pages, add them to the logoutURLs parameter for the WebGate. See the information on AccessGate configuration in the *Oracle Access Manager Access Administration Guide* for details.

5. Configure the SSO Logout URL.

   See the information on configuring a single sign-on logout URL in the *Oracle Access Manager Access Administration Guide* for details. You should also add the SSO Logout URL to the list of URLs in the logoutURLs parameter.

6. Add a link with the appropriate logout URL on all Web pages where this URL is needed.

Configuring Single Sign-Off for an Integration Between Oracle Access Manager and Another Product

For third-party products that enable you to configure customized logout URLs, for example, WebSphere and SAP, the third party-product deletes its application-specific cookies, then it redirects the logout page to the Oracle Access Manager logout.html. When the WebGate finds the HTTP request for logout.html, it deletes the ObSSOCookie. For this type of logout, you only need to customize logout URL for the third-party application. You do not need to specify logout URLs in Oracle Access Manager.

However, when you configure single sign-on between Oracle Access Manager and another product, logging out of the third-party product may not automatically end an Oracle Access Manager session. For example, if you configure single sign-on between Oracle Access Manager and Oracle's Siebel product, when you log out of Siebel, you are not necessarily also logged out of Oracle Access Manager.

As described in the previous sections of this appendix, you can configure single sign-off for these scenarios. For single sign-off to work, you must ensure that, minimally, the ObTEMC and ObSSOCookie are deleted.

Oracle Access Manager provides a default logout.html file, as follows:

PolicyManager_install_dir/access/oblix/lang/en-us/logout.html

If you want to modify this file to log the user out of all application sessions that they started during the single sign-on session, you must include a Javascript function to delete all cookies that Oracle Access Manager and the other applications use. For Oracle Access Manager, you must delete the following cookies when the logout page loads:

- ObTEMC
- ObSSOCookie

For other applications, you would delete the login cookies that they set. For example, if you want to also log the user out of MyApp, and this application sets MYAPP_COOKIE, you would also delete the following cookie:

- MYAPP_COOKIE

You may also want to delete cookies that are associated with various servers that are involved in the single sign-on session. The following are examples:

- OHS-idm.demo.mycompany.com-7777
- OHS-idm.demo.mycompany.com-7778

Example A–1 illustrates a logout.html page that contains a Javascript function named `delCookie`. This function is called when the logout page is loaded in the user's browser. It deletes all Oracle Access Manager-related cookies.

Example A–1 also performs single sign-off for an application by deleting a cookie named myCustomApp that is set by an application called myCustomApp. The example assumes that the cookie contains login data that is required by myCustomApp. If the cookie exists, the application believes the user is still logged in. In the example, the line in bold would be added to delete the myCustomApp cookie. This ensures a clean logout when the logout page is loaded in the user's browser because all cookies related to the applications are deleted.

If you add a similar Javascript function to the default logout.html page, ensure that this function deletes any relevant cookies. These are cookies that that control the session state of the application. Note that for applications that do not control session state using cookies, you must configure single sign-off using a method appropriate for that application.

Example A–1 Example of Single Sign-Off by Deleting a Cookie Named myCustomApp

```
<html>
<head><link rel="stylesheet" type="text/css" href="style2/coreid.css"></link>
<meta http-equiv="Content-Type" content="text/html; ">
<meta name="Description" content="Oracle Access Manager">
<meta name="Robot" content="none">
<meta name="Copyright" content="Copyright &copy; 1996-2006, Oracle. All Rights
Reserved.">
<style type="text/css">
<!--
.unnamed1 {  font-family: Arial, Helvetica, sans-serif; font-size: 2pt}
-->
</style>
<title>Oracle Access Manager</title>
<script language="JavaScript">
        function delCookie(name,path,domain) {
        var today = new Date();
        var deleteDate = new Date(today.getTime() - 48 * 60 * 60 * 1000); // minus
2 days
        var cookie = name + "="
                    + ((path == null) ? "" : "; path=" + path)
                    + ((domain == null) ? "" : "; domain=" + domain)
                    + "; expires=" + deleteDate;
                document.cookie = cookie;
}

         function delOblixCookie() {
            // set focus to ok button
            var isNetscape = (document.layers);
            if (isNetscape == false || navigator.appVersion.charAt(0) >= 5) {
            for (var i=0; i<document.links.length; i++) {
                  if (document.links[i].href == "javascript:top.close()") {
                  document.links[i].focus();
                  break;
                           }
                  }
             }
             delCookie('ObTEMC', '/');
             delCookie('ObSSOCookie', '/');

             // Added myCustomAppCookie deletion
```

```
            delCookie('myCustomApp', '/');

            // in case cookieDomain is configured
            // delete same cookie from all subdomains
                  var subdomain;
                  var domain = new String(document.domain);
                  var index = domain.indexOf(".");
                  while (index > 0) {
                        subdomain = domain.substring(index, domain.length);
                        if (subdomain.indexOf(".", 1) > 0) {
                              delCookie('ObTEMC', '/', subdomain);
                              delCookie('ObSSOCookie', '/', subdomain);
                        }
                        domain = subdomain;
                        index = domain.indexOf(".", 1);
                   }
                }
      </script>
</head>
<body bgcolor="#ffffff" marginwidth="0" marginheight="0" topmargin="0"
leftmargin="0" onload="delOblixCookie();">

<table width="100%" border="0" cellspacing="1" cellpadding="0">
<tr>
<td rowspan="2" width="10%" bgcolor="#FFFFFF" align="center" valign="middle"> <img
src="style2/LOGINoblixlogo.gif"/></td>
</tr>
<tr>
<td bgcolor="#0099CC" align="center" valign="middle"><img
src="style2/LOGINaccesssystem.gif"/><br/></td>
<td bgcolor="#99CCCC" align="center" valign="middle"><img
src="style2/LOGINversion.gif"/></td>
</tr>
<tr>
<td> </td>
<td align="right" valign="top">
<table border="0" cellspacing="0" cellpadding="0">
<tr align="right" valign="middle">
<td>
<a href="http://www.oracle.com"><font class="basictextfonts3" size="2"
color="#003366"><b>Oracle Website</b></font></a>
|
<a href="http://www.oracle.com/support/contact.html">
<font class="basictextfonts3" size="2" color="#003366"><b>Online
Support</b></font></a>
</td>
</tr>
</table>
</td>
<td> </td>
</tr>
<tr>
<td> </td>
<td align="center">
<br/>
<h3>Oracle Access Manager Applications</h3>
<h3>You have been logged out.</h3>
<h3>For security reasons, please close the browser window.</h3></font><a
href="javascript:top.close()" onMouseOver="self.status='Close the browser
window.'; return true"><img border="0" alt="Close the browser window."
```

```
src="style2/NAVok.gif"></a></center>
</td>
<td> </td>
</tr>
<tr>
<td> </td>
<td>
<hr/>
<font class="basictextfonts3" size="1">
Copyright © 1996-2006,Oracle. All rights reserved. US Patent Numbers 6,539,379;
6,675,261; 6,782,379; 6,816,871. Portions copyright © 1991-2003, Compuware
Corporation. Includes RSA BSAFE® cryptographic or security protocol software from
RSA Security. Copyright © 2003, RSA Security Inc. All rights reserved. Oracle is a
registered trademark of Oracle Corporation and/or its affiliates.  Other names may
be trademarks of their respective owners.
</font>
</td>
<td> </td>
</tr>
</table>
</body>
</html>
```

Index

A

C

D

E

F

H

I

K

L

M

N

O

P

R

S

T

U

V

W