Oracle® Access Manager

Access Administration Guide

10*g* (10.1.4.3)

E12488-01

May 2009

This document describes how to set up and administer the Access System and how to protect resources by defining policy domains, authentication schemes, and authorization schemes. This book also describes configuring single- and multi-domain single sign-on and designing custom login forms.

ORACLE

Oracle Access Manager Access Administration Guide 10*g* (10.1.4.3)

E12488-01

Primary Author: Gail Flanegin

Contributing Author: Nina Wishbow

Contents

3 Configuring WebGates and Access Servers

5 Configuring User Authentication

6 Configuring User Authorization

7 Configuring Single Sign-On

Part III Managing the Access System

8 Access System Management

9 Managing Access System Configuration Files

Part IV Appendices

A Form-Based Authentication

E Troubleshooting Oracle Access Manager

Preface

The *Oracle Access Manager Access Administration Guide* provides information on setting up Access Administrators and Access Server settings, configuring and managing AccessGates, defining access controls and user access to applications and data, and configuring single sign-on. Its companion guide, the *Oracle Access Manager Identity and Common Administration Guide* describes the Identity System and functions that are common to both systems, such as configuring directory servers, password policies, setting up logging and auditing, and so on.

> **Note:** Oracle Access Manager was previously known as Oblix NetPoint. All legacy references to Oblix and NetPoint, for example, in path names and schema objects, should be understood to refer to Oracle and COREid, respectively.

This Preface covers the following topics:

Audience

This guide is intended for the Master Administrators who are assigned during installation and setup, as well as Master Access Administrators and Delegated Access Administrators. Administrators configure the rights and tasks available to other administrators and end users. A Master Administrator, the highest level administrator, is selected during Oracle Access Manager System setup. The Master Administrator delegates responsibilities to other administrators, as described in this book.

This document assumes that you are familiar with your LDAP directory and Web servers.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible to all users, including users that are disabled. 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.

Deaf/Hard of Hearing Access to Oracle Support Services

To reach Oracle Support Services, use a telecommunications relay service (TRS) to call Oracle Support at 1.800.223.1711. An Oracle Support Services engineer will handle technical issues and provide customer support according to the Oracle service request process. Information about TRS is available at http://www.fcc.gov/cgb/consumerfacts/trs.html, and a list of phone numbers is available at http://www.fcc.gov/cgb/dro/trsphonebk.html.

Related Documents

For more information, see the following documents in the Oracle Access Manager Release 10.1.4 documentation set:

- *Oracle Access Manager Introduction*—Provides an introduction to Oracle Access Manager, a road map to Oracle Access Manager manuals, and a glossary of terms.
- *Oracle Access Manager Release Notes*—Read these for the latest Oracle Access Manager information.
- *Oracle Access Manager Patchset Notes Release 10.1.4 Patchset 2 (10.1.4.3.0) For All Supported Operating Systems*—Read this document if you want to apply the 10*g* (10.1.4.3) patch set to an existing 10*g* (10.1.4.2.0) deployment. It includes a list of enhancements, bug fixes, and known issues related to the patch set.
- *Oracle Access Manager Installation Guide*—Explains how to prepare for, install, and set up each Oracle Access Manager component.
- *Oracle Access Manager Upgrade Guide*—Explains how to upgrade earlier releases to the latest major Oracle Access Manager release using either the in-place component upgrade method or the zero downtime method.
- *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 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 Oracle Access Manager 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 other Oracle and third-party products.
- *Oracle Access Manager Schema Description*—Provides details about the Oracle Access Manager schema.

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 describes new features of the Oracle Access Manager release 10.1.4. This includes details for 10*g* (10.1.4.0.1), 10*g* (10.1.4.2.0), and 10*g* (10.1.4.3).

The following sections are included:

> **Note:** For a comprehensive list of all new features and functions in Oracle Access Manager 10.1.4, and a description of where each is documented, see the chapter on what's new 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 Name	BEA WebLogic Application Server BEA WebLogic Portal Server	Oracle WebLogic Server Oracle WebLogic Portal
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

Item	Was	Is
Access System Service	AM Service State Policy Manager API Support Mode	Access Management Service **Note**: Policy Manager API Support Mode and Access Management Service are used interchangeably.

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

Enhancements Available in 10*g* (10.1.4.3)

Included in this release are new enhancements and bug fixes for 10*g* (10.1.4.3) in addition to all fixes and enhancements from 10*g* (10.1.4.2.0) bundle patches through BP07. The following topics describe 10*g* (10.1.4.3) enhancements described in this book:

- Access Management Service Clarifications
- Access Tester Use with Custom Authentication and Authorization Plug-ins
- Form-based Authentication Parameters
- Global Sequence Number Corruption Recovery
- Internet Protocol Version 6
- Multi-Language Deployments and English Only Messages
- Native POSIX Thread Library (NPTL) for Linux
- Troubleshooting Tips
- WebGate User-Defined Configuration Parameters

> **See Also:** *Oracle Access Manager Introduction* for a list of all new features and functions

Access Management Service Clarifications

Several clarifications have been made with regard to the Access Management Service in WebGate and Access Server profiles. This setting is Off by default. When set to On, the Access Server starts servicing requests from AccessGates. The Access Management Service must be On for associated Access Servers and AccessGates. The Access Management Service must be On for associated Access Servers and AccessGates. WebGates do not require the Access Management Service, unless an associated Access Server uses it.

> **See Also:** Chapter 3, "Configuring WebGates and Access Servers"

Access Tester Use with Custom Authentication and Authorization Plug-ins

New information has been added to describe how to use Access Tester when you have custom authentication or authorization plug-ins.

> **See Also:** "Using Access Tester" on page 4-44

Form-based Authentication Parameters

Oracle has added a new, optional, and configurable challenge parameter (`maxpostdatabytes`) for form-based authentication schemes only. Use of the `maxpostdatabytes` challenge parameter is similar to other challenge parameters (form, creds, action, and passthrough).

See Also: Appendix A, "Form-Based Authentication"

Global Sequence Number Corruption Recovery

The oblixGSN objectclass in the directory server is used in the cache flush mechanism. It contains a global sequence number (a value in the obSeqNo attribute) that represents the flush request number. When you have multiple Access Servers writing to multiple directory servers, however, changes could cause the global sequence number in the directory servers to get out of sync. Thus, corresponding entries in the directory servers might become corrupted, which can lead to inconsistent performance in Oracle Access Manager.

Oracle Access Manager provides functionality that enables you to detect corrupted GSNs in the directory server from the command-line tool (recovergsncorruption) in the following path: *PolicyManager_install_dir*\access\oblix\tools. If corruption is discovered, you can initiate recovery processing after disabling the cache flush operation between Identity Servers and Access Servers.

See Also:

- "Detecting and Restoring Corrupted Sync Records in Environments with Multiple Directory Servers" on page 8-12
- "Cache Flush Issues with Active Directory" on page E-5

Internet Protocol Version 6

Oracle Access Manager supports Internet Protocol Version 4 (IPv4). However, you can configure Oracle Access Manager to work with clients that support IPv6 by setting up a reverse proxy server.

See Also: Appendix D, "Using Oracle Access Manager with IPv6 Clients"

Multi-Language Deployments and English Only Messages

Messages for minor releases (10*g* (10.1.4.2.0) and 10*g* (10.1.4.3)) added for new functionality might not be translated and can appear in English only.

Native POSIX Thread Library (NPTL) for Linux

Earlier releases of Oracle Access Manager for Linux used the LinuxThreads library only. Using LinuxThreads required that you set the environment variable LD_ASSUME_KERNEL, which is used by the dynamic linker to decide what implementation of libraries is used. When you set LD_ASSUME_KERNEL to 2.4.19 the libraries in /lib/i686 are used dynamically.

RedHat Linux v5 and later releases support only Native POSIX Thread Library (NPTL), not LinuxThreads. To accommodate this change, Oracle Access Manager 10*g* (10.1.4.3) is compliant with NPTL specifications. However, LinuxThreads is used by default for all except Oracle Access Manager Web components for Oracle HTTP Server 11g.

Note: On Linux, Oracle Access Manager Web components for Oracle HTTP Server 11g use only NPTL; you cannot use the LinuxThreads library. In this case, do not set the environment variable LD_ASSUME_KERNEL to 2.4.19.

Troubleshooting Tips

Several new tips have been added to the troubleshooting details in this guide:

See Also: Appendix E, "Troubleshooting Oracle Access Manager"

- "Problem with Large Authorization Expressions" on page E-4
- "Cache Flush Issues with Active Directory" on page E-5
- "File Ownership and Command Line Tools" on page E-7
- "Policy Manager Hangs During Cache Flush from Policy Manager to Access Server" on page E-10
- "Mismatch Between Content and Content Length in a Web Server 401 Response" on page E-14

WebGate User-Defined Configuration Parameters

Several new user-defined parameters have been added for use in WebGate configuration profiles.

- ContentLengthFor401Response
- idleSessionTimeoutLogic
- ProxySSLHeaderVar
- RetainDownstreamPostData
- SUN61HttpProtocolVersion

See Also: "Configuring User-Defined AccessGate Parameters" on page 3-29

WebGate Updates

- WebGates have been updated to use the same code as the Access System, and WebGate configuration parameters that once existed in WebGateStatic.lst have been moved to the Access System user interface.

  After installing the new WebGates, you can now configure such parameters as IPValidation and IPValidationExceptions from the Access System GUI. The WebGateStatic.lst file no longer exists.

  See Also: "Configuring AccessGates and WebGates" on page 3-17, "Configuring User-Defined AccessGate Parameters" on page 3-29.

- WebGates can work behind a reverse proxy.

  Information on setting up a WebGate behind a reverse proxy has been added to this book.

  See Also: "Placing a WebGate Behind a Reverse Proxy" on page 3-40.

- Preferred HTTP Hosts are now required, and special configuration is needed to support virtual Web hosting.

  In this release, you must supply a Preferred HTTP Host when configuring a WebGate. If you use virtual Web hosting, there are new parameters for specifying a preferred HTTP host when your environment supports virtual hosting.

  See Also: "Configuring Preferred HTTP Hosts, Host Identifiers, and Virtual Web Hosts" on page 3-45 and "Configuring Virtual Web Hosting" on page 3-49.

URL Prefixes and Patterns

- The documentation on URL prefixes and patterns has been updated for clarity.

  See Also: "About URL Prefixes" on page 4-16, "About URL Patterns" on page 4-17.

- WebGates can work behind a reverse proxy.

  Information on setting up a WebGate behind a reverse proxy has been added to this book.

  See Also: "Placing a WebGate Behind a Reverse Proxy" on page 3-40.

Triggering Authentication Actions After the ObSSOCookie Is Set

- You can cause authentication actions to be executed after the ObSSOCookie is set.

  Typically, authentication actions are triggered after authentication has been processed and before the ObSSOCookie is set. However, in a complex environment, the ObSSOCookie may be set before a user is redirected to a page containing a resource. In this case, you can configure an authentication scheme to trigger these events.

  See Also: "Triggering Authentication Actions After the ObSSOCookie is Set" on page 5-63.

Form-based Authentication

- Globalization support impacts basic authentication and form-based authentication methods.

  See Also: "About Basic and X.509Cert (Client Cert)" on page 5-5, "Sample Login Form" on page A-15.

- Information has been added about the differences between configuring a form on the server where the WebGate resides and configuring it on a server other than the one hosting the WebGate.

  See Also: "Configuring a Form-Based Authentication Scheme" on page A-8 and "Forms that Reside on Servers Other Than a WebGate" on page A-11.

- Information has been added about configuring single logout from applications that use form-based authentication.

  See Also: "Configuring Logout" on page B-1.

Disabling Authentication Schemes

- It is no longer necessary to disable an authentication scheme before you modify it.

  See Also: Configuring User Authentication on page 5-1.

Persistent Cookies in Authentication Schemes

- You can configure an authentication scheme that allows the user to log in for a time period rather than a single session.

  See Also: "Retaining the ObSSOCookie Over Multiple Sessions" on page 5-17.

HTTP Header Variables and Cookies

- Non-ASCII characters are not permitted in HTTP headers or cookies. Thus, non-ASCII characters are not supported in the header variable Name and Return Attribute fields when you define authentication rule actions.

  See Also: "About the Use of HTTP Header Variables and Cookies" on page 5-55.

Configuring Logout

- You can configure the Oracle Access Manager single sign-on logout URL to point to a `logout.html` file in the language of the user's browser.

  See Also: "Configuring a Single Sign-On Logout URL" on page 2-6.

- A section has been added on creating custom single sign-on logout URLs and logout pages.

  See Also: "Configuring Logout" on page B-1.

Associating WebGates with Specific Virtual Hosts, Directories, and Files

- You can configure the WebGate to only work with specific virtual hosts, directories, and files.

  See Also: "Associating a WebGate with Particular Virtual Hosts, Directories, or Files" on page 3-51.

Configuring the validate_password Plug-In

- An additional parameter has been added to this plug-in, to be used for Lost Password Management, when a WebPass is on a different server from the WebGate that protects the requested resource.

  See Also: "Validate Password Plug-In" on page 5-24

Overriding Windows-Enabled Impersonation

- The appendix on using the Access System to override Windows-enabled Impersonation has been moved.

  See Also: *Oracle Access Manager Integration Guide*.

Configuring Lotus Domino and Windows Impersonation Single Sign-On

- The information on configuring single sign-on with Lotus Domino and Windows Impersonation has been moved to another book in this suite.

  See Also: *Oracle Access Manager Integration Guide*.

Troubleshooting

- Information on troubleshooting that was dispersed throughout this manual has been consolidated in a separate appendix.

  See Also: "Troubleshooting Oracle Access Manager" on page E-1.

- You can now write diagnostic information to a log file.

  See Also: "Capturing Diagnostic Information" on page E-15.

- New troubleshooting topics have been added.

  See also:

  - "WebGate Diagnostics URL Incorrectly Report that the Access Server Is Down" on page E-13
  - "WebGate Cannot Connect to its Associated Access Server" on page E-14
  - "Error Message to Check if the Directory Server is Running or Responding" on page E-6

Part I

Configuring the Access System

Before you begin assigning Access System administrators and managing other server settings, it is a good idea to familiarize yourself with the information here.

Part I contains the following chapters:

- Chapter 1, "Overview of Access System Configuration and Administration"
- Chapter 2, "Configuring Access Administrators and Server Settings"
- Chapter 3, "Configuring WebGates and Access Servers"

1

Overview of Access System Configuration and Administration

This chapter provides an overview for people who are new to Access System setup and administration.

This chapter discusses the following topics:

> **Note:** This chapter assumes that you have at least a little familiarity with the purpose of Oracle Access Manager and the Identity System. For references to these topics, see the "Preface" on page -xv.

About the Access System

The Access System provides centralized authentication, authorization, and auditing to enable single sign-on and secure access control across enterprise resources. You use the Access System to set up security policies that control access to resources. Resources include Web content, applications, services, and objects in applications, and similar types of data in non-Web (non-HTTP) resources.

The Access System stores information about configuration settings and access policies in a directory server that uses Oracle Access Manager-specific object classes. You can use the same directory to store the Access System configuration settings, access policy data, and the Identity System user data, or this data can be stored on separate directory servers.

Access System Components

The Access System consists of the following components:

Policy Manager

The Policy Manager is installed on a Web server in the same directory as the Identity System component WebPass. See the *Oracle Access Manager Introduction* manual for an illustration that shows the location of WebPass. The Policy Manager provides a login

interface to the Access System. Master Access Administrators and Delegated Access Administrators use the Policy Manager to define resources to be protected, and to group resources into policy domains. A policy domain consists of resource types to protect, rules for protection, policies for protection, and administrative rights.

The Policy Manager has a component called the Access System Console, that permits administrators to add, change, and remove Access Clients and Access Servers, configure authentication and authorization schemes, configure master audit settings, and configure host identifiers.

You do not need to configure the Policy Manager application user interface the way you do the Identity System applications.

Access Server

The Access Server is a standalone server that provides authentication, authorization, and auditing services. You can have multiple instances installed. The Access Server validates credentials, authorizes users, and manages user sessions. The Access Server receives requests from an Access Client and queries authentication, authorization, and auditing rules in the directory server as follows:

- Authentication involves determining what authentication method is required for a resource, gathering credentials over HTTP, and returning an HTTP response that is based on the results of credential validation.
- Authorization involves granting access based on a policy and an identity established during authentication.

WebGate

The WebGate is an out-of-the-box Access Client for HTTP-based resources. WebGate is a plug-in that intercepts HTTP requests for Web resources and forwards them to the Access Server.

The Access System supports single sign-on, enabling you to establish login policies that allow users to access multiple applications with a single login.

Review of Access System Installation and Setup

During Access System installation and setup, you complete the following tasks as described in the *Oracle Access Manager Installation Guide*:

- Install and configure the Policy Manager application.
- Select a directory to store access policies.
- Configure Policy Manager to communicate with the directory server that stores access policies.
- Configure one or more authentication schemes.

  Configuring authentication schemes during setup is optional.
- Install and configure at least one Access Server and one AccessGate.
- Select the Access Server's transport security communication mode.

  You should ensure that communications are secure by selecting Simple or Cert transport security.
- Optionally, install and set up additional instances of these components interactively, or by using silent installation or cloning.

Table 1–1 summarizes Access System installation and setup, which is described in detail in the *Oracle Access Manager Installation Guide.*

Table 1–1 Overview of Access System Installation and Setup

To perform this task	Read
Install the Policy Manager	*Oracle Access Manager Installation Guide* chapter on installing the Policy Manager
Set up the Policy Manager	*Oracle Access Manager Installation Guide* section on setting up the Policy Manager
Install the Access Server	*Oracle Access Manager Installation Guide* chapter on installing the Access Server
Install a WebGate	*Oracle Access Manager Installation Guide* chapter on installing the WebGate

About Configuring Resources and Rules for Who Can Access Them

The Access System enables you to control who is allowed to access data. You can create access policies that extend beyond the Identity System applications. For example, if you have an online benefits system, you can configure access policies that only permit employees to view portions of the benefits Web site that are relevant to them. Or you can configure access policies so that external customers are allowed to see your inventory Web pages but not other corporate information.

Table 1–2 provides an overview of configuring the Access System.

Table 1–2 Overview of Access System Policy-Related Configuration

Perform this task	Description	Read
Enter host IDs	Map host name variations to a single Web server instance. This ensures that the Access System can process variations in URLs that it receives when users request resources.	"Configuring Host Identifiers" on page 3-47
Create a policy domain and define resources to protect	A resource is something you want to protect, such as a Web page, plus the actions applied to that item, for instance, an update. A policy domain defines a set of resources to protect. You identify the resources in the domain using fully qualified path names or URLs, plus rules for protection, policies for protection, and administrative rights.	"Protecting Resources with Policy Domains" on page 4-1
Create policies for URL patterns	Default rules apply to all URLs in a policy domain. You can also specify individual policies with their own authorization, authentication, and auditing rules for URL patterns and functions, for example, HTTP get, put, and so on.	"About Policy Domains and Their Policies" on page 4-6
Create an authentication scheme	Validate the identities of people who want to access your resources. Define the method of authentication (for instance, x.509 certificates), the plug-in used to map authentication credentials to a user's identity in the directory, and mapping to the user's DN in the directory.	"Configuring User Authentication" on page 5-1

Table 1–2 (Cont.) Overview of Access System Policy-Related Configuration

Perform this task	Description	Read
Create an authorization scheme	Determine if people with valid credentials are permitted (authorized) to access particular resources, and perform actions on those resources based on the authorization rules.	"Configuring User Authorization" on page 6-1
Create a master audit rule	The Access System must have a Master Audit Rule to begin adding data to the audit log file. The audit log file records administrative events such as clearing data from caches.	"About the Master Audit Rule" on page 4-34.
Configure single sign-on	Single sign-on allows users to authenticate to multiple applications with one login.	"Configuring Single Sign-On" on page 7-1
Create a shared secret	The shared secret is used to generate the key that encrypts cookies sent between the WebGate and the user's browser.	"Creating a Shared Secret Key" on page 8-4

Note: Before you define policy domains and policies, you may want to define a few Access Administrators and configure at least one Access Server and WebGate, as mentioned in Table 1–3.

About Configuring and Managing the Access System Components

You configure the Access System by defining administrators, adding components such as Access Servers and AccessGates, and setting basic system parameters.

You also manage the Access System by adding more servers, by defining caching parameters, and by extending your access policies using custom plug-ins. Table 1–3 provides an overview of managing the Access System.

Table 1–3 Overview of Managing the Access System

To perform this task	Read
Configure Access Administrators	"Configuring Access Administrators and Server Settings" on page 2-1
Configure server settings	"Configuring Access Administrators and Server Settings" on page 2-1
Configure WebGates, AccessGates and Access Servers	"Configuring WebGates and Access Servers" on page 3-1
Add Access Servers	The *Oracle Access Manager Installation Guide*. You can add components interactively, or by using silent installation or cloning.
Configure directory instances	The *Oracle Access Manager Identity and Common Administration Guide*
Configure logging, monitoring, and auditing	The *Oracle Access Manager Identity and Common Administration Guide*
Install the Access Manager SDK	The *Oracle Access Manager Developer Guide*
Configure non-HTTP access clients	The *Oracle Access Manager Developer Guide*

Table 1–3 (Cont.) Overview of Managing the Access System

To perform this task	Read
Manage caching	The *Oracle Access Manager Deployment Guide*
Configure failover	The *Oracle Access Manager Deployment Guide*
Tune performance	The *Oracle Access Manager Deployment Guide*

2

Configuring Access Administrators and Server Settings

This chapter explains how to assign Access System administrators and manage other server settings. Included here are the following topics:

- Prerequisites
- Configuring Access Administrators
- Managing Server Settings

For more information about managing the Access System, see:

- Configuring WebGates and Access Servers on page 3-1

Prerequisites

Oracle Access Manager 10.1.4 should be installed and set up as described in the *Oracle Access Manager Installation Guide*. The *Oracle Access Manager Introduction* provides an overview of Oracle Access Manager not found in other manuals. Also, familiarize yourself with the *Oracle Access Manager Identity and Common Administration Guide*, which provides a brief review of Access System applications and installation and describes functions that are common to the Access and Identity Systems, including defining logging, auditing, and password policies.

Configuring Access Administrators

The Access System enables the protection of online resources by enforcing policy-based authentication and authorization rules. The Access System also enables Web single sign-on.

In addition to the Master Administrator, there are two types of administrators who can configure and manage the Access System:

- **Master Access Administrators:** These administrators have the right to perform any task in the Access System except the right to create other Master Access Administrators.
- **Delegated Access Administrators:** These administrators only have the right to perform tasks that a Master Access Administrator delegates to them.

Table 2–1 summarizes the privileges of these types of administrators. Master Access Administrators automatically have these privileges while Delegated Access Administrators must be explicitly granted these privileges:

Table 2–1 Table of Administrative Privileges

Privilege	Description	Who Performs This Task
Generate a shared secret	Create a cryptographic key that encrypts single sign-on cookies. See "Creating a Shared Secret Key" on page 8-4.	Master Access Administrator
Configure the Master Audit Rule	The Access System will not log any audit information to the audit log file until a Master Audit Rule exists. See "About the Master Audit Rule" on page 4-34. For more information about logging, see *Oracle Access Manager Identity and Common Administration Guide*.	Master Access Administrator
Flush the password policy cache	See "Flushing Password Policy Caches" on page 8-6.	Master Access Administrator
Manage AccessGates	View, create, and configure one or more instances of an AccessGate. See "Configuring AccessGates and WebGates" on page 3-17.	Master or Delegated Access Administrator
Manage Access Servers	Configure an Access Server to communicate with AccessGates and a directory server. See "Configuring Access Servers" on page 3-3.	Master or Delegated Access Administrator
Manage Access Server clusters	See "Managing Access Server Clusters" on page 3-10.	Master or Delegated Access Administrator
Manage Authentication Schemes	Authentication is the process of proving that a user is who he or she claims to be. See Chapter 5, "Configuring User Authentication" on page 5-1.	Master or Delegated Access Administrator
Manage Authorization Schemes	Authorization is the process of determining if a user has the right to access a requested resource. See Chapter 6, "Configuring User Authorization" on page 6-1.	Master or Delegated Access Administrator
Manage Host Identifiers	Identify the names by which users can identify a host. See "Configuring Preferred HTTP Hosts, Host Identifiers, and Virtual Web Hosts" on page 3-45.	Master or Delegated Access Administrator
Manage Resource Type definitions	Define the kind of resource to be protected, including its associated operations. See "Default Resource Types" on page 4-12.	Master or Delegated Access Administrator
Manage User Configuration	Create and modify a list of users who are prohibited from accessing any of your resources and flush these users from the cache. See "About Access System Configuration and Management" on page 8-1.	Master or Delegated Access Administrator

The following sections describe how to configure these administrators and delegate administrative tasks. You complete these tasks using the Access System Console, System Configuration function.

Note: Delegating administrative responsibilities for a policy domain is somewhat different from the delegation of other responsibilities. See "Delegating Policy Domain Administration" on page 4-46 for details.

Configuring Master Access Administrators

Only Master Administrators can create Master Access Administrators. A Master Access Administrator can perform any function in the Access System except for creating other Master Access Administrators, and can delegate administrative functions.

Note: You must be a Master Access Administrator to create a shared secret key that encrypts single sign-on cookies. You should generate a cryptographic key as soon as possible after installing Oracle Access Manager, otherwise a less secure default is used. See "Creating a Shared Secret Key" on page 8-4.

To add a Master Access Administrator

1. From the Access System Console, select System Configuration, then click the Administrators link in the left navigation pane.

   The Configure Administrators page lists current Master Access Administrators.

2. Click the Master Access Administrators link.

   The Modify Master Access Administrators page appears.

3. Click Select User.

   A page appears that contains search fields.

4. Use the search fields to select the people that you want.

   The search fields consist of attributes that you want to search, search criteria such as "contains," and search strings or partial strings. Select the number of search results that you want to view at a time and click Go.

5. Click Done to return to the Modify Master Access Administrators page.

   The names of any new people you chose using the Selector are displayed in the Modify Master Access Administrators page.

6. Use the checkboxes to deselect any names that you need to remove from your list.

7. Review your selections to ensure that your list is complete.

8. Click Save to save the changes (or Cancel to exit without changing).

Configuring Delegated Access Administrators

When the responsibility for managing the Access System falls on a few people, you may want these people to appoint others to share the work. People currently responsible for resources generally know best to whom to delegate responsibility. The ability to delegate Access System administration to other people enables you to scale administration of your resources, empowering those closest to the resources and most knowledgeable about them to manage them.

A Master Access Administrator can create a group of users and assign administrative rights to the group. The Master Access Administrator can assign the same administrative rights to multiple groups. For example, Group1 and Group2 can both be assigned the right to manage Access Servers.

The following functions can be delegated:

- Add, modify, delete AccessGate configurations.

- Add, modify, delete Access Server configurations.
- Add, modify, delete Access Server clusters.
- Add, modify, delete authentication schemes.
- Add, modify, delete authorization schemes.
- Add, modify, delete host identifiers.
- Add, modify, delete resource type definitions.
- Modify the revoked user list.

  To manage the revoked user list, a delegated administrator must have access to the searchbase containing the entry for the user and must have appropriate attribute read permissions.

You can add a user to multiple groups. For example, if you create one group of Delegated Administrators to manage authentication schemes and authorization schemes, and another group to manage Access Servers and Access Server clusters, the same user can belong to both groups.

When an administrator performs certain tasks, Oracle Access Manager creates an informational log. See the *Oracle Access Manager Identity and Common Administration Guide* for details.

Policy domain administration can also be delegated. See "Delegating Policy Domain Administration" on page 4-46 for details.

Note: A delegated administrator can be assigned to a resource type before any host IDs are set. However, if the host IDs are defined at a later time, the delegated administrator will no longer be able to add resources to the policy domain. The Master Administrator will need to reassign the delegated administrator to a policy domain that has the associated host identifier to enable the delegated administrator to add resources to that domain.

Creating a Group of Delegated Access Administrators

The following procedure illustrates how to add Delegated Administrators to the Access System.

To create a group of Delegated Access Administrators

1. From the Access System Console, click System Configuration, then click the Administrators link in the left navigation pane.

   The Configure Administrators page appears.

2. Under the title Groups of Delegated Administrators, click the Add button.

   The Create a New Group of Delegated Administrators page appears. You can complete all information requested or create an empty group with no administrative rights or members.

3. Provide the information requested.

4. For example:

   Name: A name for this group

   Description: Optional description

Administrative Rights: Select the rights you want to give to this group

5. Click the Select User button, beside the Members label, to display the Selector.
6. Use the Selector to add people to this group, then click Done when you are finished to return to the Create a new group of Delegated Administrators page.
7. Click Save to complete the process.

Modifying a Group of Delegated Administrators

The following procedure illustrates how to alter a group of Delegated Administrators in the Access System.

To modify a group of delegated administrators

1. From the Access System Console, click the System Configuration tab, then click the Administrators link in the left navigation pane.
2. Click the link for the group to modify.

   The Modify Group of Delegated Administrators page appears.
3. Click Modify.

   The page changes to show editable fields for group name, description, and so on.
4. Make your changes and click Save.

Managing Server Settings

The Access System Console, System Configuration function, enables you to view and alter Access Server and directory server settings, configure an SSO Logout URL, and configure email addresses for user feedback. The following topics are covered:

- Viewing Server Settings
- Customizing Email Addresses
- Configuring a Single Sign-On Logout URL
- Configuring the Directory Server

Note: Only Master Administrators can alter these settings.

Viewing Server Settings

You use the Access System Console to view server settings for items such as email addresses, directory servers, and the SSO logout URL.

To view server settings

1. Launch the Access System Console.
2. Click System Configuration, then select Server settings.

   The View Server Settings page appears.

Customizing Email Addresses

You use the Customize Email function to specify email addresses for user feedback.

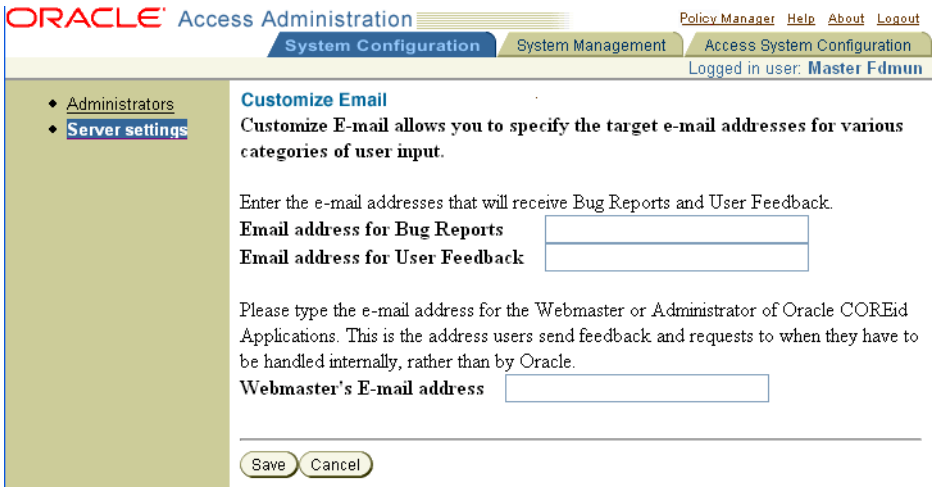
The end user accesses email addresses by clicking the About link at the top of the page, then clicking Submit Admin Feedback or Submit Oracle Feedback.

To customize email

1. In the Access System Console, click System Configuration, then select Server settings.

   The View Server Settings page appears.

2. Click Customize Email to display this page.



3. Type email addresses in the following fields:

Address	Description
Email address for Bug Reports	This address must be changed to be sent to a person or alias in your organization. This person or department can either solve the problem or contact Oracle for help.
Email address for User Feedback	When a user submits an Oracle Feedback form, the data is sent to the address specified. The default is feedback@Oracle.com.
Webmaster's Email address	When a user submits an Admin Feedback form, the data is sent to the address specified. The default is webmaster@company.com.

4. Click Save to save your changes (or Cancel to exit without saving).
5. Restart your Web server.

Configuring a Single Sign-On Logout URL

Single sign-on (SSO) is the ability to access multiple resources with a single login. The Access System performs single sign-on for users by setting an ObSSOCookie for each user or application that accesses a resource protected by the Access System. The ObSSOCookie enables users to access other resources protected by the Access System that have the same or a lower authentication level. See Chapter 7, "Configuring Single Sign-On" on page 7-1 for details.

You can configure a single sign-on logout URL and an associated logout page to remove the ObSSOCookie. This forces the user to re-authenticate the next time he or she accesses a resource protected with the Access System.

Oracle provides a logout.html page that is presented to users upon logout and that runs the function that removes session cookies. This form is located in:

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

For information on configuring this logout page or creating a custom one, see "Configuring Logout" on page B-1. This section only discusses the logout URL.

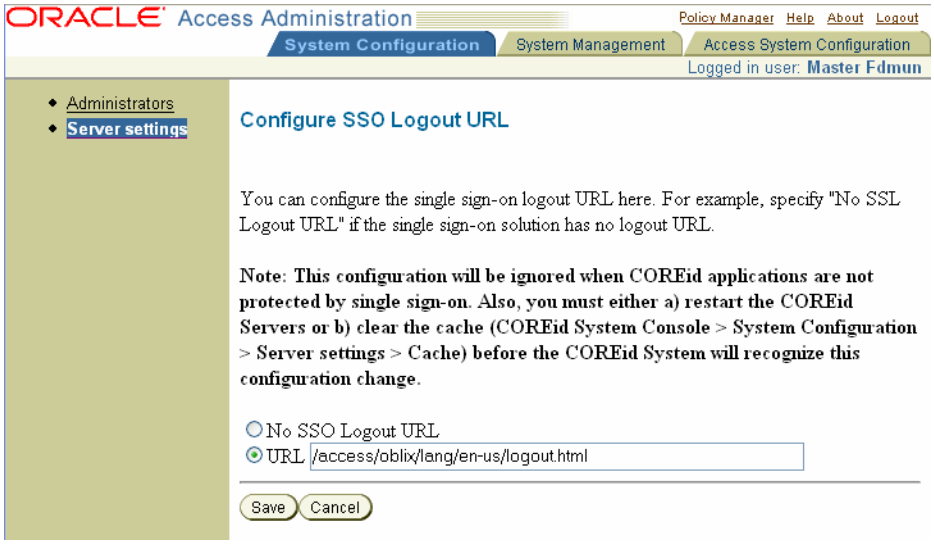
If you have multiple languages installed, you can configure the Oracle Access Manager single sign-on logout URL to point to a `logout.html` file in the language of the user's browser. To configure the logout URL, you provide the `%lang%` parameter in the single sign-on logout URL. Access Manager replaces `%lang%` with the browser's language at run time.

Note: If you use the Basic Over LDAP authentication scheme on some versions of Internet Explorer, you may experience unexpected results with the single sign-on logout URL. Internet Explorer caches user credentials when a Basic Over LDAP authentication scheme is used. For some versions of Internet Explorer, this means that users can continue to access resources after logging out. If you experience this problem with the single sign-on logout URL, Oracle recommends that you use a Form over LDAP authentication scheme.

To configure the SSO Logout URL

1. In the Access System Console, click System Configuration.
2. Click Server Settings in the left navigation pane.
3. Click the Configure SSO Logout URL link.

   The following page appears.



4. Choose the option you want:
   - If you use a third-party program for logging users out, select No SSO Logout URL
   - If you want to have the Identity System and Access System automatically call this page when the user clicks Logout, select URL.

Note: You must manually create a link to this logout.html page from other resources that are protected by the Access System. Create the link on the pages that you want to contain the logout feature.

5. Click Save.
6. Flush the Access Server cache after changing the SSO Logout value.

   See "Automatic Access System Cache Flush" on page 9-1 for more information.
7. Flush the Identity Server cache after changing the SSO Logout value.

   For more information about managing Identity Server caches, see the *Oracle Access Manager Identity and Common Administration Guide* and the *Oracle Access Manager Deployment Guide*.

Configuring the Directory Server

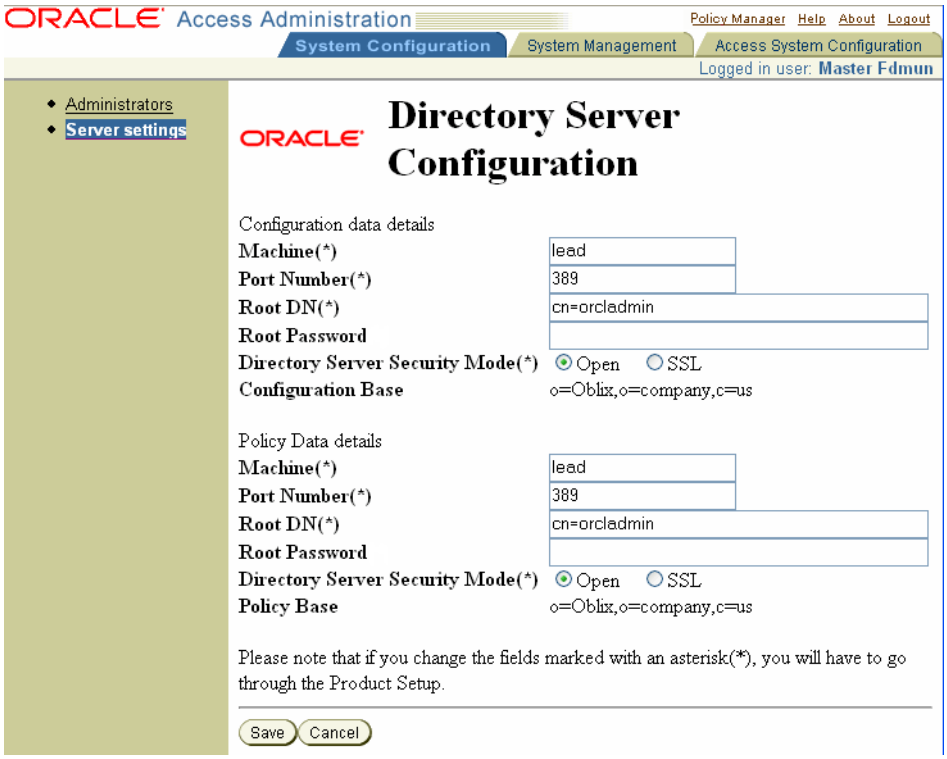
You use the Directory Server Configuration page to modify various directory server settings using the Access System Console. This is similar to modifying Directory Server details using the Identity System Console, as discussed in the *Oracle Access Manager Identity and Common Administration Guide*. Directory server details available in the Access System Console include those for configuration data and policy data.

To configure the directory server

1. From the Access System Console, click System Configuration, then click Server settings.

   The View Server Settings page appears.
2. Click the Directory Server link.

   The Directory Server Configuration page appears. Notice that the page is divided into two areas: one for configuration data and one for policy data. The configuration base and policy base on this page cannot be changed.



The configuration base identifies the location of all Oracle Access Manager-specific information. You cannot change this information. The Policy Base identifies the location in the DIT under which all Access System policy data is stored, which you cannot change.

> **Note:** If you change the information in any field marked with an asterisk (*), you must repeat product setup as described in the *Oracle Access Manager Identity and Common Administration Guide*.

3. Specify configuration information for configuration data, as shown in the following table.

Field	Description
Machine(*)	Name or IP address of the computer where the directory server managing the user data, configuration data, or policy data is installed
Port Number(*)	Port number of the directory server managing the user data, configuration data, or policy data is listening
Root DN(*)	Root DN of the directory server
Root Password	Root password of the directory server
Directory Server Security Mode(*)	Security mode the directory server uses to protect its communications

4. Specify configuration information for Policy data, as shown in the previous table.
5. Click Save to save your changes (or Cancel to exit without saving).

3

Configuring WebGates and Access Servers

WebGates, AccessGates, and Access Servers are key components when a user attempts to access a protected resource. An Access Server provides an authentication, authorization, and auditing engine. You install a WebGate for each Web server that hosts Web content that you want to protect. The WebGate intercepts HTTP requests for Web content on the server and forward the requests to an Access Server. An AccessGate is a custom WebGate that can intercept requests for HTTP and non-HTTP resources.

This chapter explains how to define and manage AccessGate and Access Server instances.

This chapter includes the following topics:

About Configuring the Access System

Controlling access to applications and content is the cornerstone of e-business infrastructure. You want to allow some users to use certain resources and deny access to others. You control access to your company's resources through the Access System. The Access System provides base components, consisting of a server and plug-ins, that are the foundation upon which you define access controls.

The following sections discuss setting up the base components of the Access System. These components provide the underlying software that makes it possible to control user access to resources.

As discussed in *Oracle Access Manager Introduction* and this manual, the following are the primary components of the Access System:

- **Access Server**—A standalone server that provides authentication, authorization, and auditing services for AccessGates.
- **WebGate**—WebGate is an out-of-the-box plug-in that is installed on each server that you want to protect. The WebGate intercepts Web resource (HTTP) requests and forwards them to the Access Server for authentication and authorization.
- **AccessGate**—A custom WebGate that processes resource requests from users or applications. It intercepts user requests for resources and forwards them to the Access Server for authentication and authorization. Unlike a WebGate, you can define a custom AccessGate that intercepts requests for non-http resources.

For information about how Access Servers and WebGates protect resources, see "The Access Login Process" on page 3-52.

You administer the Access System through a Web-based user interface that consists of the Policy Manager and the Access System Console.

Policy Manager—This is an Access System application that enables you to create and manage policy domains to protect resources, and to test policy enforcement.

Access System Console—This is an Access System application that provides functions for the following configuration and management tasks:

- **System Configuration**—Functions include configuring administrators and server settings. The View Server Settings page provides information on email, the single sign-on logout URL, and the directory server. See "Configuring Access Administrators and Server Settings" on page 2-1 for details.
- **System Management**—Enables you to identify Access Servers on which to run diagnostics, manage user access-privilege reports, and manage sync records. See "About Access System Configuration and Management" on page 8-1 for details.
- **Access System Configuration**—Includes the following functions (see also "Managing Access System Configuration Files" on page 9-1):
  - Defining Access Server clusters
  - Creating and configuring AccessGates
  - Configuring Access Servers
  - Configuring user authentication
  - Configuring user authorization
  - Configuring common information
  - Configuring preferred hosts and host identifiers

Prerequisites for Configuring AccessGates and Access Servers

Oracle Access Manager 10.1.4 should be installed and set up as described in the *Oracle Access Manager Installation Guide*. The *Oracle Access Manager Introduction* provides an overview of Oracle Access Manager not found in other manuals. Also, familiarize yourself with the *Oracle Access Manager Identity and Common Administration Guide* which introduces Access System applications, installation, configuration, administration, and common functions.

Important: You must have appropriate rights to complete activities in this chapter. See "Configuring Access Administrators and Server Settings" on page 2-1 for more details.

Configuring Access Servers

As described in the *Oracle Access Manager Installation Guide*, you must install at least one Access Server. Oracle recommends that you install at least two Access Servers on different computers to ensure uninterrupted service to your users. Each Access Server must be configured to communicate with one or more AccessGate instances, and to communicate with a directory server.

You must have appropriate rights to configure the Access System. See "Configuring Access Administrators and Server Settings" on page 2-1 for more information.

Task overview: Creating an Access Server

1. Create an Access Server instance in the Access System Console, as described in "Adding an Access Server Instance" on page 3-5.
2. Install the Access Server, as described in the *Oracle Access Manager Installation Guide*.
3. Configure the Access Server, as described in "Modifying Access Server Details" on page 3-9.
4. Synchronize the clocks on the Oracle Access Manager hosts.

   Access Server clocks can be ahead of WebGates by no more than 60 seconds. Webgate clocks should never be ahead of Access Server clocks.

   Access Servers record their activity in Greenwich Mean Time (GMT) because you could have servers operating in several time zones. Synchronizing clocks on Oracle Access Manager hosts is critical. See the *Oracle Access Manager Installation Guide* for details.
5. Ensure optimal performance by tuning the directory that the Access Server communicates with.

   See your directory vendor's documentation for details. For Oracle Internet Directory, see the chapter on performance optimization in the *Oracle Internet Directory Administrator's Guide* for details.

Using the Access System Console, Access System Configuration function, you can perform several key tasks including the following:

- Viewing Access Server Configuration Details
- Adding an Access Server Instance
- Modifying Access Server Details
- Deleting an Access Server
- Clustering Access Servers
- Managing Access Servers from the Command Line

Viewing Access Server Configuration Details

You can view all the configured Access Servers in the Access System Console.

To view Access Server configuration details

1. Launch the Access System Console.
2. Click Access System Configuration, then select Access Server Configuration.

   The existing Access Servers are listed on the configuration landing page.

3. Select an Access Server to view its configuration.

   The configuration details of the Access Server appear, as described next. For details about configuring these parameters, see "Adding an Access Server Instance" on page 3-5.

Access Server Configuration Parameters

The Access Server configuration parameters appear on the Access System Console, Access Server Configuration page. Table 3–1 lists the configuration parameters:

Table 3–1 Access Server Configuration Parameters

Field	Description
Name	Name of the Access Server.
Hostname	Name of the Web server that is hosting the Access Server.
Port	Port number the Access Server is listening to.
Debug	Indicates whether debugging is on or off. See "Adding an Access Server Instance" on page 3-5 for details.
Debug File Name	The name of this Access Server's debug file. The absolute path to the debug file is also indicated. If the file does not exist, it is created after you restart the Access Server.
Transport Security	Level of transport security to and from the Access Server. Available options are: **Open**—No transport security. **Simple**—Encrypted transport security with prepackaged certificates. **Cert**—Encrypted transport security.
Maximum Client Session Time (hours)	The duration, in hours, for a connection between AccessGate and an Access Server.
Number of Threads	Maximum number of service threads allowed on the Access Server. By default, the number of threads is set to 60. The number of threads may have implications for system performance. See the *Oracle Access Manager Deployment Guide* for details.
Access Management Service Also known as Policy Manager API Support Mode	Identifies whether the Access Management Service is enabled (On) or not (Off). This setting is Off by default. When set to On, the Access Server starts servicing requests from AccessGates. The Access Management Service must be On for associated Access Servers and AccessGates. See also "AccessGate Configuration Parameters" on page 3-19.
Audit to Database	Writes audit information to a database. If you set the value to On, you must install a supported database and do additional configuration. See the *Oracle Access Manager Identity and Common Administration Guide* for details.
Audit to File	Writes audit information to a file.
Audit File Name	Path to this Access Server's audit file.
Audit File Size	Maximum size of the audit file, in bytes.
Buffer Size	Size of the audit buffer, in bytes.
File Rotation Interval	Time, in seconds, that this audit file can exist.

Table 3–1 (Cont.) Access Server Configuration Parameters

Field	Description
Engine Configuration Refresh Period	Frequency, in seconds, of configuration updates to this server. **Note:** Changes you make to this parameter do not take effect until the previous Engine Configuration Refresh Period has expired.
URL Prefix Reload Period	Frequency, in seconds, with which new URLs are recognized by this Access Server.
Password Policy Reload Period	Frequency, in seconds, with which new password policies are recognized by this Access Server.
Maximum Elements in User Cache	Number of authenticated users that can be saved in the cache for this Access Server. The default is 10000. A value of -1 disables the cache.
User Cache Timeout	Maximum time, in seconds, for inactive user data to reside in cache.
Maximum Elements in Policy Cache	Maximum number of elements that can be stored in the policy cache.
Policy Cache Timeout	Maximum time, in seconds, for inactive policy data to reside in cache.
SNMP State	Specifies whether SNMP is enabled or not. For details on SNMP monitoring, see the *Oracle Access Manager Identity and Common Administration Guide*.
SNMP Agent Registration Port	The port number of the SNMP agent.
Session Token Cache	The session token cache can be used when decrypting the user's session token on the Access Server. If enabled, the Access Server checks if the decrypted session token resides in the cache. If not, the Access Server decrypts the incoming session token. For a description of the session token, see "Single Sign-On Cookies" on page 7-2.
Maximum Elements in Session Token Cache	The maximum number of decrypted session tokens that can be kept in the cache. This cache should be tuned periodically because the cache can become very large if the Access System manages many users and the activity rate is high. The default value is 10,000.

Adding an Access Server Instance

Before installing the Access Server, you must add a new instance in the Access System Console. At this time, you need only specify the Access Server name, hostname, port, and transport security mode. After installation, you can modify the configuration.

The following procedure describes how to add and configure the instance.

Note: You must add the Access Server instance to the Access System Console before installing the component.

To add an Access Server instance

1. From the Access System Console, select Access System Configuration, then click Access Server Configuration.

   The Access Server Configuration page appears.

2. Click Add.

   The Add a New Access Server page appears.

You must enter information in the Name, Hostname, and Port fields. All other fields are optional.

3. In the Name field, type a name for this server.

   Type an alphanumeric string without spaces.

   Note: You cannot give the same name to an AccessGate and an Access Server.

4. In the Hostname field, type the name or IP address of the computer hosting this server.
5. In the Port field, type the port number of the computer hosting this server.
6. In the Debug field:
   - Click On to capture all messages sent from each AccessGate and Policy Manager to this Access Server.

     The messages are stored in a Debug file if a Debug file is provided. Otherwise, the messages are printed out to stderr.
   - Click Off if you do not want to capture this information.

     Capturing messages confirms that communication is taking place between AccessGate instances and this Access Server.

   Note: Capturing debug messages records user passwords, a potential security problem, and causes the Access Server log file to grow rapidly. Use debugging only when diagnosing a problem.

7. In the Debug File Name field, type the path to this Access Server's debug file.
8. In the Transport Security field, select a method for encrypting messages between this Access Server and the AccessGates it is configured to talk to.

   For AccessGates and Access Servers that are configured to communicate with each other, be sure to choose the same encryption method.

   Your choices are Open mode, Simple mode, or Cert mode.

   For a description of configuring transport security modes, see the *Oracle Access Manager Identity and Common Administration Guide*.
9. In the Maximum Client Session Time (hours) field, type the number of hours that a connection between an AccessGate and this Access Server can last.

   The default is 24 hours.

   The longer the session time, the more vulnerable your system is to attack.
10. In the Number of Threads field, type the number of threads this Access Server will start.

    The default entry is 100. The minimum is 1.
11. Beside Access Management Service select the option for your environment.

Note: For Access Servers associated with AccessGates, ensure that the Access Management Service is On. WebGates do not require the Access Management Service to be On, unless associated Access Servers have Access Management Service On. For more information on caching, see the *Oracle Access Manager Deployment Guide*.

Automatic cache flush requires the Access Manager SDK that is bundled with the Identity Server. During automatic cache flushing, the Identity System uses an AccessGate to communicate with the Access Server (using APIs available in the SDK). In this case, setting the Access Management Service to "On" is required in both Access Server and associated AccessGate profiles. See "AccessGate Configuration Parameters" on page 3-19 for details.

12. Choose the Auditing option you want for your environment:
    - Audit to Database (on/off)— Selecting On requires specific configuration in the Access System, plus installation of a supported database, as described in the *Oracle Access Manager Identity and Common Administration Guide*.
    - Audit to File (on/off)—Selecting On requires specification of the following additional items:
      - Audit File Name—Type the path to this Access Server's audit file. The file is stored on the computer hosting this Access Server. Each Access Server has its own audit file. The information captured for the file is determined by the Master Audit Rule. See "About the Master Audit Rule" on page 4-34 for details.
      - Audit File Size (bytes)—Type a number representing the number of bytes this audit file can hold. If you change the default size, you need to restart the server after committing the change. When the maximum size is reached, the current file is closed and stamped with the date and time, and a new audit file with the original name is created.
13. In the Buffer Size (bytes) field, type a number representing the number of bytes that the audit file's buffer can hold.

    The default is 512000. Using a buffer increases performance by reducing the number of times information is read to the audit file.
    - If you type a value in this field, audit information is stored in the buffer before it is written to the audit file. When the buffer reaches the size you specify, it transfers its contents into the audit file.
    - If you enter 0 in this field, audit information is written directly to the audit file.
14. In the File Rotation Interval (seconds) field, type a number representing the number of seconds that this audit file can exist.

    The default is 0. A setting of 0 means that the audit file never times out, and audit information continues to be added to the file.

    When the limit is reached, this file is rotated, which means it is closed, stamped with the date and time, and a new audit file with the original name is created.
15. In the Engine Configuration Refresh Period (seconds) field, type a value in seconds to indicate the frequency with which configuration changes to this server take place.

    The default is 14400. A setting of 14400 means the audit file name and related parameters are refreshed once every 4 hours. If set to 0, they are never refreshed.

They are loaded when the server comes up and remain the same while the server is up.

The changes are implemented with the frequency you indicate in this field. For example, if you type 600 seconds, configuration changes are implemented within 10 minutes.

For more information, see "Modifying Access Server Details" on page 3-9.

Note: Changes you make to this parameter do not take effect until the previous Engine Config Refresh Period has expired.

16. In the URL Prefix Reload Period (seconds) field, type a number representing the frequency with which new URLs are recognized by this Access Server.

    The default is 7200.

    For example, if you type 600 in this field (600 seconds = 10 minutes), URLs are reloaded from the directory server every 10 minutes. This is helpful in cases where a particular URL Prefix cache flush request did not reach an Access Server.

17. In the Password Policy Reload Period (seconds) field, type a number representing the number of seconds that specifies the reload interval for the password policies.

    The default is 7200.

18. In the Maximum Elements in User Cache field, type a number representing the number of authenticated users that can be saved in the Access Server's cache.

    The default is 10000. A value of -1 disables the cache.

    When the maximum is reached, the newest user activity is added to the cache, and the oldest is deleted.

19. In the User Cache Timeout (seconds) field, type a number representing the number of seconds that entries remain in the user cache until they are purged.

    The default is 1800 (30 minutes). Setting the timeout to 0 means that the cache element never expires.

    After the timeout on cached user entry has expired, the Access Server goes to the directory server to get user profile data needed during authentication actions and authorization actions.

20. In the Maximum Elements in Policy Cache field, type a number representing the number of items and activities associated with activities within policy domains, such as URLs mapped to specific rules, that can be cached.

    The default is 100000.

    When the maximum is reached, the newest user activity is added to the cache, and the oldest is deleted.

21. In the Policy Cache Timeout field, type a number representing the number of seconds that entries about policies can last before they are purged.

    The default is 7200 (two hours).

22. In the SNMP State field, click On to enable SNMP or click Off to disable SNMP.

23. In the SNMP Agent Registration Port field, enter the port number for the SNMP Agent.

24. Indicate if the Session Token Cache is Enabled or Disabled.

If it is enabled, the Access Server stores the session token in the cache. For a description of the session token, see "Single Sign-On Cookies" on page 7-2.

25. Indicate the maximum number of elements that can be stored in the session token cache.

    You may need to tune this number periodically because this cache can become very large.

26. Click Save to save this new Access Server (or click Cancel to exit the page without saving).

27. Repeat the steps in this procedure for each Access Server that you want to install in your e-business infrastructure.

Now that you have created an Access Server instance, you can install this Access Server. When installing, use the Name, Hostname, and Port number information you typed in this page.

See the *Oracle Access Manager Installation Guide* for information on installing an Access Server.

Configuring a Directory Server Profile for the Access Server

A default directory profile is created for the Access Server during Access Server installation. For information on how to view or modify this profile, see information on directory profiles in the *Oracle Access Manager Identity and Common Administration Guide*.

If you install multiple Access Server instances, each server uses the same default directory server profile. If you modify a shared directory server profile for a particular Access Server instance, all of the other Access Server instances are affected. If you do not also change the profiles for these servers, you receive a warning whenever you:

- View the server configuration
- Restart the server
- Reconfigure the server

Modifying Access Server Details

Occasionally, you may need to change an Access Server's configuration settings. You can modify an Access Server instance through the Access System Console. If you change any field marked with an asterisk, you must restart the Access Server.

Note: You cannot change an Access Server's name. To give an Access Server instance a new name, you must delete and uninstall the current instance, then create a new one.

To modify an Access Server

1. Launch the Access System Console.
2. Navigate to Access System Configuration, then click Access Server Configuration.

   The Access Server Configuration page appears. The name, host, and port of each configured Access Server are listed on this page.
3. Click the link of the Access Server you want to modify.
4. On the Details for Access Server page, click Modify.

The Modify Access Server page appears. For details about all parameters, see "Access Server Configuration Parameters" on page 3-4.

5. Enter new values (see "Adding an Access Server Instance" on page 3-5.)
6. Select Update Cache to immediately send your changes to this Access Server's cache.
7. Click Save to save your changes (or click Cancel) and return to the previous page.

Deleting an Access Server

To remove an Access Server from your system, you must first delete its configured instance, then uninstall the Access Server.

When you delete an Access Server, all AccessGate instances that are configured to send requests to the server are automatically notified. Before deleting an Access Server, make sure that all AccessGate instances are configured to send requests to at least one other Access Server.

To delete an Access Server

1. Launch the Access System Console.
2. Click Access System Configuration, then click Access Server Configuration.

   The existing Access Servers are listed on the page.
3. Select the server you want to delete.
4. Click Delete.

   You are prompted to confirm your decision.
5. Click OK to delete the instance (or click Cancel to stop the deletion).
6. Uninstall the Access Server.

Clustering Access Servers

In large Oracle Access Manager implementations, there can be thousands of AccessGates. Whenever a new Access Server is added, the administrator must manually configure all the AccessGates to communicate with the Access Server. In addition, the administrator must also configure failover and load balancing for the new Access Server, as described in the *Oracle Access Manager Deployment Guide*.

Grouping Access Servers into clusters reduces the time needed to manage these tasks, because Oracle Access Manager automatically performs some of the configuration tasks. After you create a cluster, you add Access Servers to it and then associate one or more AccessGates with the cluster. Oracle Access Manager automatically configures all the AccessGates associated with the cluster to communicate with all the Access Servers in the cluster.

Managing Access Server Clusters

If you are a Master Access Administrator or a Delegated Access Administrator with appropriate rights to manage Access Server clusters, you can:

- Add an Access Server to multiple clusters.
- Associate multiple AccessGates with a cluster.
- Associate multiple clusters with an AccessGate.

Oracle Access Manager dynamically configures failover and load balancing for all the servers in a cluster and ensures that requests are routed to those Access Servers with the lightest load. For details about configuring failover, see the *Oracle Access Manager Deployment Guide*.

> **Note:** All Access Servers in a cluster (and associated WebGates and AccessGates) must have the same transport security mode and Access Management Service setting. Automatic cache flush requires an AccessGate and the Access Management Service set to On.

To add an Access Server cluster

1. Launch the Access System Console.
2. In the Access System Configuration page, click Access Server Clusters.

   The existing Access Server clusters are listed on the page.
3. Click Add.

   The Create a New Cluster of Access Servers page appears.
4. Enter a unique name for the cluster.
5. Select a transport security mode for the cluster.

   Open mode is the default. You can select Open, Simple, or Cert. All Access Servers in a cluster must have the same transport security mode.
6. Specify the Access Management Service state.
   - On—Click the On button.
   - Off—By default, it is turned Off.
7. Click Next to go to the next page (or click Cancel if you do not want to save the cluster).
8. On the next page, select the Access Server that you want to add to the cluster by clicking an Access Server in the list.

   The Access System only displays those Access Servers that have the same transport security mode and Access Management Service state (also known as Policy Manager API Support Mode) as the one specified for the cluster.
9. Click the double right arrow (>>) button to add the Access Server to the cluster.

   To remove an Access Server from the cluster, select it in the Access Servers in Clusters box and click the double left arrow (<<) button.
10. Click Save to save your changes, or click Cancel if you do not want to save your changes.

    Click Back to return to the first page.

> **Note:** If you click Back and change the transport security mode or the Access Management Service state, then click Next, you must re-select Access Servers with the new security mode or Access Management Service state.

To view or modify an Access Server cluster

1. Launch the Access System Console and click Access Server Clusters.

The existing Access Server clusters are listed on the page.

2. Click a cluster to view its details.

   The Details for Access Server Cluster page appears. The details of Access Servers in the cluster are listed.

3. Click Modify to modify a cluster's details.

   The Modify Cluster page appears.

4. You can add or delete Access Servers.

   - To add an Access Server to the cluster, select the server from the Available Access Servers list and click the >> button to add it to the cluster.
   - To remove an Access Server from the cluster, select the server in the Access Servers in Cluster box and click the << button to remove it from the cluster.

5. Click Save (or click Cancel if you do not want to save your changes).

Managing Access Servers from the Command Line

You can perform an automated installation of the Access Server using a file that contains installation parameters and values. This is called installing in silent mode. Silent mode permits installation without user intervention.

See Also: "Using the ConfigureAAAServer Tool"

To install an Access Server in silent mode

1. At the command line, enter the following command:

   ```
   configureAAAServer.exe install install_dir -S -f aaa_input.xml
   ```

   where *aaa_input.xml* is a file that contains installation parameters and values.

   The Access System provides two sample input files named aaa_input.xml and silent-mode-sample-AAA-Input.xml. The files are located in:

   ```
   AccessServer_install_dir\access\oblix\tools\configureAAAServer
   ```

   where *AccessServer_install_dir* is the directory in which the Access Server is installed. For information on silent mode, installation, see the *Oracle Access Manager Installation Guide*.

Using the ConfigureAAAServer Tool

You can perform Access Server-related administration tasks through a command-line tool called configureAAAServer. This tool can be used in both Windows and Solaris installations.

Commands that you can use with configureAAAserver tool:

- install
- reconfig
- chpasswd
- remove

Windows Systems—Use configureAAAServer.exe. Use the remove and install commands to remove or re-install an Access Server Service.

UNIX Systems—Use the **`start_configureAAAServer`** script to invoke the configureAAAServer tool. To see the options, you can run this tool without any options.

To access the configureAAAserver tool

1. Navigate to the folder where configureAAAserver is located.

   The default location is:

   AccessServer_install_dir\access\oblix\tools\configureAAAServer

   Note: On UNIX systems, use start_configureAAAServer.

2. Use the configureAAAserver tool in a procedure, as needed.

To reconfigure an Access Server

1. Navigate to the folder where configureAAAserver is located.

   For example:

   AccessServer_install_dir\access\oblix\tools\configureAAAServer

2. Run the following executable:

   configureAAAServer reconfig *AccessServer_install_dir*

3. Specify the following when prompted:
   - The transport security mode in which you want Access Server to run
   - The transport security mode in which the directory server is running
   - The host computer on which the directory server resides
   - The port number on which the directory server listens
   - The bind DN of the directory server
   - The password of the directory server
   - The directory server to which you are connecting
   - The location where configuration data is stored
   - The configuration DN
   - The policy base
   - The Access Server ID

   See "Configuring the Directory Server" on page 2-8 for information on directory server configuration.

4. Restart the Access Server.

To modify common parameters

1. Navigate to the folder where configureAAAserver is located.

   The default location is:

   AccessServer_install_dir\access\oblix\tools\configureAAAServer

where *AccessServer_install_dir* is the directory where the Access Server was installed.

2. Run the following executable:

   ```
   configureAAAServer reconfig AccessServer_install_dir
   ```

   You are then asked if you want to specify failover information for configuration or policy data.

3. Select Yes (Y).
4. Specify whether the data is stored in the configuration directory tree, or the policy tree.

   The following options appear:

   a. Add a failover server
   b. Modify a failover server
   c. Delete a failover server
   d. Modify common parameters
   e. Quit

5. Select Modify common parameters.
6. Specify values for the following common configuration parameters as needed:

   a. **Maximum Connections**—The maximum number of connections that an Access Server can establish with the associated directory servers for load balancing.
   b. **Sleep For (seconds)**—The frequency with which the Access Server checks its connections to the directory server. For example, if you set a value of 60 seconds, the Access Server checks its connections every 60 seconds from the time it comes up.
   c. **Failover Threshold**—The number representing the point when the Access Server opens a new connection to a directory server. For example, if you type 3 in this field, and the number of connections from the Access Server to the directory server falls to 2, a new connection is opened between the Access Server and the configured directory servers.
   d. **Maximum Session Time**—The maximum time period that a session between an Access Server and a directory server is valid.

      For more information, see the *Oracle Access Manager Deployment Guide*.

7. Select Quit to exit.

   You are prompted to commit the changes.

8. Select Y to commit your changes (or select N to cancel your changes).

To remove an Access Server service

1. Navigate to the folder where configureAAAserver is located.

   The default location is:

   ```
   AccessServer_install_dir\access\oblix\tools\configureAAAServer
   ```

   Note: On UNIX systems, use start_configureAAAServer.

2. From the command line, run the following executable:

   `configureAAAServer remove AccessServer_install_dir serviceName`

   where *AccessServer_install_dir* is the directory in which the Access Server was installed and *serviceName* is the name of a service such as AccessManager_AccessServer.

   A message appears stating that the registry entries are being removed. This confirms that the Access Server has been removed.

   Note: The serviceName variable is applicable only for Microsoft Windows. The serviceName is the name you specify for the Access Server on the Access System Console.

To re-install an Access Server service

1. Navigate to the folder where configureAAAserver is located, for example:

   AccessServer_install_dir\access\oblix\tools\configureAAAServer

   Note: On UNIX systems, use start_configureAAAServer.

2. From the command line, run the following executable:

   configureAAAServer install *AccessServer_install_dir* serviceName

   where *AccessServer_install_dir* is the directory in which the Access Server was installed and *serviceName* is the name of a service such as AccessManager_AccessServer.

3. Specify the following:
   - Whether or not you want to reconfigure the Access Server
   - The transport security mode for the Access Server
   - The transport security mode for the Oracle Access Manager directory server
   - The host computer on which the directory server resides
   - The port number on which the directory server listens
   - The bind DN of the directory server
   - The password of the directory server
   - The directory server to which you are connecting
   - The configuration DN
   - The location of the policy data
   - The policy base
   - The Access Server ID
4. Note the name of the Access Server service.

   A message appears stating that the Access Server has been successfully installed.
5. Start the Access Server from the Windows Control Panel services.

Setting the Number of Queues from the Command Line

Requests are queued as they are sent to an Access Server. A thread processes each request. For example, if you have two request queues and 60 threads, the Access Server spawns 120 threads.

You cannot specify the number of queues in the Access System Console. When you configure an Access Server, however, you specify the number of threads in the Number of Threads field. The default setting is 60.

Use a command line entry to specify the number of queues each Access Server can support. Keep the number of queues in balance with the number of threads. Typically, one queue is adequate for each WebGate.

A command is available with Solaris, Windows 2000, or Windows NT. On Solaris, you open a shell window to use this command. On Windows, use the Start Parameter field in the Services window to use this command.

> **Tip:** Additional information on threads and queues is provided in the *Oracle Access Manager Deployment Guide*.

To set the number of queues on Solaris

1. Open a shell window.
2. At the command line, enter the following command:

   ```
   start_access_server -QN
   ```

   where N is the number of queues.

To set the number of queues on Windows 2000

1. Navigate to **Start**, **Programs**, **Administrative Tools**, **Services**, and then **COREidAAAServerID**.

   where *ID* is the name of the Access Server.
2. Right-click COREidAAAServerID and select Properties.

   The Properties window appears.
3. To specify the number of queues, in the General tab, enter

   -QN

   where *N* is the number of queues in the Start Parameter field.

To set the number of queues on Windows NT

1. Navigate to **Start**, **Control Panel**, **Services**, and then **COREidAAAServerID**

   where *ID* is the name of the Access Server.
2. Right-click COREidAAAServerID and select Properties.

   The Properties window appears.
3. To specify the number of queues, in the General tab, enter

   -QN

   where *N* is the number of queues in the Start Parameter field.

Configuring AccessGates and WebGates

At least one Access Server and one AccessGate must be configured and installed for the Access System to run. The AccessGate that you configure can be a WebGate, which is an out-of-the-box AccessGate for http resources. The Access Server must be installed before you add the AccessGate profile and install the AccessGate.

The rest of this section discusses the following topics:

- Viewing AccessGate Profiles
- AccessGate Configuration Parameters
- Adding an AccessGate
- Modifying an AccessGate
- Deleting an AccessGate

Task overview: Configuring an AccessGate

1. Create one or more AccessGate instances in the Access System Console, as described in "Adding an AccessGate" on page 3-24.
2. Associate each AccessGate instance with an Access Server, as described in "Associating AccessGates and WebGates with Access Servers" on page 3-41.
3. Install an AccessGate for each instance that you created in the Access System Console, as described in the *Oracle Access Manager Installation Guide*.
4. Change the AccessGate settings as needed, as described in "Modifying a WebGate" on page 3-36.

> **Note:** When configuring a WebGate, Oracle recommends setting DenyOnNotProtected to true to ensure that Web content is protected. See "Denying Access to All Resources by Default" on page 3-50 for details.

Viewing AccessGate Profiles

You can view existing AccessGate profiles in the Access System Console by searching for the AccessGate.

You use a Search page to search for an AccessGate by any of its attributes. Depending on the attribute you select, the search conditions and values vary. For example, if you select Security as your search attribute, Equals may be the condition that is displayed in the list, and the possible values would be one of the available security modes. The system then displays the existing AccessGates that have been configured with the specified security mode.

The search attributes, conditions, and values for an AccessGate are listed in Table 3–2.

Table 3–2 AccessGate Search Attributes, Conditions, and Values

Attribute	Condition	Input Type	Description
Name	▪ Contains ▪ Starts with ▪ Ends with ▪ Equals	Text box	Searches by AccessGate name.

Table 3–2 (Cont.) AccessGate Search Attributes, Conditions, and Values

Attribute	Condition	Input Type	Description
Host Name	▪ Contains ▪ Starts with ▪ Ends with ▪ Equals	Text box	Searches for AccessGates that are installed on the specified host computer.
Description	▪ Contains ▪ Starts with ▪ Ends with ▪ Equals	Text box	Searches for AccessGates with a description field that contains a matching string.
Security Mode	Equals	Radio Button: Open, Simple, and Cert	Searches for AccessGates based on the transport security mode configured for them.
Policy Manager API Support Mode	Equals	Radio Button: On or Off	Searches for AccessGates based on whether the Access Management Service is On or Off. Set the value to On to list AccessGates that have this enabled. Set the value to Off to list AccessGates for which this is disabled. See also Table 3–3, " AccessGate Configuration Parameters".
State	Equals	Radio Button: Enabled and Disables	Searches for AccessGates that are enabled or disabled.

To view AccessGate profiles

1. Launch the Access System Console and click **Access System Configuration**, and then click **AccessGate Configuration**.

   The Search for AccessGates page appears.

   The Search list contains a selection of attributes that can be searched, as described in Table 3–2.

   The remaining fields allow you to specify search criteria that are appropriate for the selected attribute.

2. Select the search attribute and condition from the lists (or click the All button to find all AccessGates).

3. Click Go.

   The search results are displayed on the page.

4. Click an AccessGate's name to view its details.

   The configuration details of the AccessGate appear.

To view the AccessGates associated with an Access Server

1. Launch the Access System Console and click **Access System Configuration**, then click **Access Server Configuration**.

   The Access Server Configuration: List all Access Servers page appears.

2. Click the link for the desired Access Server.

   The Details for Access Server page appears.

3. Click the View Associated AccessGates button at the bottom of the Details for Access Server page.

   The AccessGates associated with server page appears.

4. Check Primary Server to view AccessGates for which the Access Server is configured as a primary server.

   Check Secondary Server to view AccessGates for which the Access Server is configured as a secondary server.

   Select All to list all the specified AccessGates, or enter a number to specify the number of search results you want displayed on the page.

5. Click Go to display the search results.

   The details of the AccessGates associated with the Access Server are displayed on the page.

6. If there are multiple pages, click Next to go to the next page, or click Previous to go back to the previous page.

7. Click Back to return to the Access Server page.

AccessGate Configuration Parameters

All Access Servers (in a cluster, or not), and associated WebGates and AccessGates, must have the same transport security mode and Access Management Service setting.

Table 3–3 summarizes the AccessGate configuration parameters:

Table 3–3 AccessGate Configuration Parameters

Field	Description
AccessGate Name	Name of the AccessGate.
Description	Additional information to identify this AccessGate.
State	Whether or the AccessGate is enabled or disabled.
Hostname	Name of the computer hosting the AccessGate.
Port	Optional, for information only. Identifies the Web server port protected by the AccessGate when deployed as a WebGate. This field should be empty for an AccessGate configuration that supports an application using the Access Manager SDK.
AccessGate Password	Optional, unique password for the AccessGate, created when you defined the AccessGate in the Access System Console. When the AccessGate connects to an Access Server, it uses the password to authenticate itself to the Access Server. This prevents unauthorized AccessGates from connecting to Access Servers and obtaining policy information.
Debug	Turns debugging on or off.
Maximum User Session Time (seconds)	Maximum amount of time in seconds that a user's authentication session is valid, regardless of their activity. At the expiration of this session time, the user is re-challenged for authentication. This is a forced logout. Default = 3600 A value of 0 disables this timeout setting.

Table 3–3 (Cont.) AccessGate Configuration Parameters

Field	Description
Idle Session Time (seconds)	Amount of time in seconds that a user's authentication session remains valid without accessing any AccessGate protected resources. Default = 3600 A value of 0 disables this timeout setting.
Maximum Connections	The maximum number of connections that this AccessGate can establish with Access Servers. This number must be the same as or greater than the number of Access Servers that are actually associated with the WebGate. Default = 1
Transport Security	Level of transport security to and from the Access Server, can be set to: ▪ Open--No transport security ▪ Simple--SSL v3/TLS v1.0 secure transport using dynamically generated session keys ▪ Cert--SSL v3/TLS v1.0 secure transport using server side x.509 certificates
IPValidation	IP address validation is specific to WebGates and is used to determine whether a client's IP address is the same as the IP address stored in the ObSSOCookie generated for single sign-on. See "Configuring IP Address Validation for WebGates" on page 3-37 for details.
IPValidationException	IPValidationException is specific to WebGates. This is a list of IP addresses that are excluded from IP address validation. It is often used for excluding IP addresses that are set by proxies. See "Configuring IP Address Validation for WebGates" on page 3-37 for details.
Maximum Client Session Time (hours)	Connection maintained to the Access Server by the AccessGate. If you are deploying a firewall (or another device) between the AccessGate and the Access Server, this value should be smaller than the timeout setting for the firewall. Default: 24 hours.
Failover Threshold	Number representing the point when this AccessGate opens connections to Secondary Access Servers. If you type 30 in this field, and the number of connections to primary Access Servers falls to 29, this AccessGate opens connections to secondary Access Servers.

Table 3–3 (Cont.) AccessGate Configuration Parameters

Field	Description
Access Server Timeout Threshold	Applicable only to WebGates. Number in seconds to wait for a response from the Access Server. If this parameter is set, it is used as an application TCP/IP timeout instead of the default TCP/IP timeout. For example, suppose a WebGate is configured to talk to one primary Access Server and one secondary Access Server. If the network wire is pulled from the primary Access Server, the WebGate waits for the TCP/IP timeout to learn that there is no connection to the primary Access Server. The WebGate tries to reestablish the connections to available servers starting with the primary Access Server. Again, the WebGate waits for the TCP/IP timeout to determine if a connection can be established. If it cannot, the next server in the list is tried. If a connection can be established to another Access Server (either a primary or secondary), the requests are re-routed. However this can take longer than desired. Rather than rely on the default TCP/IP timeout, you can specify the Access Server Timeout Threshold in the Access System Console, Access System Configuration, AccessGate Configuration. The default value of -1 means the default network TCP/IP timeout is used. A typical value for this parameter is between 30 and 60 seconds. If set to a very low value, the socket connection can be closed before a reply from Access Server is received, resulting in an error. When finding new connections, WebGate checks the list of available servers in the order specified in its configuration. If there is only one primary Access Server and one secondary Access Server specified, and the connection to the primary Access Server times out, the WebGate still tries the primary Access Server first. As a result, the WebGate cannot send requests to an Access Server for a period greater than twice the setting in the Access Server Timeout Threshold. If the Access Server takes longer to service a request than the value of the timeout threshold, the WebGate abandons the request and retries the request on a new connection. Note that the new connection that is returned from the connection pool can be to the same Access Server, depending on your connection pool settings. Also, other Access Servers may also take longer to process the request than the time specified on the threshold. In these cases, the WebGate can continue to retry the request until the Access Servers are shut down. You can control the number of retry attempts with the user-defined parameter client_request_retry_attempts. See "Configuring User-Defined AccessGate Parameters" on page 3-29 for details.
Sleep For (seconds)	Number in seconds that represents how often this AccessGate checks its connections to Access Servers. For example, if you set a value of 60 seconds for the Sleep For parameter, AccessGate checks its connections every 60 seconds from the time it comes up.

Table 3–3 (Cont.) AccessGate Configuration Parameters

Field	Description
Maximum Elements in Cache	Number of elements maintained in the cache. Cache elements are the following: ■ URLs—The URL cache maintains information about a URL, including if it is protected and the authentication scheme used if it is protected. ■ Authentication schemes—This cache stores authentication scheme information for a specific authentication scheme ID. The value of this setting refers to the maximum consolidated count for elements in both of these caches. Default = 10000
Cache Timeout (seconds)*	Amount of time cached information remains in the AccessGate cache when neither used nor referenced. Default = 1800
Impersonation Username	The name of the trusted user that you created to be user for impersonations. The value should contain the domain name, for example, testdomain@user.net. You specify the trusted username here to bind it to this AccessGate (WebGate) so that the AccessGate can use it for impersonation. For information about impersonation and explanation of how to create a trusted user for impersonation, see the *Oracle Access Manager Integration Guide*.
Impersonation Password	The password for the trusted user to be used for impersonation. You must enter this password twice; that is, you are asked to re-type it.
Access Management Service Also known as Policy Manager API Support Mode	This parameter determines whether the Access Management Service is On, or Off. It is Off by default. The Access Management Service should be: ■ On if the Access Server is associated and communicating with AccessGates (which communicate using APIs in the SDK). AccessGates (and associated Access Servers and clusters) require Access Management Service On. AccessGate is used for features such as Identity to Access Server Cache Flush and Self Registration Auto Login features. ■ Off if only WebGates are associated with the Access Server. WebGates do not require the Access Management Service. If only WebGates are associated with an Access Server, both can have the Access Management Service Off (the default). Note: All Access Servers (in a cluster, or not) and associated AccessGates or WebGates must have the same transport security mode and Access Management Service state.

Table 3–3 (Cont.) AccessGate Configuration Parameters

Field	Description
Primary HTTP Cookie Domain	This parameter describes the Web server domain on which the AccessGate is deployed, for instance, .mycompany.com. You must configure the cookie domain to enable single sign-on among Web servers. Specifically, the Web servers for which you configure single sign-on must have the same Primary HTTP Cookie Domain value. The Access System uses this parameter to create the ObSSOCookie authentication cookie. This parameter defines which Web servers participate within the cookie domain and have the ability to receive and update the ObSSOCookie. The WebGate cookie domain parameter is not used to populate the ObSSOCookie; rather it defines which domain the ObSSOCookie is valid for, and which Web servers have the ability to accept and change the ObSSOCookie contents.
Preferred HTTP Host	This required field determines how the host name appears in all HTTP requests as they attempt to access the protected Web server. The host name in the HTTP request is translated into the value entered into this field regardless of the way it was defined in a user's HTTP request. There are special considerations for field if you use virtual hosting. See "Configuring Virtual Web Hosting" on page 3-49 for details. Except for cases where virtual hosting is required, Oracle recommends that you always specify a value in the Preferred HTTP Host field for each WebGate, and ensure that it maps to a host name in the Host Identifiers field. You use the host identifiers when defining policies. See also, "Configuring Preferred HTTP Hosts, Host Identifiers, and Virtual Web Hosts" on page 3-45. New parameters can be added to Policy Manager globalparams.xml, that help monitor the Preferred HTTP Host field in a WebGate configuration in the Access System Console. ■ `AllowEmptyPreferredHost` ■ `PreferredHostValidityCheckEnabled` For more information, see "Invalid or Missing Preferred HTTP Host Identifier in WebGate Profile" on page E-13. For parameters, see the *Oracle Access Manager Customization Guide*.
DenyOnNotProtected	This setting applies only to WebGates. The default value for this parameter is false. In this case, there is no protection, and access is enabled. More importantly, access may be granted inadvertently. For example, if someone attempts to access a resource using the decimal value of an IP address in the URL, access may be granted unless the host identifier includes this representation of the address. Setting DenyOnNotProtected to true is the most secure way to protect Web server content. If you set DenyOnNotProtected to true, all requests for Web pages on the Web server protected by the WebGate are denied unless access is explicitly allowed by a policy. When this is set to true, you need to create an anonymous authentication method and allow access to content using an anonymous access policy. For information describing how to use the DenyOnNotProtected switch, see "Denying Access to All Resources by Default" on page 3-50.

Table 3–3 (Cont.) AccessGate Configuration Parameters

Field	Description
CachePragmaHeader and CacheControlHeader	These settings apply only to WebGates and control the browser's cache. By default, CachePragmaHeader and CacheControlHeader are set to no-cache. This prevents WebGate from caching data at the Web server application and the user's browser. However, this may prevent certain operations such as downloading PDF files or saving report files when the site is protected by a WebGate. You can set the Access Manager SDK caches that the WebGate uses to different levels. See http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html section 14.9 for details. All of the cache-response-directives are allowed. For example, you may need to set both cache values to public to allow PDF files to be downloaded.
LogOutUrls	This setting applies only to WebGates. The LogOutUrls parameter enables you to configure one or more specific URLs that log out a user. You can provide a portal logout URL for this parameter. For example, suppose that a finance portal is protected by a Webgate, and the portal logout page is finance_exit.html. If you add finance_exit.html as a LogOutUrls entry, users who are logged out of the finance portal are also logged out of Oracle Access Manager. Oracle recommends that you provide the SSO Logout URL as one of the values for the LogOutUrls parameter. This will add consistency and enable a global logout value for multi-domain single sign-on implementations. See "Configuring Logout for an Identity System Resource" on page 3-29 for details. Also, if you have configured a WebGate to protect a WebPass or Policy Manager, the LogOutUrls parameter must contain the SSO Logout URL values. This is required since the WebPass or Policy Manager logout button redirects the user to the SSO Logout URL.
User-Defined Parameters	These settings apply only to WebGates. As of Oracle Access Manager 10.1.4, the WebGateStatic.lst file is no longer present. Some of the parameters that were set in WebGateStatic.lst are now specific data entry fields in the AccessGate configuration page. Others have been made available as user-defined parameters on this page. See "Configuring User-Defined AccessGate Parameters" on page 3-29 for details.

Adding an AccessGate

You must add an AccessGate instance in the Access System Console before installing the AccessGate. Required parameters before installation include the AccessGate name, hostname, transport security mode, and the preferred HTTP host. All other AccessGate parameters can be configured after installation, as described in the following sections.

See "AccessGate Configuration Parameters" on page 3-19 for details about all parameters. See the *Oracle Access Manager Installation Guide* for details about installing an AccessGate.

Note: Once you assign and save an AccessGate name, you cannot change the name. To rename an AccessGate, you must delete and uninstall the instance, then create a new AccessGate.

To create an AccessGate instance

1. Launch the Access System Console and click Access System Configuration.
2. Click Add New AccessGate in the left navigation pane.

   The Add New AccessGate page appears.
3. Fill in the form as follows:
   - **AccessGate Name**: Type the name of this AccessGate instance. Type an alphanumeric string without spaces. Note that an AccessGate and an Access Server cannot have the same name.
   - **Description**: Type a summary that will help you identify this AccessGate later on.
   - **Hostname**: Type the name or IP address of the server hosting this AccessGate.
   - **Port**: Type the Web server port protected by the AccessGate when deployed as a WebGate.

     This field is optional and is provided for informational purposes only. It is recommended that you enter the Web server port number for a WebGate. Other AccessGates may or may not use a port.

     This field must be empty when the AccessGate describes the configuration in support of an application using the Access Manager SDK.
   - **AccessGate Password**: Type an alphanumeric string to represent a password for this AccessGate.

     The AccessGate uses this password to identity itself to an Access Server. If you provide a password here, you must enter the same password when reconfiguring the AccessGate and during AccessGate installation.

     Note: If this AccessGate is a WebGate, this password must be the same one specified when you installed the WebGate.
   - **Re-type AccessGate Password**: Re-type the password.

     If the entries in these two steps do not match, when you click Save, an error message appears, and you must repeat this process.
4. **Debug**: The Debug field is relevant only for WebGates.

   In this field:
   - Click On to write debug messages between the AccessGate and Access Server to the standard out for most platforms. Note that on IIS this information is not written to the standard out because the IIS server runs as an NT service.
   - Click Off if you do not want to capture this information.

Important: Capturing debug messages records user passwords, a potential security problem, and causes the Access Server log file to grow rapidly. Debugging should only be turned on when diagnosing a problem.

5. Set the following values to determine the user session time and time out:

   Maximum user session time: Type the maximum amount of time, in seconds, that a user's authentication session is valid, regardless of their activity. At the expiration of this session time, the user is re-challenged for authentication. This is a forced logout.

   Idle Session Time: Type the amount of time in seconds that a user's authentication session remains valid without accessing any AccessGate protected resources. The default is 3600.

6. **Maximum Connections**: Type the maximum number of connections this AccessGate can establish with associated Access Servers. This number may be greater than the number allocated at any given time.

   The default is 1.

7. **Transport Security**: Select a method for encrypting messages between this AccessGate and the Access Servers it is configured to talk to.

   For AccessGates and Access Servers that are configured to communicate with each other, be sure to choose the same encryption method.

   Your choices are:

   – Open
   – Simple
   – Cert

   For a description of configuring transport security modes, see the *Oracle Access Manager Identity and Common Administration Guide* or see the table of AccessGate configuration parameters in "AccessGate Configuration Parameters" on page 3-19.

   Note: If you want to change an AccessGate mode from simple or cert to open, you must move the /oblix/config/simple directory (if in simple mode) or the /oblix/config/*.pem files (if in cert mode) to a new folder. Then you must run the configureAccessClient program.

8. Perform the following steps:

   Supply **IPValidation** for a WebGate to determine if a client IP address is the same as the IP address stored in the ObSSOCookie generated for single sign-on.See "Configuring IP Address Validation for WebGates" on page 3-37 for details.

   Supply values for **IPValidationException** if you want to supply IP addresses to exclude from IP address validation. See "Configuring IP Address Validation for WebGates" on page 3-37 for details.

9. **Maximum Client Session Time**: Specify the connection maintained to the Access Server by the AccessGate.

   If you selected Open in the Transport Security field, this field is ignored.

If the Maximum Client Session Time is 0, the AccessGate establishes a new connection to the Access Server for each request that it makes to the Access Server. There may be multiple AccessGate requests for each user request to the AccessGate.

The default is 24 hours. This value must be larger than the Sleep For parameter. Using the same session key for longer than 24 hours can make your system vulnerable to attack.

10. **Failover Threshold**: Type the number representing the point when this AccessGate opens connections to secondary Access Servers. If you type 30 in this field, and the number of connections to primary Access Servers falls to 29, this AccessGate opens connections to secondary Access Servers.

    You can type a number ranging from 1 to the total number of primary servers. If you do not type a value, the number of maximum connections is used.

    For details about configuring failover, see the *Oracle Access Manager Deployment Guide*.

11. **Access Server timeout threshold**: Applicable only to WebGates. Specify the time (in seconds) during which the AccessGate must wait for a response from the Access Server. If this parameter is set, it is used as an application TCP/IP timeout instead of the default TCP/IP timeout.

12. **Sleep For (seconds)**: Type a number (in seconds) that represents how often this AccessGate checks its connections to Access Servers.

    If a connection to an Access Server is broken, but the AccessGate finds that the Access Server is now up, it tries to reconnect to that server.

    The default is 60 seconds. The shorter the value, the quicker AccessGate can reestablish a connection to an Access Server that has come back up. But the overhead for checking connections is higher.

    An entry in this field does not affect failover to other servers, which is always immediate when needed.

13. Fill in caching details as follows:

    - **Maximum elements in cache**: Enter the maximum number of elements that can be maintained in the URL and authentication scheme caches. This number represents the grand total elements in both caches.

      The default is 10000.

    - **Cache timeout (seconds)**: Specify the time period during which cached information remains in the AccessGate cache when neither used nor referenced.

      The default is 1800.

14. Complete details for impersonation for IIS only, as follows:

    - **Impersonation username**: Specify the name of the trusted user that you created to be used for impersonations. The name should contain the domain name, for example, testdomain@user.net. You specify the user name here to bind it to this AccessGate.

    - **Impersonation password**: Enter the password for the impersonation user name.

    - **Re-type impersonation password**: Enter the password for the impersonation user name.

15. For **Access Management Service**, set the state to On or Off.

    See "AccessGate Configuration Parameters" on page 3-19 for details.

16. Continue completing the information as follows:

    - **Primary HTTP Cookie Domain**: Type the AccessGate's domain name.

      For example,

      ```
      .yourcompany.com
      ```

      Note: The dot (".") in the initial position of the domain name is required. See "Configuring Single Sign-On" on page 7-1 for information on how the primary HTTP cookie domain is used.

    - **Preferred HTTP Host**: Specify how the hostname appears in all HTTP requests as they attempt to access the protected Web server. The hostname within the HTTP request is translated into the value entered into this field regardless of the way it was defined in a user's HTTP request.

      The Preferred Host function prevents security holes that can be inadvertently created if a host's identifier is not included in the Host Identifiers list. However, it cannot be used with virtual Web hosting. For virtual hosting, you must use the Host Identifiers feature.

      Unless you disable the preferred HTTP host feature, you must enter one of the variations entered in the Host Identifiers feature to ensure that single sign-on works properly. See "Configuring Preferred HTTP Hosts, Host Identifiers, and Virtual Web Hosts" on page 3-45 and "Configuring Virtual Web Hosting" on page 3-49 for details.

      Note: If you are configuring a WebGate, and your browser is Internet Explorer, do not use the number 80. Using 80 as a port number can lead to operational errors.

17. **DenyOnNotProtected**: This setting applies only to WebGates. Specify true to deny all access to resources on the Web server protected by WebGate unless access is allowed by a policy. A value of true requires that you set an anonymous authentication method and allow access to content using an anonymous access policy. See "Denying Access to All Resources by Default" on page 3-50 for details.

18. Fill in the following values for WebGates only after reviewing information in "AccessGate Configuration Parameters" on page 3-19:

    CachePragmaHeader:

    CacheControlHeader:

19. **LogOutUrls**: This setting applies only to WebGates.

    To ensure that users log out completely from Identity and Access applications when they click Logout, set the value of this parameter to the value of the SSO logout URL. See "AccessGate Configuration Parameters" on page 3-19 and "Configuring Logout for an Identity System Resource" on page 3-29 for details.

20. **User-Defined Parameters (WebGates only)**: To configure the WebGate to work with particular browsers, proxies, and so on, you can set specific user-defined

parameters. See "Configuring User-Defined AccessGate Parameters" on page 3-29 for details.

21. Click Save to save this new instance of AccessGate (or click Cancel to return to the previous page without saving).

    Now that you have created an AccessGate instance, you can install and set up this instance. When installing, use the Name, Hostname, and Port number information you typed in this page. See the *Oracle Access Manager Installation Guide* for details.

Configuring Logout for an Identity System Resource

When the Identity System applications are protected by a WebGate, a logout button is not automatically configured for the Policy Manager and the Identity System Console. You must configure the logout button and logout URL, as explained in the following procedure.

To configure the logout button

1. As described in "Configuring a Single Sign-On Logout URL" on page 2-6, configure a URL that points to a logout page that you want to show to the user when they log out of the application.

   Note: Alternatively, you can specify the logout URL on the SSOLogoutURL parameter in the OblixBaseParams.lst file. This file is located in:
   PolicyManager_install_dir/access/oblix/apps/common/bin/

2. To ensure that the WebGate logs out the user from the Identity or Access application when they click Logout, ensure that the LogOutUrls parameter is set to the same value as the SSO Logout URL.

   See "Modifying an AccessGate" on page 3-32 for details.

Configuring User-Defined AccessGate Parameters

In earlier versions of Oracle Access Manager, a file named WebGateStatic.lst to configure various settings for a WebGate. The settings in this file have moved to the AccessGate configuration page. Some of the settings are displayed as static parameters on this page (for example, `DenyOnNotProtected`). Several of these settings now appear as user-defined parameters on the configuration page.

The AccessGate user-defined parameters address the following issues:

- Working with URLS that are encoded in UTF-8.
- Working with Microsoft Passport or Integrated Windows Authentication on IIS.
- Working with older Netscape Web servers.
- Setting the frequency with which the shared secret is updated.
- Preserving compatibility with NetPoint 5.x systems.
- Working with reverse proxies that use SSL between the client and a reverse proxy and non-SSL between the reverse proxy and the Web server.

To implement user-defined parameters, you must enter them in the AccessGate configuration page and contact Oracle for a patch for the WebGate.

The following are the AccessGate user-defined parameters:

RetainDownstreamPostData—Adding this user-defined parameter and setting the value to `true` resolves a problem that occurs when WebGate for Apache 2.0 or Apache 2.2 prevents POST data from being read by downstream applications. Form-based authentication schemes using the "passthrough" challenge-parameter and policies using the "Query String Variable(s)" option are affected.

UrlInUTF8Format—In an environment that uses Oracle HTTP Server 2, this parameter must be set to true to display latin-1 and other character sets.

UseIISBuiltinAuthentication—By default, this parameter value is false. Set UseIISBuiltinAuthentication to true only if you are using Microsoft Passport or Integrated Windows Authentication on this computer. It is used only for IIS, and is ignored if the WebGate is installed for another type of Web server.

SlowFormLogin—In some versions of the Netscape Web server, form login over https may not work as expected. In this case, the SlowFormLogin parameter eliminated the problem. This parameter is not necessary with the latest version of the Netscape Web server.

InactiveReconfigPeriod—The WebGate has an update thread that reads the shared secret from the Access Server every 1 minute when the WebGate is active. The Access Server server returns the shared secret in its own cache (the Access Server cache). By default, this value is updated every 10 minutes. For example, the Access Server reads the shared secret from the directory at an interval of 10 minutes and this cached value is returned to WebGate. You can change this setting. See "Changing the WebGate Polling Frequency" on page 3-32 for details.

In the idle state the WebGate reads the shared secret from the Access Server using the InactiveReconfigPeriod value. If this value is not set, the WebGate polls the Access Server for the shared secret value at an interval of 1 minute even though the updated shared secret value will be returned only after 10 minutes.

WaitForFailover—Used only for compatibility with NetPoint 5.x systems, this parameter has been replaced by Access Server TimeOut Threshold on the AccessGate configuration page.

Both the WaitForFailover parameter, and its replacement the *Access Server Timeout Threshold* parameter, control the TCP/IP timeout between the WebGate and the Access Servers it communicates with. The default value is "-1," which means the network default TCP/IP timeout value is used.

Ensure that both the WaitForFailover parameter and the Access Server Timeout Threshold configured on this page use the same value.

ProxySSLHeaderVar: This parameter is used when the WebGate is located behind a reverse proxy, SSL is configured between the client and the reverse proxy, and non-SSL is configured between the reverse proxy and the Web server. It ensures that URLs are stored as https rather than http. The proxy ensures that URLs are stored in https format by setting a custom header variable indicating whether it is servicing an SSL or non-SSL client connection. The value of the ProxySSLHeaderVar parameter defines the name of the header variable the proxy must set. The value of the header variable must be "ssl" or "nonssl". If the header variable is not set, the SSL state is decided by the SSL state of the current Web server.

client_request_retry_attempts—A WebGate-to-Access Server timeout threshold specifies how long (in seconds) the WebGate waits for the Access Server before it considers it unreachable and attempts the request on a new connection. If the Access Server takes longer to service a request than the value of the timeout threshold, the WebGate abandons the request and retries the request on a new connection. Note that

the new connection that is returned from the connection pool can be to the same Access Server, depending on your connection pool settings. Also, other Access Servers may also require more time to process the request than the time specified on the timeout threshold. In some cases, the WebGate can retry the request until the Access Servers are shut down. You can configure a limit on the number of retries that the WebGate performs for a non-responsive server using the client_request_retry_attempts parameter. The default value for this parameter is -1. Setting the parameter value to -1 (or not setting it at all) allows an infinite number of retries.

ContentLengthFor401Response: To set the Content-Length for all 401 responses, add the following as a user defined parameter and value: `ContentLengthFor401Response` 0. Zero (0) is the only value you can use. Any other value will be ignored. If you do not use this parameter and value, a mismatch between the content and content length might occur. This would result in either no data displayed in the browser or an error message in the browser.

SUN61HttpProtocolVersion: SUN v6.1 Web server might have a problem with redirection after reading POST data. If the connection uses the keepAlive (HTTP/1.1) protocol, data is not flushed properly. Thus, redirection might not work consistently. The SUN 6.1 Web server can be forced to use the HTTP/1.0 protocol when you assign the user-defined parameter `SUN61HttpProtocolVersion` with a value of `1.0` in the AccessGate Configuration page. Any value other than 1.0 will be ignored.

`idleSessionTimeoutLogic`: In release 7.0.4 WebGates enforce their own idle session timeout only. In 10*g* (10.1.4.0.1), behavior changed and WebGates enforced the most restrictive timeout value among all WebGates the token had visited. With 10*g* (10.1.4.3), the 7.0.4 behavior has been reinstated as the default. The default 7.0.4 behavior can be reconfigured by setting a User-Defined Parameter (`idleSessionTimeoutLogic`) in the AccessGate Configuration page of the Access System Console by setting the `idleSessionTimeoutLogic` parameter as follows:

- A value of `leastComponentIdleTimeout` instructs the WebGate to use the "most restrictive" timeout value for idle session timeout enforcement.
- A value of `currentComponentIdleTimeout` instructs the WebGates to use the "current WebGate" timeout value for idle session timeout enforcement.

See Also: "Adding an AccessGate" on page 3-24

Reducing Network Traffic Between Components

The WebGate-to-Access Server configuration polling reduces the traffic between both the WebGate and Access Server and the Access Server and the LDAP directory server.

Process overview: WebGate-to-Access Server configuration polling

1. When the WebGate is inactive for 60 seconds, it reduces the frequency of polling for its configuration information.

   The polling frequency is determined by the parameter InactiveReconfigPeriod, which is a user-defined parameter that is set in the AccessGate configuration page. See "Configuring User-Defined AccessGate Parameters" on page 3-29 for details. The value for InactiveReconfigPeriod is specified in minutes. Within ten seconds of resuming activity, the WebGate performs reconfiguration polling once a minute.

2. At startup, the WebGate checks the bootstrap configuration to see if any important parameters have changed.

   This makes the re-initialization process unnecessary in most cases and reduces the transient Access Server load.

3. WebGate and Access client configurations are cached in the Access Server.

   The default cache timeout is 59 seconds. This should cause no modifications to the system behavior on non-Apache Access clients. The Apache Web server with WebGate avoids unnecessary hits to the directory server. The caching parameters can be set in the Access Server globalparams.xml file in *AccessServer_install_dir*/access/oblix/apps/common/bin/globalparams.xml file. The parameter clientConfigCacheMaxElems sets the maximum size of the cache (default 9999). The parameter clientConfigCacheTimeout determines the maximum lifetime of any element in the cache (default 59 seconds). For more information, see the parameters chapter of the *Oracle Access Manager Customization Guide*.

There are two ways to reduce off-time network traffic between both the WebGate and Access Server and the Access Server and the LDAP directory server:

- Changing WebGate polling frequency for configuration information
- Changing the default configuration cache timeout for WebGate and Access client configurations that are cached in the Access Server, as mentioned in the previous Step 3.

Changing the WebGate Polling Frequency

One way to reduce off-time network traffic between both the WebGate and Access Server and between the Access Server and the LDAP directory server is to change the WebGate polling frequency for configuration information.

To change the configuration polling frequency

1. Add the InactiveReconfigPeriod parameter to the configuration page for this WebGate.
2. Specify the value for InactiveReconfigPeriod in minutes.

   The default is 1 minute. When the WebGate is inactive for more than 60 seconds (for example, when no authentication requests are being processed), it reduces the frequency of polling for its configuration information. Within ten seconds of resuming activity, the WebGate resumes reconfiguration polling once every minute.

   If set to -2, Webgate never polls.

   If set to a value greater than 0 it polls at the specified interval.

   If set to -1 and Webgate is inactive and has been for 1 minute, then Webgate does not poll. WebGate resumes reconfiguration polling when it returns to an active state.

Modifying an AccessGate

Occasionally you may need to modify an AccessGate's parameters. You can modify an AccessGate through the Access System Console or through a command line tool named *configureAccessGate*. Typically, you use the command line tool to change the transport security mode. This tool can be used in both Windows and Solaris installations.

Note: If you change fields marked with an asterisk(*), you must restart the server hosting this AccessGate. Once you assign and save an AccessGate name, you cannot change the name. To rename an AccessGate, you must delete and uninstall the instance, then create a new AccessGate.

To modify an AccessGate through the Access System Console

1. Launch the Access System Console and click **Access System Configuration**, and then click **AccessGate Configuration**.

   The Search for AccessGates page appears.

2. Select the search attribute and condition from the lists, or select All to find all AccessGates.

   The Search list is a selection list of attributes that can be searched, as described in Table 3–2. The remaining fields allow you to specify search criteria that are appropriate for the selected attribute.

3. Click Go.

   The search results are displayed on the page.

4. Click the name of the AccessGate you want to modify.

   The AccessGate Details page appears.

5. Click Modify.

   The Modify AccessGate page appears. You can enter new information on this page

   You cannot change an AccessGate's name. To rename an AccessGate, you must delete it from the Access System Console and then uninstall it. You then create a new AccessGate.

6. Type new values as needed.

7. Click Save to save your changes (or click Cancel to exit the page without saving).

To modify an AccessGate through the command line

1. Go to:

   AccessGate_install_dir\access\oblix\tools\configureAccessGate

   where *AccessGate_install_dir* is the directory where AccessGate is installed.

2. From the configureAccessGate directory, run the following command:

   `configureAccessGate -i AccessGate_install_dir -t AccessGate|WebGate options`

3. Specify parameters using the commands listed in Table 3–4.

Table 3–4 configureAccessGate and configureWebGate Commands

Command	Operation
`-i install_dir`	Specifies the installation directory for the AccessGate or WebGate.
`-t <AccessGate|WebGate>`	Specifies whether an operation is for AccessGate or WebGate.
`-w <AccessGate|WebGate ID>`	Identifies the name of the AccessGate or WebGate.
`-m <open|simple|cert>`	Specifies the transport security mode for an operation.
`-c <request|install>`	Specifies a certificate request or installation.

Table 3–4 (Cont.) configureAccessGate and configureWebGate Commands

Command	Operation
`-S`	Runs configureWebGate tool in silent mode without prompting for user input. If optional parameters are not present, an error message is displayed.
`-P <AccessGate|WebGate password>`	Specifies the password for the transport security certificate for AccessGate/WebGate. This value is required only if you have specified a password for the AccessGate/WebGate.
`-h Access Server Host Name`	Specifies the computer name where the Access Server installed.
`-p Access Server Port`	Specifies the port number of the computer where the Access Server is installed.
`-a Access Server ID`	Specifies the name of the Access Server.
`-r Access Server Pass Phrase`	Specifies the password for the Access Server. This is a global password and must be the same as the specified password for the AccessGate/WebGate.
`-Z Access Server Retry count`	Optional. Specifies the number of times the AccessGate/WebGate attempts to connect to the Access Server when the configureAccessGate tool is used.
`-k`	This option results in prompting for the password only for Simple or Cert mode transport security.

To reconfigure transport security mode through the command line

1. To reconfigure an AccessGate transport security mode, run the following command:

   ```
   configureAccessGate -i AccessGate_install_dir -t <AccessGate|WebGate> -R
   ```

   For example:

   ```
   configureAccessGate -i C:\COREid\WebComponent\access -t AccessGate -R
   ```

2. The system prompts you to for a transport security mode:

If you select Open. . .	If you select Simple. . .	If you select Cert. . .
The transport security mode is reconfigured to run in Open mode	1. Supply the AccessGate password, as follows: If you specified a password during installation or reconfiguration of the AccessGate, enter it. If you did not, press Enter to skip the prompt. 2. Supply the Global Access Protocol Pass Phrase. After you enter it, the system generates and installs the certificate.	1. Supply the AccessGate password, as follows: If you specified a password during installation or reconfiguration of the AccessGate, enter it. If you did not, press Enter to skip the prompt. 2. Supply the Global Access Protocol Pass Phrase. After you enter it, the system generates and installs the certificate. Next, complete the steps that follow.

Note: The Global Pass Phrase must always be the same for all AccessGates, WebGates, and Access Servers within an Access System installation.

For Cert mode: The system prompts you to specify whether you want to request a certificate or install a certificate.

- If you specify a certificate request, the system prompts you for the following organization information:

  Country name

  State or Province

  Locality

  Organization name

  Organizational unit

  Common Name:For example, HostName.DomainName.com

  Email address

- After you enter the information, a certificate request is generated and placed in the *Component_install_dir*\access\oblix\config\aaa_req.pem file.

  where *Component_install_dir* is the directory in which the Access System component is installed.

- You must have this certificate request signed by the Certificate Authority.
- The system prompts you for the full paths to the location of the Certificate Key file, the Certificate file, and the Certificate Chain file.

  After you specify the paths, the transport security mode is reconfigured.

To change the transport security mode password

1. From the command line, run the following command:

   ```
   configureAccessGate -i AccessGate_install_dir -t AccessGate -k
   ```

2. Enter the following information:
   - The old password
   - The new password
   - Reconfirm the new password

   The password is changed.

Deleting an AccessGate

If you delete an AccessGate, the applications and content on the hosts with which it was connected are not be protected by the Access System. Ensure this is what you want to do before deleting an AccessGate.

To delete an AccessGate

1. Uninstall the AccessGate from the host.

2. Launch the Access System Console and click **Access System Configuration**, and then click **AccessGate Configuration**.

   The Search for AccessGates page appears.

3. Select the search attribute and condition from the lists, or select All to find all AccessGates.

   The Look For list is a selection list of attributes that can be searched, as described in Table 3–2. The remaining fields allow you to specify search criteria that are appropriate for the selected attribute.

4. Click Go.

   The search results are displayed on the page.

5. Check the AccessGate that you want to delete and click Delete.

   You are prompted to confirm your decision.

6. Click OK to delete the AccessGate (or click Cancel to stop the deletion).

Managing WebGates

A WebGate is an out-of-the-box Access Client for HTTP-based resources. A WebGate is an NSAPI or ISAPI plug-in that intercepts HTTP requests for Web resources and forwards them to the Access Server.

The process of configuring a WebGate is similar to configuring an AccessGate. See "Adding an AccessGate" on page 3-24. The following topics provide additional information:

- Synchronizing Clocks with the Access Server
- Modifying a WebGate
- Configuring IP Address Validation for WebGates
- Viewing WebGate Diagnostics
- Checking the Status of a WebGate
- Placing a WebGate Behind a Reverse Proxy

Synchronizing Clocks with the Access Server

All Access Servers and their corresponding WebGates must be time-synchronized. Each secure request includes a timestamp.

For successful operation:

- Ensure all computers are synchronized within 60 seconds.
- Ensure each computer running a WebGate is not running ahead of the Access Servers with which it is associated.

Modifying a WebGate

Occasionally you may need to modify a WebGate's parameters. You can modify a WebGate through the Access System Console or through a command line tool named *configureWebGate*. Typically, you use the command line tool to change the transport security mode. This tool can be used in both Windows and Solaris installations.

To modify a WebGate through the command line

1. To modify a WebGate, navigate to the directory:

   ```
   WebGate_install_dir\access\oblix\tools\configureWebGate
   ```

   where *WebGate_install_dir* is the directory in which WebGate is installed.

2. From the configureWebGate directory, run the following command:

   ```
   configureWebGate -i WebGate_install_dir -t WebGate
   ```

Specify parameters using the commands listed in Table 3–4.

Example of using configureWebGate

The following is an example of configuring a WebGate using the configureWebGate tool on Microsoft Windows:

```
C:\COREid\webcomponent\access\oblix\tools\configureWebGate> configureWebGate -i
c:\COREid\webcomponent\access -t WebGate -w andium_AG -m cert -c install -S -P
milpid -h andium -p 5160 -a andium_AS -r 99malibu -Z 5
```

Configuring IP Address Validation for WebGates

IP address validation is specific to WebGates. It determines if a client's IP address is the same as the IP address stored in the ObSSOCookie generated for single sign-on. The IPValidation parameter turns IP address validation on and off. If IPValidation is true, the IP address stored in the ObSSOCookie must match the client's IP address, otherwise, the cookie is rejected and the user must reauthenticate. The default IPValidation setting is true.

The IPValidation parameter can cause problems with certain Web applications. For example, Web applications managed by a proxy server typically change the user's IP address, substituting the IP address of the proxy. This prevents single sign-on using the ObSSOCookie.

The IP Validation Exceptions parameter lists IP addresses that are exceptions to this process. If IPValidation is true, the IP address can be compared to the IP Validation Exceptions list. If the address is found on the exceptions list, it does not need to match the IP address stored in the cookie. You can add as many IP addresses as needed. These addresses are the actual IP addresses of the client, not the IP addresses that are stored in the obSSOCookie. If a cookie arrives from one of the exception IP addresses, the Access System ignores the address stored in the ObSSOCookie cookie for validation. For example, the IP addresses in the IP Validation Exceptions parameter can be used when the IP address in the cookie is for a reverse proxy.

To configure single sign-on between WebGate and an access client that does not have the client IP address at authentication, the IP validation can be explicitly turned off (set IP Validation to false). When the IP Validation parameter is set to false, the browser or client IP address is not used as a part of the ObSSOCookie. However, Oracle recommends that you keep IP validation on whenever possible.

To configure the IPValidation parameter setting

1. From the Access System Console, click the tab for Access System Configuration.
2. Click AccessGate Configuration in the left navigation pane.
3. Find an AccessGate and click its link.
4. In the details page for this AccessGate, click Modify.

5. To turn off validation, in the IPValidation field, select the Off option.
6. To turn on validation, in the IPValidation field select the On option and in the IPValidationExceptions field enter any IP addresses to exclude from validation.

   Use standard notation, for example, 10.20.30.123 for the addresses. Press the plus or minus buttons to add or delete IP addresses.

Viewing WebGate Diagnostics

A WebGate Diagnostic URL is available to display information regarding an Access Server connected to a WebGate. It also displays associated directory server information.

Diagnostic URL links are as shown in Table 3–5:

Table 3–5 Diagnostic URL links

Platform	URL
Domino	`http://WebGate_computer:portnumber/access/oblix/apps/webgate/bin/webgate.cgi?progid=1`
IIS	`http://WebGate_computer:portnumber/access/oblix/apps/webgate/bin/webgate.dll?progid=1`
Netscape and Apache	`http://WebGate_computer:portnumber/access/oblix/apps/webgate/bin/webgate.cgi? progid=1`

Where *WebGate_computer* is the computer in which the Web server and WebGate are installed and *portnumber* is the port number of the Web server.

Note: For IIS6, to use the Diagnostic URL feature, you must enable the direct access of webgate.dll through the IIS Lockdown tool.

When you access this URL, your browser displays the information shown in Table 3–6:

Table 3–6 Information Displayed in the Browser After Entering the Diagnostic URL

Topic	Description
Access Server	Hostname of the Access Server, its port number, and the number of connections with this WebGate
State	Status of the Access Server, either Up or Down
Created	Date and time this Access Server was installed
Install_Dir	Installation directory of this Access Server
Num Of Threads	Maximum number of threads allowed on the Access Server

Table 3–6 (Cont.) Information Displayed in the Browser After Entering the Diagnostic

Topic		Description
Directory Information	**Directory**	Type of information stored in this directory instance, User, Policy, or Oblix
	Host:Port	Hostname and port number of this directory instance
	State	Operational status of the directory server, Up or Down
	Priority	Priority of this Access Server to this WebGate, primary or secondary
	Mode	Directory server connection mode, Open or SSL
	Size Limit	Maximum entries the LDAP server returns for a search
	Time Limit	How long an LDAP operation in the LDAP server runs
	Login DN	Root DN of the directory server instance
	Created	Date and time this directory server instance was created

Checking the Status of a WebGate

Depending on the type of Web server you use, you can issue a URL to check the status of a WebGate from any browser.

To check the status of a WebGate

Issue one of the following URLS in the browser:

On IIS:

`http://`*`servername`*`:`*`port`*`/access/oblix/apps/webgate/bin/webgate.dll?progid=1`

On iPlanet and Apache:

`http://`*`servername`*`:port/access/oblix/apps/webgate/bin/webgate.cgi?progid=1`

On Domino:

`http://`*`servername`*`:port/access/oblix/apps/webgate/bin/nwebgate.dll?progid=1`

Checking the Number of Connections

If you modify the configuration of a WebGate or an AccessGate, the change takes effect in less than a minute. For example, if you add a new primary server and increase the number of connections to the Access Server by one, this happens without a restart of the server.

The old connections between the Access Server and the WebGate (or AccessGate) are discarded after a few minutes, when any pending requests are finished. If you issue a netstat command before the old connections are discarded, you might find double the number of connections since the server was started. However, this number quickly drops to the number of configured connections, usually in a few minutes. Every time that connection information is modified, the number of connections detected on a netstat command doubles for a few minutes, then drops back to the configured number.

Placing a WebGate Behind a Reverse Proxy

You can use WebGates with reverse proxies.

The following are benefits of a reverse proxy:

- All Web content can be protected from a single logical component as long as all requests go through the proxy.

  This is true even for platforms that are not supported by Oracle Access Manager. If you have different types of Web servers (for example, iPlanet, Apache, and so on) on different platforms (for example, Windows XP, Linux, and so on), all content on these servers can be protected. A reverse proxy can be a workaround for unsupported Web servers, eliminating the need to write custom AccessGates for unsupported Web servers and on platforms that do not have WebGate support, for example, MacOS.

- A reverse proxy offers architecture flexibility.

  Reverse proxies can allow deployments to expose an application that is available on the intranet to the extranet. Or applications that are available on the extranet can be exposed to the intranet. This can be done without any changes to the application that is already deployed.

- You only need to install a separate WebGate on the reverse proxy, rather than on every Web server.

  This allows for a single management point and can help with manageability of the system. You can manage the security of all of the Web servers through the reverse proxy without establishing a footprint on the other Web Servers.

The main pitfall of using a proxy is the extra work involved in setup. If you deploy the WebGate on a Web server that is behind a reverse proxy, the following are configuration requirements:

- Ensure that any Web server that uses the reverse proxy for authentication only accepts requests from the reverse proxies.

  This will also require that WebGates deployed on this Web server be configured to not enforce IP validation for requests from the reverse proxy server that front-ends the WebGate. This is done by configuring the known IP addresses of the reverse proxy server or servers in the IP Validation list. Note that while you can achieve the same effect by turning IP validation off for the WebGate, this is not a recommended approach due to security risks.

  Ensuring that the Web server only accepts requests from reverse proxies is typically done by adding an ACL statement in the server. This prevents users from bypassing the reverse proxy and directly accessing restricted content.

- Update the virtual hosts that are configured in the Policy Manager so that the Access System intercepts requests that are sent to the reverse proxy.

- Prevent people from circumventing the proxy by entering URLs that point directly to the back-end system.

  You can prevent this problem through the use of Web Server Access Control Lists or firewall filters.

- Since all user requests are processed by the proxy, you must deploy enough proxy servers to enable the system to handle the load.

- Redirect all existing URLs to the host name and port number of the reverse proxy server.

This often requires configuring the reverse proxy to perform content inspection and rewriting to prevent any absolute HTML links, for instance, to prevent broken link. This is achievable with most reverse proxies, and this is something you can configure independently of the Access System,.

- It is a best practice that URL links exposed to the front-ended applications rely on only relative URLs (../../*sub-path*/*resource*) rather than absolute URLs (http://*hostname.domain*:[*port*]/*path*/*resource*).

  Absolute URLs can break links on the end user's browser when deployed behind a reverse proxy.

Associating AccessGates and WebGates with Access Servers

You can associate an AccessGate with either individual Access Servers or with Access Server Clusters. For each AccessGate, you must select at least one Access Server or Access Server cluster with which it can communicate. The Access Server or the Access Server cluster must already be configured and installed.

Note: The process of associating a WebGate is the same as the process of associating an AccessGate.

You can view associated AccessGates in the Access Server details page or the Access Server Cluster details page. You can also view associated Access Servers and Access Server clusters in the AccessGate's details page. If there are any errors in the configuration between an AccessGate and an Access Server or an Access Server cluster, the error is displayed on the page.

For example, the security mode for an AccessGate could be different from the security mode of an associated Access Server or Access Server cluster. In such cases, the error is displayed on the page.

The rest of this section discusses the following topics:

- About Associating AccessGates with Clusters
- Associating an AccessGate
- Viewing AccessGates Associated with an Access Server
- Disassociating an AccessGate

About Associating AccessGates with Clusters

When you associate an AccessGate with an Access Server cluster, the AccessGate automatically communicates with all the Access Servers in the cluster. When you disassociate an AccessGate from a cluster, the connections between the AccessGate and the Access Servers in the cluster are automatically deleted.

When you add an Access Server to a cluster, all the AccessGates associated with the cluster automatically communicate with the new Access Server. When you delete an Access Server from the cluster, the connection between the AccessGate and the Access Server is automatically deleted.

Load balancing is automatically configured, based on the number of connections that you specified when you configured the AccessGate and the number of Access Servers in the cluster. For details about configuring load balancing, see the *Oracle Access Manager Deployment Guide*.

Failover is automatically configured when you associate an AccessGate with an Access Server cluster that you define as a primary or a backup cluster. For details about configuring failover, see the *Oracle Access Manager Deployment Guide*.

Associating an AccessGate

Use the following procedures to associate an AccessGate with an Access Server or cluster.

Task overview: Associating an AccessGate with an Access Server or cluster includes

1. Associating the individual components, as described in "To associate an AccessGate with an Access Server" on page 3-42
2. Configuring the communication between the components, as described in "To configure an Access Server to communicate with this AccessGate" on page 3-42
3. Associating the AccessGate with a cluster, as described in "To associate an AccessGate with an Access Server cluster" on page 3-43

To associate an AccessGate with an Access Server

1. Launch the Access System Console and click **Access System Configuration**, then click **AccessGate Configuration**.

   The Search for AccessGates page appears.

2. Select the search attribute and condition from the lists, or select All to find all AccessGates.

   The Look For list is a selection list of attributes that can be searched, as described in Table 3–2. AccessGate Search Attributes, Conditions, and Values. The remaining fields allow you to specify search criteria that are appropriate for the selected attribute.

3. Click Go.

   The search results are displayed on the page.

   The AccessGate Details page appears.

   If the AccessGate is not associated with any Access Server, do the following:

   a. Click Associate Access Servers

      The Associate Access Servers with AccessGate page appears.

   b. Select Individual Servers to associate the AccessGate with an Access Server.

   c. Click Next

   The page lists all primary and secondary Access Servers configured to communicate with the AccessGate.

To configure an Access Server to communicate with this AccessGate

1. From the Access System Console, click the Access System Configuration tab, then click AccessGate Configuration in the left navigation pane.

   Note: Remember that you must configure and install an Access Server before it can receive requests from an AccessGate.

2. Enter search criteria for an AccessGate, or click All in the search bar and click Go.
3. Click the link for an AccessGate to modify.
4. Click List Access Servers.
5. Click Add.
6. From the list, select an Access Server.
7. In the Select priority field, choose Primary or Secondary to specify whether the Access Server is a primary server or a secondary server.
8. Enter the maximum number of connections this AccessGate can establish to this Access Server.

   The default is 1.
9. Click Add to complete the configuration of this server, or click Back to return to the previous page.

To associate an AccessGate with an Access Server cluster

1. Launch the Access System Console and click **Access System Configuration**, and then click **AccessGate Configuration**.

   The Search for AccessGates page appears.
2. Select the search attribute and condition from the lists, or select All to find all AccessGates.

   The Search list is a selection list of attributes that can be searched, as described in Table 3–2 on page 3-17. The remaining fields allow you to specify search criteria that are appropriate for the selected attribute.
3. Click Go.

   The search results are displayed on the page.
4. Click the link for the AccessGate to associate with an Access Server cluster.

   The AccessGate Details page appears.

   - If the AccessGate is not associated with any clusters, do the following:
     - Click List Access Servers to see a list of configured Access Servers.

       The page lists all primary and secondary (backup) Access Server clusters configured to communicate with the AccessGate.
     - Click List Clusters to associate the AccessGate with clusters.

       The page lists all primary and backup clusters configured to communicate with the AccessGate.
5. Click Add to associate a Access Server cluster with the AccessGate.

   The Add a new Access Server Cluster to the AccessGate page appears.

   Note: You must configure an Access Server Cluster before it can receive requests from an AccessGate.
6. Select an Access Server cluster from the list.
7. In the Select Cluster Type field, choose Primary or Backup to specify whether the Access Server cluster is a primary cluster or a backup cluster.

The AccessGate opens connections to the Access Servers in the primary cluster. If the AccessGate cannot open the specified number of connections, it opens connections with the Access Servers in the backup cluster.

For details about configuring failover and load balancing, see the *Oracle Access Manager Deployment Guide*.

8. Click Save to save your changes (or click Cancel if you do not want to save your changes).

Viewing AccessGates Associated with an Access Server

You can view AccessGates that are associated with a particular Access Server.

To view AccessGates associated with a cluster

1. Launch the Access System Console and click **Access System Configuration**, and then click **AccessGate Configuration**.

   The Search for AccessGate page appears.

2. Select the search attribute and condition from the lists, or select All to find all AccessGates.

   The Search list is a selection list of attributes that can be searched, as described in Table 3–2 on page 3-17. The remaining fields allow you to specify search criteria that are appropriate for the selected attribute.

3. Click Go.

   The search results are displayed on the page.

4. Click the link for an AccessGate.

5. Click List Clusters.

6. Click an Access Server cluster to view its details.

   The Details for Access Server Cluster page appears.

7. In the Details for Access Server Cluster page, click View Associated AccessGates.

   The Associated AccessGates page appears.

8. Select a Primary Cluster to view AccessGates for which the cluster is configured as a primary cluster.

   Select a Backup Cluster to view AccessGates for which the cluster is configured as a secondary cluster.

   Select All to list all the specified AccessGates or enter a number to specify the number of search results you want displayed on the page.

9. Click Go to display the search results.

   The details of the AccessGates associated with the Cluster are displayed on the page.

   If there are multiple pages, click Next to go to the next page or click Previous to go back to the previous page.

10. Click Back to return to the previous page.

Disassociating an AccessGate

Use the following procedures to disassociate an AccessGate from an Access Server or cluster.

To disassociate an AccessGate from an Access Server or an Access Server cluster

1. Launch the Access System Console and click **Access System Configuration**, and then click **AccessGate Configuration**.

   The Search for AccessGates page appears.

2. Select the search attribute and condition from the lists, or select All to find all AccessGates.

   The Look For list is a selection list of attributes that can be searched, as described in Table 3–2 on page 3-17. The remaining fields allow you to specify search criteria that are appropriate for the selected attribute.

3. Click Go.

   The search results are displayed on the page.

4. Click the AccessGate that you want to disassociate from an Access Server or an Access Server cluster.

   The AccessGate Details page appears.

5. Click List Access Servers or List Clusters.

   The Access Servers or Access Server clusters associated with the AccessGate are listed on the page.

6. Choose whether to disassociate a server cluster or a server.

   - To disassociate an Access Server cluster, select the cluster and click the Delete button.
   - To disassociate an Access Server, select the Access Server and click the Delete button.

   The connection is deleted.

Configuring Preferred HTTP Hosts, Host Identifiers, and Virtual Web Hosts

You can identify Web server hosts to Oracle Access Manager in various ways, for example, by providing a computer name or an IP address. The following are examples of how the same host can be addressed:

- site.com
- site.com:80
- www.site.com
- www.site.com:80
- 216.200.159.58
- 111.111.11.1:80
- 3232236564 (decimal addressing)

There are two fields on the WebGate configuration page for identifying Web servers that host resources that you want to protect:

- Preferred HTTP Host: The Preferred Host field is required.

  The WebGate intercepts all user requests to this host, even if the user's request is a variation of the Preferred HTTP Host. For example, if the value of this field is myhost22:8080 the WebGate will protect all requests to this URL. But it will also intercept other ways of addressing the host, for example, http://123.123.123.123:8080, where 123.123.123.123 is the IP address of the server.

- Host Identifiers: Host identifiers are a list of all URL addressing methods.

  You create policies based on the host identifier. A WebGate protects all requests that match the addressing methods that you configured in the Host Identifier, and that match the Host Identifier used in the policy.

You can configure a third feature, DenyOnNotProtected, to deny users access to all resources unless access is explicitly allowed by a rule or policy. See "Denying Access to All Resources by Default" on page 3-50 for details.

Task overview: Protecting resources on a host

1. Configure a preferred HTTP host that corresponds to the computer that hosts the servers or virtual servers to be protected.

   See "About Preferred HTTP Hosts" on page 3-46 and "To configure a preferred HTTP host for a virtual Web server" on page 3-50 for details.

2. Configure Host Identifiers for each Web site or virtual Web site that you want to protect.

   See "Configuring Host Identifiers" on page 3-47 for details.

3. Create policies to protect these resources, using the host identifiers to differentiate among virtual hosts that are protected by the same WebGate.

   For example, you can configure policies that allow one user to access only of the one virtual sites that are protected by the WebGate. See "Protecting Resources with Policy Domains" on page 4-1 for details.

The rest of this section discusses the following topics:

About Preferred HTTP Hosts

You are always required to supply a Preferred HTTP Host when configuring a WebGate. See "Adding an AccessGate" on page 3-24 for details.

The rest of this section discusses the following topics:

About Preferred HTTP Hosts Without Virtual Web Hosting

If your environment does not use Web hosting, the value for Preferred HTTP Host is the name of the computer where the WebGate is installed, for example, myhost22, the port were the Web site is hosted, for example, 8080. The complete value would be

specified as myhost22:8080. The Preferred HTTP Host field must contain a host name that matches all possible methods by which the host can be addressed. The name you enter in this field must also match one of the names entered in the Host Identifiers field. See "Configuring Host Identifiers" on page 3-47 for details.

When you specify a value for the Preferred HTTP Host for a WebGate, and you specify the same value in a Host Identifier field, all policies that use the host identifier apply to all requests that the WebGate processes, regardless of how the request was addressed originally. For example, if you have a WebGate on a Web server on host1.company.com, you can set the Preferred HTTP Host value to `host1.company.com`. Any policy that you configure for host1 will be applied to a request sent to the host, no matter how the user addresses the resource. For example, suppose that the user sets up an /etc/hosts file to map to the arbitrary host name my.host.com and requests the following URL:

http://my.host.com/someResource

The WebGate uses the Preferred HTTP Host when setting policies for this request, despite the URL that the user supplied. Requests for resources that match values in the Preferred HTTP Host field are redirected to the Access Server for policy evaluation. The Preferred HTTP Host feature prevents security holes that can be inadvertently created if a host's identifier is not included in the Host Identifiers list.

About the Preferred HTTP Host Setting for a Virtual Host

Configuring a Preferred HTTP Host is also required for environments that use virtual hosting. For virtual hosts, you specify a reserved value in the Preferred HTTP Host field. See "Configuring Virtual Web Hosting" on page 3-49 for details. This field forces WebGate to pass the preferred host string to the Access Server for policy evaluation instead of the host typed into the browser by the user. No matter what is typed into the browser, the Access Server always sees the preferred host.

Configuring Host Identifiers

As described in the previous section, a host can be known by multiple names. Use the Host Identifiers feature to enter the official name for the host, and every other name by which the same host can be addressed by users. A request sent to any address on the list is mapped to the official host name.

As described in "Protecting Resources with Policy Domains" on page 4-1, when you define policies to protect resources on the hosts, you use the name in the host identifiers field. When the WebGate intercepts a request, it checks the request for an address. If the address is on the host identifiers list, this address is mapped to the official host name, and the Access System can apply the rules and policies that protect the resource.

In your Host Identifiers list:

- Each host name must be unique.
- Each *host name*:*port* pair must be unique.
- Each *host name*:*port* pair must belong to only one host identifier.
- Each *host name*:*port* pair must match the end user's entry exactly.

With decimal addressing it may be impractical to define all possible URL combinations for the same site. Using the following formula to calculate possible decimal addresses for the original address 01.02.03.04, where each 0 is an 8 bit octet, you will find many ways to represent the original IP address:

Formula:

01*256^3+02*256^2+03*256+04

The following URL values are all for the same site:

http://%61%73%74%65%72%69%78

http://%31%39%32%2E%31%36%38%2E%34%2E%32%30/

http://%33%32%33%32%32%33%36%35%36%34

For more information, you may want to look at the site http://www.karenware.com/powertools/ptlookup.asp.

See "To add a Host Identifier" on page 3-49 for the steps to create a host identifiers list.

You must specify a Preferred Host, and you can specify one or more Host Identifiers. You can provide host identifiers to identify a single host, and to identify multiple virtual Web hosts.

The rest of this section discusses the following topics:

- Including Authenticating Hosts
- Viewing or Deleting Host Identifiers
- Adding a Host Identifier

Including Authenticating Hosts

If you redirect an authentication challenge to be processed by another host, you must add the name of that host to the Host Identifiers list. The host name that you enter in the Challenge Redirect field must be available in the Host Identifiers list when adding or modifying an authentication scheme. For example, if a user is redirected to an SSL-enabled server for authentication, that server must be included.

When adding URL prefixes to a policy domain, the Delegated Access Administrator must specify a server hosting the URL prefix. When a user attempts to access a URL that is protected by the policy domain, the user is redirected to the server specified in the Challenge Redirect field for authentication.

Viewing or Deleting Host Identifiers

The Host Identifier details page displays the name, description, and host name variations, as described in the following procedure.

To view or delete Host Identifiers

1. Launch the Access System Console and click Access System Configuration.

   The Access System Configuration page appears.

2. In the left navigation pane, click Host Identifiers.

   The List all Host Identifiers page appears. The existing host identifiers are listed on the page.

3. To view a host identifier, select its name from the list.

4. To delete a host identifier, select its name from the list and click Delete.

Adding a Host Identifier

When you configure host identifiers in an environment that does not use virtual Web hosting, you must enter one of the host name variations in the Preferred HTTP Host

field to ensure that single sign-on works properly. If you attempt to add a host name variation that already exists for a different host identifier, a message alerting you of the duplication is displayed. You can save or cancel your changes.

If you support virtual Web hosting as described in "About Preferred HTTP Hosts" on page 3-46, you supply a reserved name in the Preferred HTTP Host field instead of a host name variation.

To add a Host Identifier

1. Launch the Access System Console and click the Access System Configuration tab.

   The Access System Configuration page appears.

2. In the left navigation pane, click Host Identifiers.

   The List all Host Identifiers page appears.

3. Click Add to add a new host identifier.
4. In the Name field, type the name of the host.
5. In the optional Description field, type a short description.
6. In the Hostname variations field, enter all variations for identifying this host.

   The Access System does not add a default port number if you do not provide one.

7. Click Save.

Configuring Virtual Web Hosting

You can install a WebGate on a Web server that contains multiple Web site and domain names. The WebGate must reside in a location that enables it to protect all of the Web sites on that server.

The virtual Web hosting feature of many Web servers enables you to support multiple domain names and IP addresses that each resolve to their unique subdirectories on a single virtual server. For example, you can host abc.com and def.com on the same virtual server, each with its own domain name and unique site content. You can have name-based or IP-based virtual hosting.

Prior to Oracle Access Manager 10.1.4.0.1, you were not required to specify a Preferred HTTP host. This was useful for virtual Web hosting because you could specify several host identifiers and have them resolve to different virtual Web hosts. As of Oracle Access Manager 10.1.4.2, to support virtual Web hosting, you must specify a specific value, described in the following paragraphs, in the Preferred HTTP host field.

The following summarizes configuration for virtual servers:

- To support virtual hosts on most Web servers other than Apache-based servers, you must set the Preferred HTTP Host value to HOST_HTTP_HEADER.

  If you specify this value, when user's browser sends a request, the WebGate sets the value of the Preferred HTTP Host to the host value in the request. For example, suppose a user enters the string myweb2 in a URL:

  http://myweb2

  On the Web server, if one of the Web sites has a host named myweb2, the request is served by the matching virtual site.

- On Apache-based servers, for example, Apache, Apache 2, IBM HTTP Server, Oracle HTTP Server, and so on, the Preferred HTTP Host value must be set to SERVER_NAME.

Note: The SERVER_NAME value is not supported for any host other than an Apache-based server. If you set this value for a non-Apache-based server, users will be unable to access any resources that are protected by WebGate on that Web server. Users will, instead, receive an error that the WebGate configuration is incorrect.

To configure a preferred HTTP host for a virtual Web server

1. Launch the Access System Console, click Access System Configuration, then click AccessGate Configuration.

   The Search for AccessGates page appears.

2. Select the search attribute and condition from the lists, or select All to find all AccessGates.

   See Table 3–2 on page 3-17 for a description of searchable attributes. You can specify search criteria that are appropriate for the selected attribute.

3. Click Go.

   The search results are displayed on the page.

4. Click the name of the AccessGate you want to modify.

   The AccessGate Details page appears.

5. Click Modify.

   The Modify AccessGate page appears.

6. In the Preferred HTTP Host field, enter the following:

   - On most Web servers, enter HOST_HTTP_HEADER.
   - On an Apache-based server, enter SERVER_NAME.

7. For IIS, to support virtual hosting, you also need to go to the IIS console and configure each virtual Web site to contain three fields:

   – Host Header Name

   – IP address

   – Port

   See the following for details:

   - http://www.simpledns.com/kb.aspx?kbid=1149
   - http://support.microsoft.com/kb/q190008/

Denying Access to All Resources by Default

Access System default behavior is to allow access when a resource is not protected by a rule or policy. This is accomplished using one Boolean flag, DenyOnNotProtected, located in the AccessGate configuration page. The default setting is false, which means that access is *allowed* to resources not protected by a rule or policy.

When set to true, the DenyOnNotProtected parameter lets you establish the opposite behavior. When set to true, DenyOnNotProtected denies access to all resources to which access is not explicitly allowed by a rule or policy. This can limit the number of times the WebGate queries the Access Server, and can improve performance for large or busy policy domains.

Because different Web servers and Access Clients have different requirements, DenyOnNotProtected is implemented through WebGate. DenyOnNotProtected *cannot* be used with other types of AccessGates.

> **Important:** DenyOnNotProtected overrides Host Identifiers and Preferred Host. Oracle recommends setting DenyOnNotProtected to true. Setting DenyOnNotProtected to false can cause security holes in large installations with multiple Host Identifiers, virtual hosts, and other complex configurations.

To deny access to all unprotected resources

1. From the Access System Console, click the tab for Access System Configuration.
2. Click AccessGate Configuration in the left navigation pane.
3. Find an AccessGate and click its link.
4. In the details page for this AccessGate, click Modify.
5. Change the DenyOnNotProtected setting to true to *deny* access to all unprotected resources.
6. Restart the WebGate to enable the change to take effect immediately.
7. If you have set DenyOnNotProtected to true, you must also protect Login.html with the Anonymous authentication scheme; otherwise, the page will not display when you access its associated resource.

Example of Using DenyOnNotProtected

Suppose you have a computer with IP addresses A and B associated with it, both on port 80, and using the same configuration file. For Netscape and iPlanet, this would be the obj.conf files. Both of these virtual servers are protected by the same AccessGate or WebGate.

The goal is to protect all content on both virtual servers without using a Preferred Host. To meet this goal, you may set up a host ID for all variations of A, and then protect some content on A by defining policies for specific URLs. You need not set a Preferred Host for either AccessGate A or B. You may also set the value of DenyOnNotProtected to true for the WebGate protecting the AccessGate, so by default all content is protected on A and B.

With this setup, when a user tries to access a URL on A, the policies are evaluated first and if no corresponding Access Policy is found, content is denied only for A.

Associating a WebGate with Particular Virtual Hosts, Directories, or Files

When using an Apache reverse proxy for single sign-on, in the Apache httpd.config file you specify the Web sites to run on the Apache server. The settings can be global across all Web sites or local to a Web site. You can restrict the Oracle Access Manager loading references in the httpd.config file to be associated with a specified site, with virtual hosts, specific directories or even files.

To associate the WebGate with specific targets, you move the following directives the the http.conf file:

```
AuthType Oblix
require valid-user
```

You can put these directives in a block that tells Apache to use WebGate for every request. You can also move the directives to a block that limits when the WebGate is called. The following is an example of putting the LocationMatch directive after a VirtualHost directive:

```
  DocumentRoot /usr/local/apache/htdocs/myserver
  ServerName myserver.example.net
  AuthType Oblix
  require valid-user
```

After you move the LocationMatch block to the VirtualHost directive, the WebGate will only work for that virtual host. You can add the LocationMatch block to as many virtual hosts as you want. The following examples shows how you could protect one virtual server:

```
ServerAdmin webmaster@example.net
DocumentRoot "Z:/Apps/Apache/htdocs/MYsrv"
ServerName apps.example.com
    ProxyRequests On
    SSLEngine on
    SSLCACertificateFile Z:/Apps/sslcert_myapps_ptcweb32/intermediateca.cer
    SSLCertificateFile Z:/Apps/sslcert_myapps_ptcweb32/sslcert_myapps_ptcweb32.cer
    SSLCertificateKeyFile
Z:/Apps/sslcert_myapps_ptcweb32/sslcert_myapps_ptcweb32.key
    ErrorLog logs/proxysite1_log
    CustomLog logs/proxysite1_log common
    ProxyPass /https://apps.example.com/
    ProxyPassReverse /https://apps.example.com/
    ProxyPass /bkcentral https://apps.example.com/bkcentral
    ProxyPassReverse /bkcentral https://apps.example.com/bkcentral
    ProxyPass /NR https://apps.example.com/NR
    ProxyPassReverse /NR https://apps.example.com/NR

      AuthType Oblix
      require valid-user

#*** BEGIN Oracle Access Manager WebGate Specific ****

LoadModule obWebgateModule
Z:/apps/Oracle/WebComponent/access/oblix/apps/webgate/bin/webgate.dll
WebGateInstalldir Z:/apps/Oracle/WebComponent/access
WebGateMode PEER

 SetHandler obwebgateerr

SSLMutex sem
SSLRandomSeed startup builtin
SSLSessionCache none

SSLLog logs/SSL.log
SSLLogLevel info
# You can later change "info" to "warn" if everything is OK
```

The Access Login Process

When a user or an application, such as a .JSP or Java application, attempts to access an Identity System application or an Access System-protected Web resource, the login process is set in motion. This process varies depending on the following factors:

- **Is the resource protected, and by what authentication scheme?**

If the resource is protected by a WebGate, the Access System challenges the user as specified by the challenge method defined in the authentication scheme.

If the resource is an unprotected Identity System application (for example, the User Manager), the Identity System uses its own login form to challenge the user for credentials.

- **Can the user be authenticated?**

  To ensure that the user is who he or she claims to be, WebGate challenges the user for credentials. If the user supplies appropriate credentials, the WebGate authenticates the user, generates the ObSSOCookie, and sets it in the user's browser.

  For information on the ObSSOCookie, see "Cookies Generated During Login" on page 3-58.

- **Have you configured single sign-on for the Access System?**

  The Access System's single sign-on capability enables users to access multiple protected URLs or applications with a single login. If single sign-on has been implemented in a domain, the user must authenticate only once to access multiple resources protected by an authentication scheme which has the same level or a lower level of security. The ObSSOCookie is passed from the user's browser to any WebGates configured for the domain.

  When Access System single sign-on is implemented in a multi-domain environment, an authentication is honored by all the hosts in two or more domains.

  See "Configuring Single Sign-On" on page 7-1 for more information on configuring SSO for single- and multi-domain environments.

- **Is the user allowed to access the content, and what actions can he or she perform?**

  WebGate queries the Access Server to determine if the user is authorized to access the resource. Access Server performs the authorization. If the user is authorized, the Access Server checks for a policy that specifies the actions that the user is allowed perform.

  The Access Server sends the information to WebGate, which then returns the requested resource to the user.

Figure 3–1 illustrates the Access System’s authentication process.

Figure 3–1 The Access System's Authentication Process

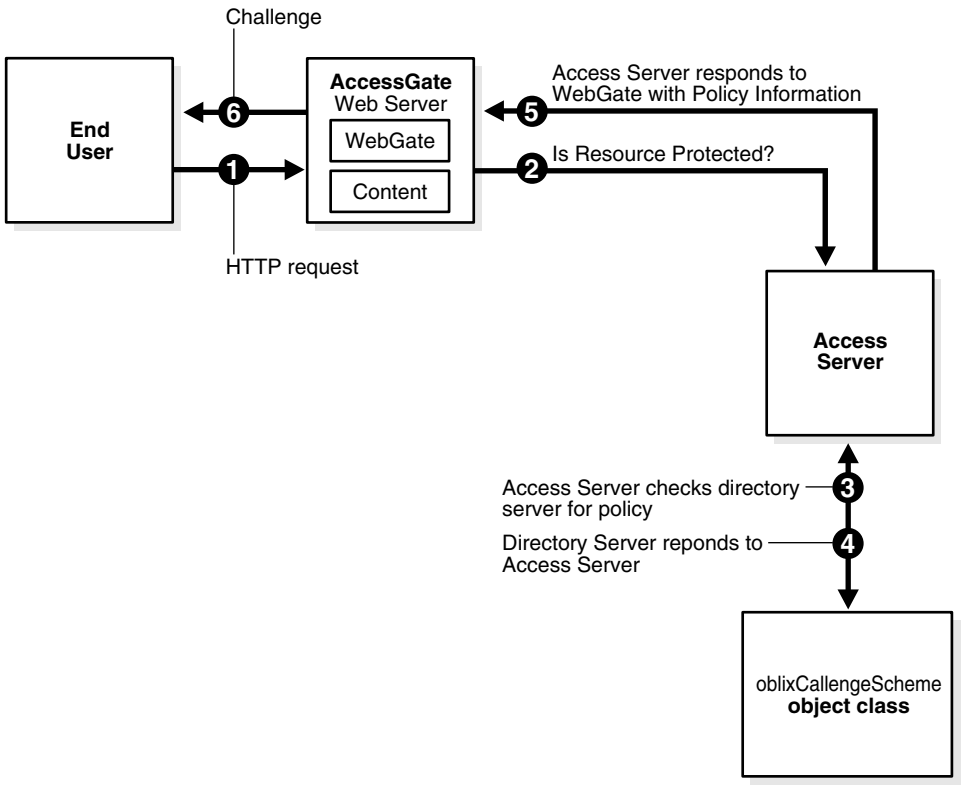
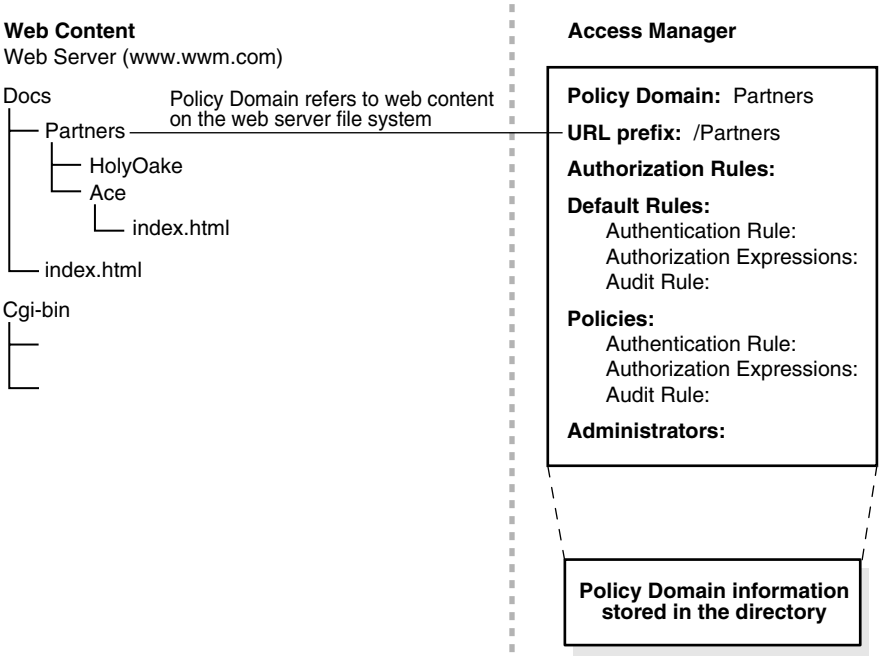


Figure 3–2 illustrates the Access System's authorization process.

Figure 3–2 The Access System's Authorization Process



Combined Authentication and Authorization Process

Figure 3–3 shows all of the processes involved in serving unprotected resources, verifying a user's identity for a protected resource, and authorizing the user and serving the protected resource.

Figure 3–3 Overall Process for Evaluating If a Resource is Protected, Authentication, Authorization, and Serving the Resource

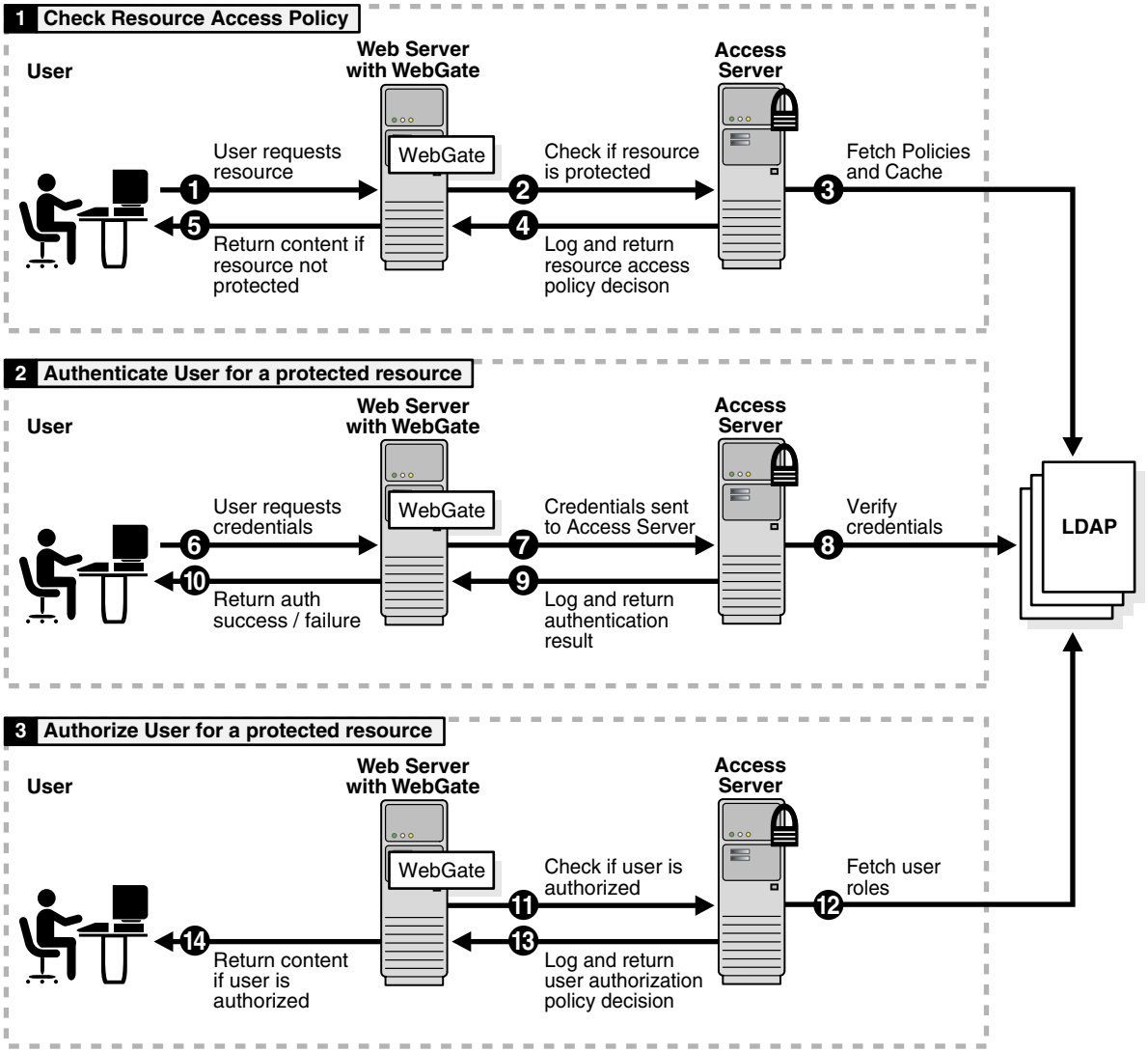


Figure 3–3 illustrates the following:

- An initial request is intercepted by a WebGate, which forwards the request to the Access Server.
- Policies are retrieved from the directory server to determine if the resource is protected.

  If the resource is not protected, it is served.
- If the resource is protected, the WebGate prompts for credentials, and forwards them to the Access Server.

The Access Server matches the credentials with records in the directory.

- If the authentication succeeds, a second round-trip is performed to check the authorization policies in the directory.

  If the authorization rules permit, the resource is served.

Login Processes

This section describes different scenarios where a user attempts to access an Access System-protected resource.

Process overview: Access when an Identity System application is not protected by WebGate

1. A user attempts to access an Identity System application that is not protected by a WebGate.
2. The Identity System challenges the user for credentials such as user name and password.
3. If the user authenticates successfully, the Identity application generates the ObTEMC and the ObTEMP cookies.

   See "Cookies Generated During Login" on page 3-58 for information on cookies.
4. The Identity System then allows the user to access the Identity application. The user can perform specific actions in accordance with how the application is configured.

Process overview: Access when the resource is protected by WebGate

1. A user attempts to access a resource that is protected by a WebGate.

   When the user attempts to access a Web resource or an application, WebGate intercepts the request and queries the Access Manager SDK caches to determine if the requested resource and the associated operations are protected.
2. If information on the requested resource is not in the cache, WebGate makes a request to the Access Server for the security policy to determine if the resource is protected by the Access System.
3. If the resource is an unprotected Identity System resource, WebGate forwards the request to the server storing the resource, and the Identity System application authenticates the user as in "Process overview: Access when an Identity System application is not protected by WebGate" on page 3-56.

   If the resource is protected, WebGate looks for the ObSSOCookie to determine whether the user has already been authenticated.
4. If the user is not authenticated, the server challenges the user for credentials. The challenge method varies depending on the authentication scheme used.

   If a form-based authentication scheme is used, WebGate generates the ObFormLoginCookie. See "Cookies Generated During Login" on page 3-58 for details. For general information about authentication schemes, see "Defining a New Authentication Scheme" on page 5-9.
5. If the user authenticates successfully, WebGate generates the ObSSOCookie and sets it for inclusion in the next response to the user's browser.

Depending on the actions specified for authentication success and authentication failure, the user may be redirected to a specific URL, or user information may be passed on to other applications through a header variable or a cookie value.

6. WebGate then queries the Access Server for information on authorization for the resource.
7. The Access Server queries and evaluates the appropriate authorization policies stored in the directory server, and passes on the information to WebGate.

   For SSO to a an Identity System application, the authorization policy must set an action to set the HTTP_OBLIX_UID header to the user identity for the Identity System application. If this header is not set, the application authenticates the user as in "Process overview: Access when an Identity System application is not protected by WebGate" on page 3-56.
8. If the user is authorized, access to the requested content is allowed, and the HTTP_OBLIX_UID header is set.

   Depending on the authorization actions specified for authorization success and authorization failure, the user can be redirected to a specific URL, or user information can be passed on other applications through a header variable or a cookie value.
9. The Identity System application reads the HTTP_OBLIX_UID header variable to get the identity. The Identity System application determines the user's access rights from the identity.

Process overview: Identity resource protected by WebGate

1. A client application, such as a .jsp or a Java application, attempts to access a URL to a Identity resource that is protected by WebGate.

   The client application uses Access Manager API to interface with the Access System's Authentication and Authorization services. The application supplies the client's user credentials to the Access Manager API.
2. The Access Server queries and evaluates the appropriate authentication rule.
3. The Access Manager API authenticates the user and creates a session token.
4. The Web application sets the ObSSOCookie with the session token for the domain containing the client application and sends the cookie along with the request to the client application.
5. The WebGate queries the Access Server for information on authorization for the client user.
6. The Access Server queries and evaluates the appropriate authorization policies stored in the directory server, and passes on the information to WebGate.

   For SSO to an Identity System application, the authorization policy must set an action to set the HTTP_OBLIX_UID header. The header must set the user identity to be used by the Identity System application. If this header is not set, the application authenticates the client application itself, as in "Process overview: Access when an Identity System application is not protected by WebGate" on page 3-56.
7. If the client application is authorized, access to the requested content is allowed and the HTTP_OBLIX_UID header is set.

   Depending on the authorization actions specified for authorization success and authorization failure, the user may be redirected to a specific URL, or user

information may be passed on other applications through a header variable or a cookie value.

Note: If the client application is authorized, it is allowed to access the resource.

Cookies Generated During Login

Depending on the scenario, the Access System generates one or more cookies. Cookies contain information such as the user DN, the client's IP address, and the cookie expiry time.

- ObSSOCookie
- ObFormLoginCookie
- ObTEMC cookie
- ObTEMP cookie
- ObPERM cookie

ObSSOCookie

The ObSSOCookie is an encrypted single sign-on cookie that is generated by the Access Server when a user authenticates successfully. The ObSSOCookie is a session-based cookie that stores user identity information. You can cache the information, if necessary.

See "Configuring Single Sign-On" on page 7-1 for more information on the ObSSOCookie.

Some browsers, for example, Netscape, Mozilla, or Firefox share the session state among all open browser windows. If you log in to Oracle Access Manager and the browser stores the obSSOCookie, the same cookie is shared with all open browser windows. If you open another browser window and access a protected resource, you are automatically authenticated as the same user that has the current ObSSOCookie stored in the browser.

The only way to destroy the ObSSOCookie in these browsers is to delete or clear the ObSSOCookie, configure a Logout URL, or close all browser sessions on your desktop. See "Configuring Logout" on page B-1 for details.

ObBasicAuthCookie

The ObBasicAuthCookie is an HTTP basic method is used to protect a resource. See "Default Authentication Schemes" on page 5-4 for details.

ObFormLoginCookie

The ObFormLoginCookie is generated when a form-based authentication scheme is used to protect a Web resource. WebGate uses the ObFormLoginCookie to direct the user back to the requested resource after successful authentication.

The ObFormLoginCookie maintains the original request information. By default, this cookie is set when the browser is first redirected to the form. The ObFormLoginCookie contains the following information for the original request:

- The requested URL
- The requested operation

- An authentication scheme
- The host to return to URL

See "Form-Based Authentication" on page A-1 for more information.

ObTEMC cookie

The ObTEMC cookie, an encrypted session-based cookie, is generated by the Identity application when a user authenticates successfully. The ObTEMC cookie contains the following information:

- User distinguished name (DN) and the original DN. Original DN information is stored only if the Identity Substitute Rights feature is used. For details about adding substitute administrators and substitution rights, see the *Oracle Access Manager Identity and Common Administration Guide*.
- A flag specifying whether the user is a Master Administrator, an Identity Administrator, or an Access Administrator.
- If single sign-on (SSO) has been implemented, the SSO Login ID.
- The time stamp.

  Every time a user performs an action, the time stamp is updated in the cookie to reflect the last time the session was used. If SSO has been implemented, however, the time stamp is ignored.
- The IP address of the client computer.

ObTEMP cookie

The ObTEMP cookie is a session-based cookie that is generated by the Identity application when a user authenticates successfully. The ObTEMP cookie contains the following application information:

- Login name
- User type
- Number of search-generated results (selector info)
- Uncommitted changes in various configuration applets

ObPERM cookie

The ObPERM cookie is a permanent cookie that is stored on the client computer. Between user sessions, the ObPERM cookie stores the following application information:

- Application style
- Custom view search results

Part II

Protecting Resources

Protecting resources includes configuring Oracle Access Manager policy domains as well as user authentication and authorization, and single sign-on.

Part II contains the following chapters:

4

Protecting Resources with Policy Domains

The Access System enables you to control who can access resources such as Web content and traditional applications by defining policy domains. Policy domains usually contain one or more URL prefixes that identify resources on the Web. Policy domains contain authentication and authorization rules that determine who can access the resources, and at what time. Additionally, a policy domain can protect non-Web based resources.

You can also create policies within a policy domain to define fine-grained protection for resources, for example, to protect a specific Web page or set of pages.

For each policy domain and policy, you can define audit rules to monitor and record system events, successful and failed user authentications, successful and failed authorization of users, and so on.

This chapter explains how to configure policy domains, policies, and auditing rules. This chapter discusses the following topics:

Prerequisites

Oracle Access Manager should be installed and running, as described in the *Oracle Access Manager Installation Guide.*

You should also read the following before creating a policy domain:

- Chapter 3, "Configuring WebGates and Access Servers" on page 3-1 describes how to configure AccessGates and Access Servers, something you must do before the policy domains you create can take effect.
- Chapter 5, "Configuring User Authentication" on page 5-1 describes how to create authentication rules and schemes that you include in your policy domains.
- Chapter 6, "Configuring User Authorization" on page 6-1 describes how to create authorization rules that you include in your policy domains.

In addition, a Master Administrator must complete the tasks described in the following task overview before any policy domains can be created.

Task overview: Prerequisite tasks for a Master Administrator

1. Define the policy base during Policy Manager setup, as described in "About the Policy Base" on page 4-2.

   To review the policy base from the Access System Console, click System Configuration, Server Settings. On the View Server Settings page, locate the Policy Data Configuration section to obtain computer name, port, root DN, directory server security, and policy base.

2. Define the policy domain root during Policy Manager setup, as described in "About the Policy Domain Root" on page 4-2.
3. Create a Master Access Administrator to create policy domains, resource types, and access control schemes, and to assign the role of Delegated Administrator of a policy domain to other people:
   - See "Configuring Master Access Administrators" on page 2-3.
   - See "Delegating Policy Domain Administration" on page 4-46.

About the Policy Base

During Policy Manager installation, the Master Administrator specifies where to store policy data. During Policy Manager setup, a policy base is also created. The policy base is the top node in the directory tree for object classes for policy domains and their policies.

All information about a policy domain is stored in relation to the policy base. All information about a policy domain, including its rules and the resources it protects, is stored at the same level under the policy base.

The Master Administrator must define a policy root before you can configure a policy domain or policies. See "About the Policy Domain Root".

Protecting WebPass and Policy Manager URLs: Oracle recommends that you protect WebPass and Policy Manager instances with WebGate. You can do this by installing these Web component instances against the same Web server and enabling default Identity domain and Access domain policies during Policy Manager setup. You can further protect WebPass and Policy manager URLs by creating policies that restrict access to URLs to only administrative users.

About the Policy Domain Root

During Policy Manager setup, the Master Administrator specifies a policy domain root. The policy domain root is a URL prefix under which all resources are protected.

The default policy domain root must be broad to encompass all of your resources. The default root is /.

> **See also:** For details about URLs, see "Configuring URLs for Resources" on page 4-14.
>
> For information on defining the policy domain root during Policy Manager setup, see the *Oracle Access Manager Installation Guide.*

About Policy Domain Administration

To protect resources, you create *policy domains.* A policy domain consists of:

- The resources you want to protect

  You can protect Web-based and non Web-based resources, including Web pages, site domains (collections of Web resources), servers, files, applications, and other executable programs.

  To include resources in a policy domain, you specify a URL that includes the resource at some level. You can specify the URL of a single resource, for example, an application, or you can protect entire directories of folders, files, and executables under a URL.
- Authentication, authorization, and auditing rules
- An authorization expression
- Policies to protect subsets of resources within the policy domain

  A policy consists of a set of resources, authentication, authorization, auditing rules, and an authorization expression.
- The rights given to various administrators to create and modify the policy domain

The rest of this section discusses the following topics:

> **Note:** Before you can protect a resource, a Master Administrator must define the policy domain root and policy base, as described in "Prerequisites" on page 4-1. The Master Administrator must also define a Master Access Administrator as described in "Configuring Master Access Administrators" on page 2-3.

About Creating the First Policy Domain

A Master Access Administrator must create the first policy domain after the policy domain root is defined. He or she can then create policy domains for URLs beneath the first one and delegate administration of those policy domains to other administrators.

For details about the policy domain root, see "About the Policy Domain Root" on page 4-2.

Task overview: Creating the first policy domain

1. Define the resource types for any resources to be included in the domain whose types are not already defined by default.

See "Configuring Resource Types" on page 4-11 for details.

2. Create the Master Audit Rule.

   See "About the Master Audit Rule" on page 4-34 for details.

3. Create the Authentication Scheme for the policy domain.

   See "About Authentication and Authentication Schemes" on page 5-1 and "Managing Authentication Rules" on page 5-48 for details.

4. Create the Authorization Scheme for the policy domain.

   - See "Configuring Authorization Rules" on page 6-7 for information about using an Access System-provided authorization scheme
   - See "About Authorization Schemes for Custom Plug-Ins" on page 6-45 for information about custom schemes.

5. Configure the URLs for the resources of the first policy domain.

   See "Configuring URLs for Resources" on page 4-14 for details.

6. Create the Authentication Rule for the policy domain.

   See "Managing Authentication Rules" on page 5-48 for details.

7. Create actions for the Authentication Rule (optional).

   See "Managing Authentication Actions" on page 5-53 for details.

8. Create one or more Authorization Rules.

   See "About Authorization Rules" on page 6-4 for details.

9. Create actions for the Authorization Rules.

   See "Setting Actions for Authorization Rules" on page 6-41 for details.

10. Create an authorization expression for the policy domain containing one or more authorization rules.

    See "About Authorization Expressions" on page 6-14 for details.

11. Create the audit rule for the policy domain.

    See "Creating an Audit Rule for a Policy Domain" on page 4-42 for details.

12. Test the policy domain.

    See "Using Access Tester" on page 4-44 for details.

13. Delegate management of the domain to a Delegated Access Administrator.

    See "Delegating Policy Domain Administration" on page 4-46 for details.

See Also: "Creating a Policy Domain" on page 4-26

About Managing a Policy Domain

As a Delegated Access Administrator, you can manage a policy domain for which you have been granted administrative rights.

The following task overview describes tasks you can perform as a Delegated Access Administrator. You perform these tasks as needed; none is required.

Task overview: Managing a policy domain

1. Replace the authentication rule for the policy domain or its policies.

For this task, you must first delete the policy domain's or the policy's existing authentication rule.

To create an authentication rule, you select an authentication scheme that was created by a Master Access Administrator, and configure the rule's actions. See "Managing Authentication Rules" on page 5-48. for details

2. Replace the authorization expression for the policy domain or its policies.

   For this task, you must first delete the content of the policy domain's or policy's existing authorization expression.

   To create an authorization expression for the policy domain or any of its policies, you combine one or more authorization rules created at the policy domain level. See "About Authorization Expressions" on page 6-14.

3. Create audit rules derived from the Master Audit Rule.

   See "Creating an Audit Rule for a Policy Domain" on page 4-42 and "Defining an Audit Rule for a Policy" on page 4-43.

4. Test the policy domain.

   See "Using Access Tester" on page 4-44.

Overview of Creating a Policy Domain

As a Delegated Access Administrator with a particular set of privileges, you can create policy domains. See Table 4–4 for details.

The following task overview summarizes the procedures for creating a policy domain.

Task overview: Creating a policy domain

1. Configure the URLs for the resources of the policy domain.

   See "Configuring URLs for Resources" on page 4-14 for details.

2. Create the authentication rule for the policy domain.

   See "Managing Authentication Rules" on page 5-48 for details.

   You include an authentication scheme created by a Master Access Administrator in the rule.

3. Create actions for the authentication rule.

   See "Managing Authentication Actions" on page 5-53 for details.

4. Create one or more authorization rules for the policy domain.

   See "About Authorization Rules" on page 6-4 for details.

5. Define actions to be taken for the authorization rule, if the rule fails or if it succeeds.

   See "About Authorization Actions" on page 6-37 for details.

6. Create the authorization expression for the policy domain containing one or more authorization rules.

   See "About Authorization Expressions" on page 6-14 for details.

7. Create actions to be taken for the authorization expression, depending on the evaluation of the expression: success, failure, or inconclusive.

   See "About Actions For Rules and Expressions" on page 6-37 for details.

8. Create the audit rule for the policy domain.

   See "Creating an Audit Rule for a Policy Domain" on page 4-42 for details.

9. Test the policy domain.

   See "Using Access Tester" on page 4-44 for details.

10. Configure policies for the policy domain, if any.

    See "Configuring Policies" on page 4-38 for details.

Note: You can configure a policy domain resource URL that is already covered by a resource definition in another policy domain for which you have administrative rights. For details, see "How the Policy Domain or Policy for a Resource Is Determined" on page 4-8.

About Policy Domains and Their Policies

Policy domains are logical structures defined for resources that you want to protect in the same way. Particular policies provide different and more specific coverage for a subset of the policy domain's resources.

When users request access to resources protected by a policy domain, their requests are evaluated according to the domain's authentication rule and its authorization expression.

There are several ways that users can attempt to access a resource that is protected by a policy domain, for example, by entering a URL in a browser, by running an application, or by calling some other external business logic.

The rest of this section discusses the following topics:

- Parts of a Policy Domain
- How the Policy Domain or Policy for a Resource Is Determined
- Preconfigured Policy Domains
- Who Creates Policy Domains?
- Examples of Policy Domains and Policies
- About Allocating Responsibility for a Policy Domain

Parts of a Policy Domain

A policy domain consists of the following parts:

- URLs that define paths to resources that are protected by the domain's authentication rule and authorization expression

  A policy domain can include multiple URLs that are independent of one another. Resources under one URL can reside on a different host from resources under another URL in the same policy domain. The policy domain's default rules apply to the resources it contains, unless the resource is protected by a specific policy.

- The host identifier

  You identify all resources that you add to a policy domain by the host where the resources reside and their URLs. A host can be known by multiple names. To ensure that it recognizes the URL for a resource, the Access System must know the various ways used to refer to that resource's host computer.

The Host Identifiers feature enables you to enter the official name for the host and every other name by which the host can be addressed by users. A request sent to any address on the list is mapped to the official host name. For details about the host identifier, see Chapter 3, "Configuring WebGates and Access Servers" on page 3-1.

It is possible to use the Host Identifiers feature to set up a host context for adding resources to the same policy domain on different computers. For details describing what a host context is and why you may want to use one, see "Using Host Identifiers and Host Contexts" on page 4-31.

- Rules and expressions for protection

  Rules for authentication determine how the identity of a user attempting to access a resource is to be proven. Authentication rules contain authentication schemes. Authorization rules determine whether the user has the right to access the resource. Authorization rules contain authorization schemes and are contained in authorization expressions. An authorization expression can contain one or more authorization rules. Auditing rules determine the information to be recorded in the audit log for operations pertaining to the policy domain or policy (audit). Auditing rules are derived from a Master Audit Rule. For details, see the following information:

  – For details about authentication rules and authentication schemes, see Chapter 5, "Configuring User Authentication" on page 5-1.

  – For details about authorization schemes, authorization rules, and authorization expressions, see Chapter 6, "Configuring User Authorization" on page 6-1.

  – For details about auditing rules and the Master Audit Rule, see "About Rules and Expressions" on page 4-23 and "About the Master Audit Rule" on page 4-34.

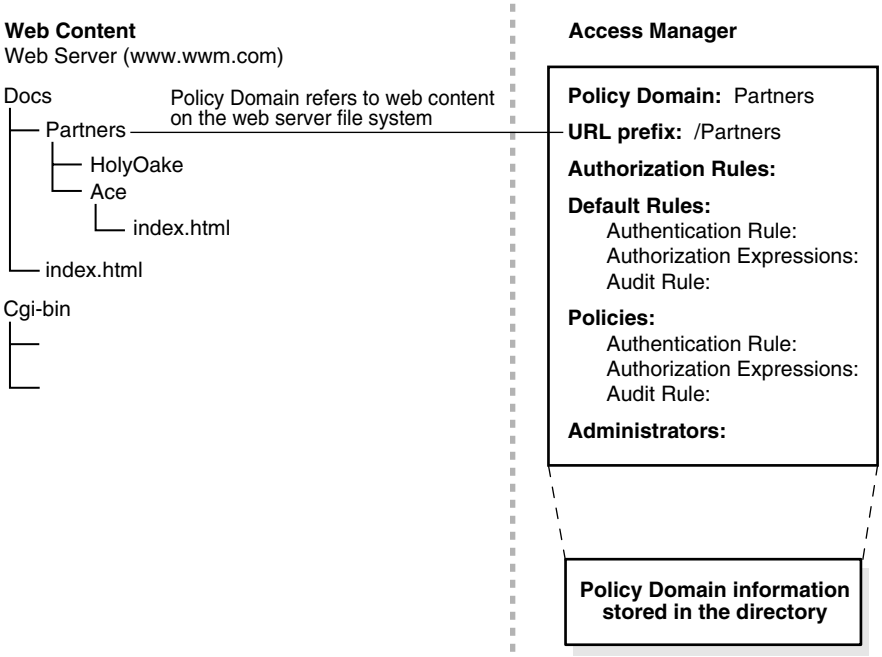
- Policies for URL patterns and the operations allowed for the type of resource to which the pattern applies

  Policies for resources within a policy domain allow you to create finer-grained ways to protect specific resources in the domain. You can specify a URL pattern or an explicit URL to identify resources. Different types of resources have their own operations. You can specify the operations—also known as request methods—that are allowed for resources of a type. Requests for resources whose URLs match the pattern are further processed against the rules of the policy.

  For details about policies, see "Configuring Policies" on page 4-38.

Figure 4–1 provides a conceptual view of the parts of a policy domain. In this figure, only Web-based resources are shown. However, Access System policy domains can also protect resources other than Web-based ones.

Figure 4–1 Policy Domain Structure



A policy domain can contain different types of resources, such as:

- An entire external Web site
- Specific pages in a Web site
- Partner portals
- A parts order application
- Invoice applications
- A benefits enrollment application on Web servers of an enterprise in many countries

For details describing resource types, see "Configuring Resource Types" on page 4-11.

How the Policy Domain or Policy for a Resource Is Determined

A resource may fit the definition of multiple policy domains. It may fall within a broad policy domain such as /mydomain. It may also fall within a more specific policy domain such as /mydomain/myresources. The Access Server checks all policy domain definitions to find the URL prefix that most closely matches the resource's URL.

Unlike the way that the Access Server checks policy domains, the Access Server checks policies in the order that you specified when you configured them. It uses the first matching policy regardless of how many more policies there are. The closest match to a requested resource may not be checked because a previous policy provided a match. For an intended policy to be used and for processing efficiency, consider the order that you assign to policies.

Preconfigured Policy Domains

You can configure policy domains to protect Identity System and Policy Manager resources (URLs) during installation. The policy domains created during installation are:

- **Identity Domain**: Protects Oracle Access Manager Identity URLs.
- **Access Domain**: Protects Oracle Access Manager Policy Manager URLs.

You must configure a WebGate and Access Server for these policy domains, and then enable or disable them together.

> **Tip:** See the *Oracle Access Manager Installation Guide* for information about configuring authentication schemes and policies during installation.

Who Creates Policy Domains?

Delegation of policy domain administration enables you to scale administration, enabling those closest to the resources and most knowledgeable about them to manage them. You can distribute policy domain creation and administration across administrators. Or, you can centralize policy creation and decentralize management and enforcement of it. A Master Access Administrator can create several policy domains and delegate administration of them to Delegated Access Administrators. The Delegated Access Administrators can manage the domains and create policy domains for resources whose URLs are more specific than those of the domain.

Examples

- A Web Master who maintains a corporate Web site is assigned the position of Delegated Access Administrator for the policy domain for a resource. The policy domain includes other resources the Web Master also manages.
- A Delegated Access Administrator may manage a particular resource, such as an enterprise-level application, for example, an airline's passenger check-in verification system. Instances of the application may run on many servers.

  Because a related, smaller application called Upgrade requires the same protection and is managed by the same administrator, both applications could belong to the same policy domain. Additionally, all instances of each application could be protected by the same policy domain.

> **See also:** For details about Master Access Administrators and Delegated Access Administrators, see "Delegating Policy Domain Administration" on page 4-46.

Examples of Policy Domains and Policies

Organizations use policy domains and policies for different purposes and in different ways.

Here are some examples of use policy of domains:

Policy Domain for Human Resources and Marketing: This example illustrates using policies to provide enhanced authorization for a resource in a policy domain. A policy domain protects human resources information made public to employees and a branch of the marketing Web site. Both sets of resources require the same kind of protection.

The following two URLs define the policy domain's logical structure:

```
/AreteAirlines/marketing/reports/
```

```
/AreteAirlines/HR/
```

If resources of either organization in the policy domain require protection rules that are more specific than the policy domain's default rules, policies can be defined to protect those resources. For example:

- The default Authorization rule for `/AreteAirlines/HR/` grants all users weekday access only.

  A policy could be defined to remove weekend access restrictions from a set of human resource files for human resources managers who tend to work on weekends.

- The same policy domain includes resources in a private directory that should be viewed only by regular employees, for example, analysts' reports.

  The private directory is subordinate to the reports directory. Resources in the private directory are protected by the default rules of the policy domain unless a policy is used to provide them different protection. A policy that restricts access to the resources in the private directory exists; it stipulates that only regular full-time employees may see the reports in the following private directory:

  ```
  /AreteAirlines/marketing/reports/private/
  ```

- The policy domain's URLs encompass a resource that is an application called badgit. The application enables HR employees to register employees of the organization for access badges. The main application processes the request and obtains information from a backend application. A policy is used to protect only this application. The policy applies to the following specific URL:

  ```
  /AreteAirlines/HR/badges/badgit.exe
  ```

Policy Domain for a University: This example illustrates the use of policies to provide relaxed authorization requirements for a resource in a policy domain. A university provides information to its students, but not to outsiders. The URL for the policy domain protecting the resources is:

```
/GlobalUniv/
```

Two policy domains with more specific URLs are created to include resources otherwise covered by the `/GlobalUniv/` policy domain.

- One of these policy domains includes the URL `/GlobalUniv/physics/`.
- The other policy domain includes the URL `/GlobalUniv/philosophy/`.

The policy domain /GlobalUniv/physics/ allows only physics students to access the policy domain's resources.

- All physics students can access resources in the /GlobalUniv/physics/feynman/diagrams/ directory because the default rules of the `/GlobalUniv/physics/` policy domain apply, and there are no specific policies applied to these resources.
- A policy can be created to allow only those students who meet the authorization criteria of the policy protecting the testResults.html page to see it. The students who took a quiz may be able to view the following Web page:

  ```
  /GlobalUniv/physics/feynman/diagrams/testResults.html
  ```

The college presents a suite of applications animating the world of black holes. The applications are available to all students, not just physics students. The URL for one of

these Enterprise JavaBean (EJB) applications is `/GlobalUniv/physics/wheeler/blackHoles/explore/styx.ejb`.

Because the application is in a directory called wheeler, which is subordinate to the physics directory, a policy must be used to remove access to the wheeler directory, providing the resources to all students.

About Allocating Responsibility for a Policy Domain

You can assign administrative roles and give the administrators responsibilities for managing policy domains for resources of the same or a different type on the same or a different host. It is a good idea to define policy domains along the lines of the resources they protect and who manages them. Who can access them is secondary, and is expressed through the access control rules of the policy domain.

The following are some examples why it may be useful to decentralize management of resources:

- You want to provide your employees with faster and better service for your online applications.

  For example, improved service helps to make applications more readily available initially and more easily recoverable if a failure of the host system occurs.

- You may want to keep your informational Web sites for employees operational and current with as little disturbance as possible.

Configuring Resource Types

A resource type describes the kind of resource to be protected, including its associated operations. Operations associated with a resource are tied to its type.

Before you can add resources to a policy domain, you must define their types and the operations associated with them that you want to protect.

The Access System provides several default resource types. A Master Access Administrator can define other resource types from the Access System Console.

By giving you the ability to define more resource types than the default ones provided, the Access System enables you to protect more than just Web-based resources.

The rest of this section discusses the following topics:

> **Note:** You can configure custom AccessGates to protect non-Web resources. See *Oracle Access Manager Developer Guide* for details.

About Resource Types

The Access Manager expects resources to be directory or file level resources, for example:

```
http://server.domain.com:port/this/path/to/the/app?param1=x&param2=y
```

Any URL that contains query strings is escaped and the entire URL is considered a directory or named resource. In the previous example, the string app?param1=X&param2=Y is escaped and treated differently from other types of URLs. For example, suppose a user requests the following resource:

```
http://server.domain.com:port/this/path/to/the/app?param2=y&param1=x
```

In theory, this is the same resource as the previous one. However, the policy domain evaluation performed by the Access System will miss this resource, and will fail to protect it. Similarly, if the user requests the following URL, the Access System will miss this resource, and will fail to protect it:

```
http://server.domain.com:port/this/path/to/the/app?param1=x&param2=y&param3=z
```

In these cases, you should define the policy domain for the resource at the URL path level, for example:

```
http://server.domain.com:port/this/path/to/the/app
```

You can add one or more policies under the policy domain to handle more specific URLs. A policy can contain a query string pattern of *param1=X&param2=Y* or separate query string value pairs.

Default Resource Types

The Access System provides the following resource types:

- Enterprise Java Beans (EJB): The EJB resource type is not required.

  You can delete it if you do not plan to protect EJB resources.

- HTTP (HTTPS): The HTTP resource type definition is required, whether or not you protect resources of this type.

  You cannot delete the HTTP resource type or modify its operations.

You must define a type for any other kinds of resources that you want to protect.

Supported HTTP Operations

The Access System supports the following HTTP operations:

- **CONNECT**: Handshakes with a URL
- **DELETE**: Deletes information from the URL, or deletes the URL itself
- **GET**: Retrieves information from the URL
- **HEAD**: Obtains information about the resource without making changes to the URL
- **OPTIONS**: Obtains information about HTTP methods available to and from the URL
- **OTHER**: Non-standard, custom operation
- **POST**: Copies information to the URL
- **PUT**: Replaces a file or document in the URL
- **TRACE**: Views information about what the URL is receiving

Supported EJB Operation

The Access System supports the EJB EXECUTE operation, which executes a bean. You can add other EJB operations.

Supported Resource Types

A policy domain can protect these types of resources and their operations:

- EJB Resources

  EXECUTE

- HTTP Resources

  GET, POST, PUT, TRACE, HEAD, CONNECT, OPTIONS, and others

  You can define policies to protect a specific operation.

- RDBMS Resources

  ADD, DELETE, and UPDATE

- Servlet resource types

Table 4–1 shows examples of HTTP resources, Java 2 Enterprise Edition (J2EE) resources, and other online application resources identified by their URLs.

Table 4–1 Application Resources and Their URLs

Resource Type	Examples
HTTP Resources	■ Directories /mydirectory ■ Pages /mydirectory/index.html ■ Web applications /applications/myexe.exe ■ Query strings `www.wwm.com/sales/result/pricelist1,2,0-a-00,000.htm?st.dl.search.qs.results`
J2EE Application Server Resources	■ Java Server Pages JSPs ■ Servlets ■ Enterprise Java Beans
Other Resources	■ Standalone programs Java, C, C++ application programs ■ ERP applications ■ CRM applications

Defining a Resource Type

To define a resource type, use the Define a New Resource Type page.

To define a resource type

1. From the landing page for the Access System, click the Access System Console link.

If you are working with the Policy Manager, click the link for the Access System Console at the top of the page.

2. From the Access System Console, click the Access System Configuration tab, then click the Common Information Configuration link in the left navigation pane.
3. Click the Resource Type Definitions sub-tab.

   The List All Resource Types page appears.
4. On the List All Resource Types page, click Add.

   The Define a new Resource Type page appears.

5. In the Resource Name field, enter a unique name for the new resource type.
6. In the Display Name field, enter the name of the resource type.
7. In the Resource Matching field, specify whether the resource type can be read as case sensitive or case insensitive.
8. In the Resource Operation field, specify the operations this resource type can perform.

   You can define custom operations, but not for HTTP resource types.

   To add or delete fields as necessary, click the plus (+) and minus (–) icons.
9. Click Save to save your changes (or click Cancel to exit the page without saving).

Configuring URLs for Resources

To use the Access System to protect your resources—for example, your business applications and content—you must create a policy domain whose URL prefixes and URL patterns identify the resources.

URL Prefixes: You use URL prefixes to define the policy domain content. For policies, you use URL patterns to identify resources protected by the policy.

You can create URL prefixes that define a broad scope of content, for example:

```
/
/sales
```

```
/humanresources
```

In this example of a policy domain, the resources to be protected exist on three different hosts. All resources under the URL prefixes are protected by the default rules of the policy domain:

Policies: You can create policies for resources within a policy domain. For example, resources for two other groups reside under /. They are engineering and marketing.

- Because no policy is defined for /engineering, its resources are still protected by the default rules of the policy domain. Default rules also apply to marketing.
- After the administrator creates a policy for resources under /engineering, the engineering resources are protected by the rules specified by the policy and not the default rules of the policy domain.

URL Patterns: You can create policies with granular URL patterns. Here is an example of a URL pattern:

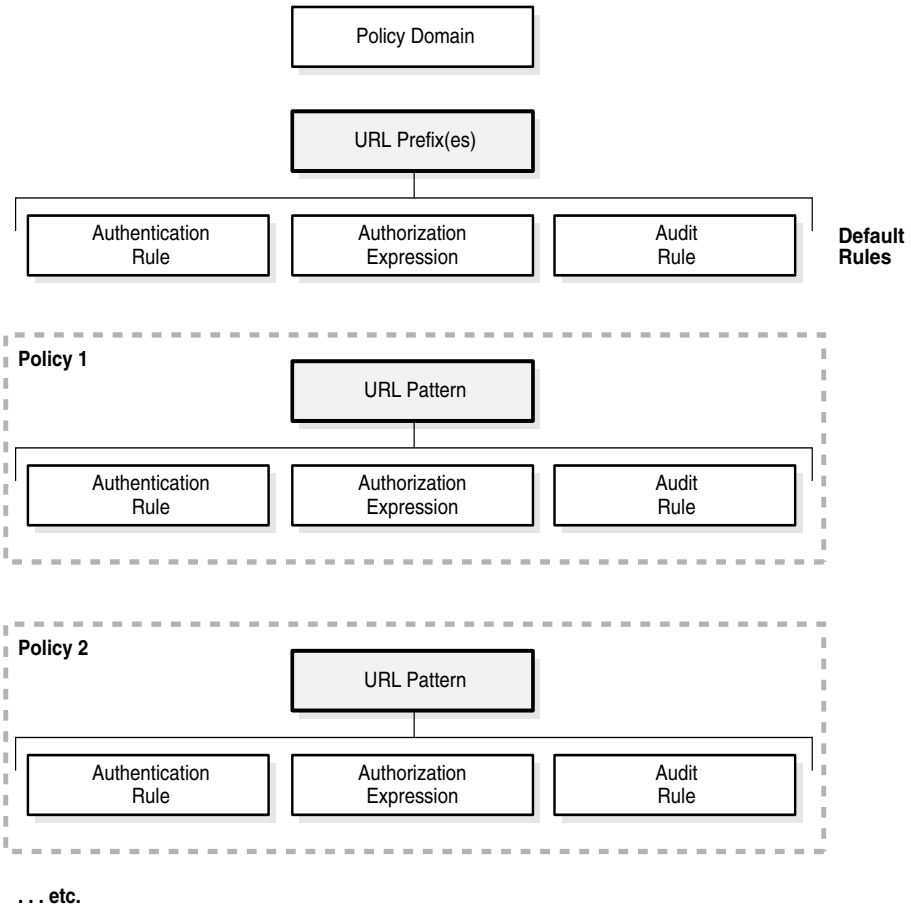
```
/.../update.html
```

This URL pattern matches these resources:

```
/humanresources/benefits/update.html
/corporate/news/update.html
update.html
```

Figure 4–2 illustrates how URL prefixes and URL patterns are used to define the resources for policy domains and their policies.

Figure 4–2 URL Prefixes and Patterns



For more information, see the following topics:

- About URL Prefixes
- How URL Patterns are Used
- URL Pattern Matching Symbols
- Invalid Patterns
- Access System Patterns

About URL Prefixes

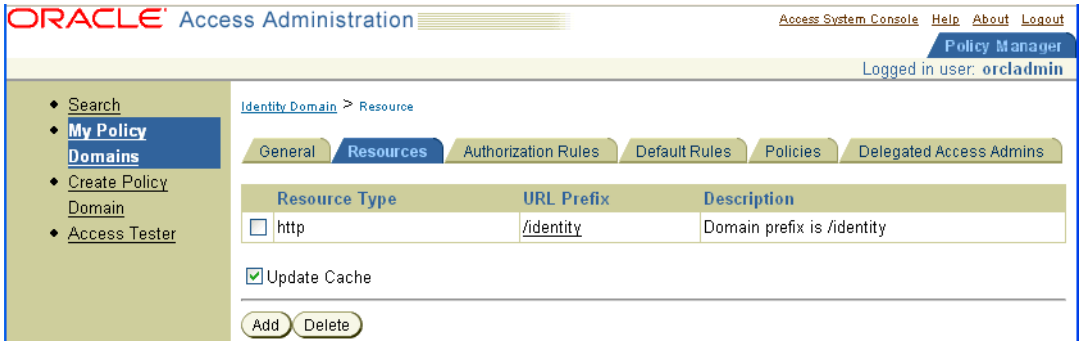
The URL prefix is the starting point for resources in a policy domain. A URL prefix defines the beginning boundary of a policy domain, that is, its first resource. A URL prefix maps to a directory on the file system of one of your application servers or Web servers.

All resources under a URL prefix are protected by the default rules of a policy domain unless more specific rules are applied to them through policies. You can assign one or more URL prefixes to a policy domain, but each URL prefix can belong to one policy domain only.

The trade-off in creating many granular policy domains is that you achieve greater security at the cost of increased overhead. The cost is incurred because the Access

Server must evaluate all policy domains to find the one that is most specific to the resource. Use of policies affords you the same benefit without the overhead.

The following is a screen shot of a page where you define a primary URL prefix for a resource, as described in "To add resources to a policy domain" on page 4-31. This screen shot shows a policy domain that protects the Identity System:



Process overview: How a URL prefix is used

1. An end user requests a resource by specifying the URL for the resource in a browser.

   For example, a user enters the following URL in her browser to request access to a data page displaying information about a specific corporate partner:

   ```
   www.AreteAirlines.com/Partners/mycorp.html
   ```

   If the user's own Web site is set up accordingly, the user may select a link which represents the resource (and the URL for it).

2. The Access System locates the requested resource.

   The Access Server assesses all of the policy domains to ascertain the one having the URL prefix most specific to the incoming URL for the resource. (The Access Server determines if the resource is covered by a policy within the domain, whose rules would then apply.)

3. If no policy applies, the Access System uses the rules of the policy domain to determine whether to allow or deny the user access to the HTML page.

A policy domain can protect content other than Web-based content, although the policy domain in Figure 4–2 covers Web-based resources.

You can specify individual policies for resources of a given type whose URLs match a URL pattern. You can also specify the kinds of operations that can be performed on the resources.

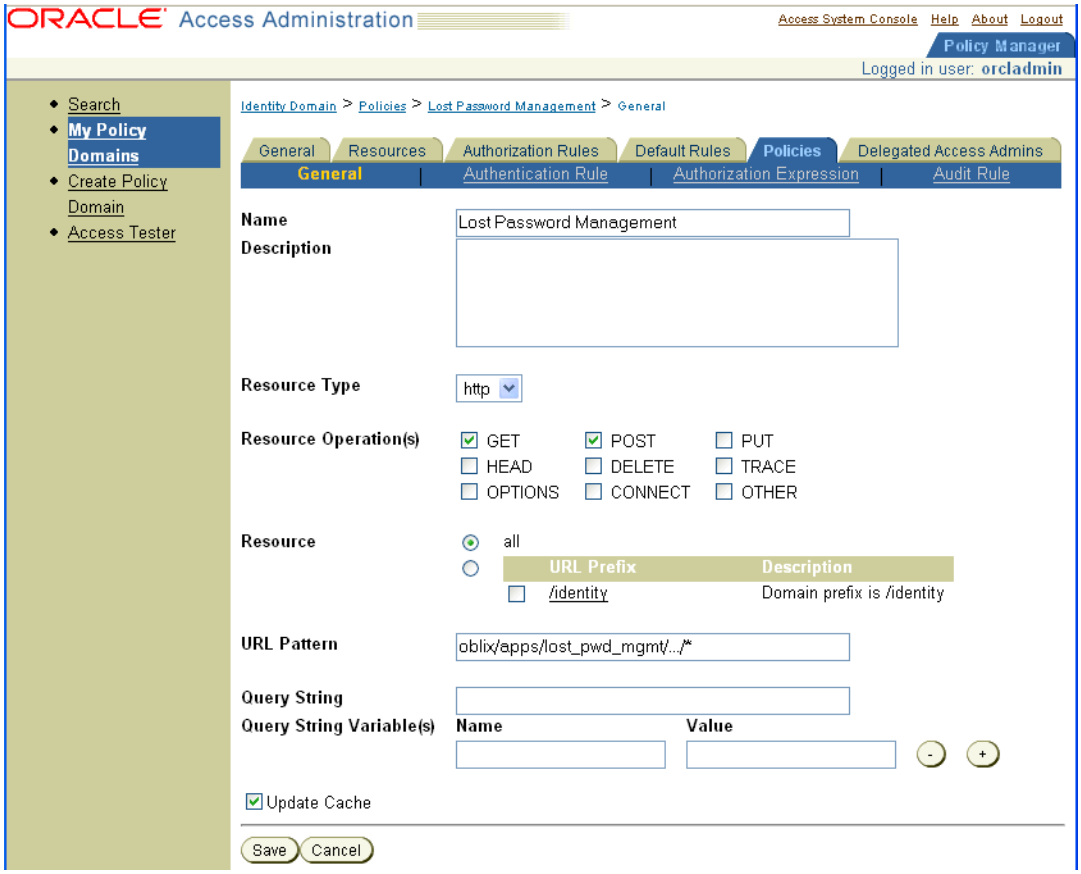
About URL Patterns

A URL pattern is an Access System-supported mechanism for identifying different resources of a certain type that are protected by a single policy. A URL pattern can be a directory, query string pattern, or query string variable. If it is an explicit fully qualified URL, then it refers to a single resource.

An example of a URL pattern that covers many resources is a URL for all HTML pages (*.html) of a department's Web site. In this case, the policy may remove restrictions imposed by the policy domain's default rules. An example of a URL pattern for a specific file is an explicit fully qualified path (URL) of a single instance of an

application. Resource operations are the functions available for each configured type of resource. For example, HTTP has GET, POST, PUT and other operations.

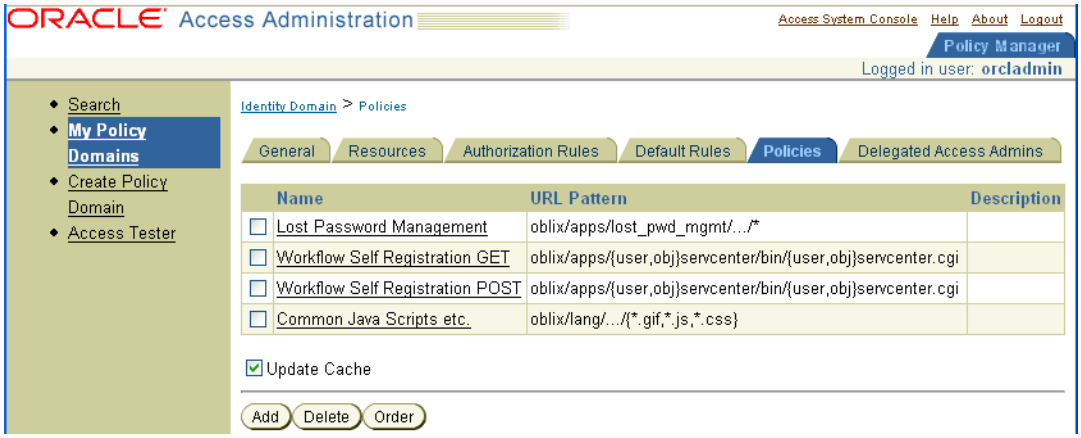
The following is a screen shot of a page where you configure a URL pattern for a fine-grained policy, as described in "Adding a Policy" on page 4-39. In particular, the following screen shot shows a policy that protects the Lost Password Management application in the Identity System. This policy is configured for the Identity domain:



When appended to the URL prefix of /identity, the URL pattern shown in the previous screen forms a pattern that protects resources located in the following URL path:

/identity/oblix/apps/lost_pwd_mgmt/.../*

The following screen illustrates the result of defining multiple URL patterns for a policy domain. In particular, the screen shot shows a list of policies and their URL patterns in the Identity domain. These URL patterns protect various applications and data directories in the Identity System:



How URL Patterns are Used

URLs for policies specify the fine-grained portion of a resource's namespace. To fully identify the URL, the host identifier and URL prefix for the policy domain are concatenated with the policy's URL pattern.

Process overview: How URL patterns are used

1. A user specifies the URL for a requested resource.
2. Based on the policy domain's host and URL prefix information, the Access Server creates a fully qualified URL that includes the URL pattern.
3. The Access Server compares the incoming URL for the requested resource to the fully qualified URL constructed from the policy domain information and the policy's URL pattern.
   - If there is a match, the policy's various rules are evaluated to determine whether the requester should be allowed or denied access to the resource.
   - If requester is allowed access, the resource is served to the user.

Figure 4–1 shows the structure of a policy domain called Partners that includes the following URL pattern:

```
/Ace/.../*
```

To get the fully qualified name of the URL pattern for the policy, the policy domain's URL prefix, /Partners, is prepended. The name of the host where the resources of the policy domain reside is specified before the URL prefix, resulting in the following URL:

```
myhost/Partner/Ace/.../*
```

For information on configuring a policy and its URL patterns, see "To add a policy" on page 4-39.

URL Pattern Matching Symbols

The Access System expresses URL patterns through a type of filtering called globbing. This filtering method combines different UNIX shell (sh, csh or tcsh) support for patterns in file names with Access System-provided patterns such as "..." (three periods), which let you span multiple directories.

Table 4–2 shows the supported patterns.

Table 4–2 Supported patterns

Pattern	Description	Examples
?	Matches any one character other than /.	a?b matches aab and azb but not a/b.
*	Matches any sequence of zero or more characters. Does not match /.	a*b matches ab, azb, and azzzzzzb but not a/b.
[set]	Matches one from a set of characters. A set can be specified as a series of literal characters or as a range of characters. A range of characters is any two characters (including -) with a hyphen (-) between them. The forward slash character (/) is not a valid character to include in a set. A set of characters will not match / even if a range that includes / is specified.	■ [nd] matches only n or d. ■ [m-x] matches any character between m and x, inclusive. ■ [--b] matches any character between - and b inclusive (except for /; see /usr/pub/ascii for order of punctuation characters). ■ [abf-n] matches a, b, and any character between f and n, inclusive. ■ [a-f-n] matches any character between a and f inclusive, -, or n. (The second - is interpreted literally because the f preceding it is already part of a range.)
{pattern1, pattern2,...}	Matches one from a set of patterns. The patterns inside the braces may themselves include any other special characters except for braces (sets of patterns may not be nested).	■ a{ab,bc}b matches aabb and abcb. ■ a{x*y,y?x}b matches axyb, axabayb, ayaxb, and so on.
/.../:	Matches any sequence of one or more characters that starts and ends with the forward slash character (/).	■ The pattern /.../index.html matches: /index.html /oracle/index.html /oracle/sales/index.html index.html It does not match xyzindex.html or xyz/index.html. ■ /oracle/.../*.html matches: /oracle/index.html /oracle/sales/order.html, and so on
\	The backslash character is used to escape special characters. Any character preceded by a backslash matches itself.	■ abc*d only matches abc*d ■ abc\\d only matches abc\d

Invalid Patterns

A URL pattern is an Access System-supported mechanism for identifying different resources of a certain type that are protected by a single policy. Patterns with the following attributes are invalid:

Patterns with the following attributes are invalid:

- A '[' without a closing ']'
- A '{' without a closing '}'
- Unescaped '{' inside {}
- Unescaped '/' inside [ ]

The following URL pattern is not recognized when it is included within {}:

```
{pattern_1, pattern_2, /.../cleanup.asp}
```

The URL pattern will only be recognized if it is used without {}:

```
/.../cleanup.asp
```

URL patterns within {} are designed for simple expressions such as the following:

```
a{ab,bc}b matches aabb and abcb
a{x*y,y?x}b matches axyb, axabayb, ayaxb, etc
```

URL patterns within {} should not contain complex sub-expressions such as those starting with "/". For example:

```
{/.../cleanup.asp   OR    /c*/webservice/webservice.asp}
```

Instead, consider creating three separate policies:

```
??/admin/*
/c*/webservice/webservice.asp
/.../cleanup.asp
```

For information on configuring a policy and its URL patterns, see "To add a policy" on page 4-39.

Access System Patterns

A policy can contain one or more of the following types of patterns. If multiple patterns are specified in one policy, they *all* must match to the incoming URL. If they do not, the policy does not apply to the URL.

This example uses the following incoming URL:

`http://www.myserver.com/oblix/sales/index.html?user=J.Smith&dept=sales`

The policy includes the following URL patterns:

- Pattern for the absolute path of the URL

  This pattern is the part of the URL that does not include the scheme (http) and host/domain (www.myserver.com), and that appears before a question mark, ?, character. In this example, the absolute path is: `/oblix/sales/index.html`.

- Pattern for name value pairs in the URL

  A set of these pairs may be configured as a pattern. The pairs apply to query data that appears after the question mark, ?, character in the URL—if the operation is GET. If the operation is POST, query data appears after the POST data. For a pair, name specifies a name value, not a pattern. The value element of the pair is configured as a pattern. For example:

Name	Pattern
user	*Smith
dept	*sales*

  If multiple name-value pairs are specified, they all must match the incoming URL. Therefore, the following URL does not match the pattern:

```
http://www.myserver.com/oblix/sales/index.html?user=J.Smith
&dept=engg
```

The important difference between this pattern and the next one is that there is no priority to these name-value pairs. The following URL satisfies the pattern:

```
http://www.myserver.com/oblix/sales/index.html?dept=sales&use
r=J.Smith
```

Note the reverse order of "dept" and "user". This is important and useful because it is commonly difficult to control the order of name-value pairs in the GET/POST query data.

- Pattern on the entire query string:

  This is useful if you want to enforce an order on the query string. For example, a pattern:

  ```
  user=*Smith*sales*
  ```

  matches the query string

  ```
  user=J.Smith&dept=sales
  ```

About Schemes

Schemes allow the Master Access Administrator to define methods that are used to authenticate users and to verify a user's right to access a resource. Schemes are reusable templates. You create schemes in the Access System Configuration area of the Access System Console.

An *authentication scheme* contains one or more steps, each of which can include one or more plug-ins. A policy domain must have at least one authentication rule and therefore one authentication scheme.

An *authorization scheme* is included in an authorization rule. You can use the default authorization scheme, or you can provide a custom one. A policy domain must have an authorization expression containing at least one authorization rule.

After you define a scheme, Delegated Access Administrators for different policy domains can use the same scheme in rules for their domains or in rules for policies within their domains.

You can define all of the schemes you and your Delegated Access Administrators will need for policy domains and policies at one time. Or, you can define schemes as they are required.

About Plug-Ins

Plug-ins are dynamically loaded shared libraries executed to perform authentication and authorization processes. They are contained in schemes, and they are used to request and process the information necessary to authenticate a user or authorize a user to access a resource.

Plug-ins perform specific tasks. For example:

- **For Authentication Schemes:** Authentication schemes contain one or more steps. The steps of an authentication scheme contain its plug-ins. The Access System provides default plug-ins, or you can provide custom ones. For example, every

chained authentication scheme must have a plug-in that maps information obtained from the user—user credentials—to user profile information.

Also, every authentication scheme includes a challenge method plug-in and a password verification plug-in. The Access System provides plug-ins for these purposes, too. If you do not use the plug-in provided for this purpose, you must replace it with one that provides the same functionality.

If you want to replace the provided plug-ins with custom ones, you must design your plug-ins to perform required tasks, to accept and pass required parameters, and to return defined function codes. For details, see "Plug-Ins for Authentication" on page 5-18.

- **For Authorization Schemes**: For authorization rules, you can use the default authorization scheme provided by the Access System, or you can use a custom one. If you want to use a custom authorization scheme, you must provide your own plug-ins for it. For details, see "Configuring User Authorization" on page 6-1.

You can create plug-ins for the following supported platforms:

- **MS Windows**: A dynamic link library (.dll) is used to implement shared libraries.
- **UNIX**: A shared library (.so) is used to implement shared libraries.

For information describing how to create custom plug-ins, see the *Oracle Access Manager Developer Guide*.

About Rules and Expressions

Rules contain schemes that define how the resources of a policy domain are to be protected, including:

- How authentication of the user is to be performed.
- Whether a user has the right to access a domain resource and any conditions defining access rights. Authorization rules are included in authorization expressions. A policy domain must contain one—and only one—authorization expression.
- Events to be audited pertaining to the policy domain or policy.

You include one or a combination of authorization rules to form an authorization expression. Rules can include actions to be executed depending on the result of the evaluation of user information against the specifications of the rule.

A policy domain can include policies, which can contain their own rules and authorization expressions. Therefore, a policy domain can contain two levels of rules:

- Those that apply by default to all resources of the policy domain.
- Those that are part of a policy and apply to specific resources within the domain. These policy rules override the default rules of the policy domain for the resources they protect.

Authorization expressions include authorization rules and the operators used to combine them. You combine rules within expressions to create from simple to complex means of specifying who is allowed or denied access to the protected resources.

Authorization rules are reusable within a policy domain. You can use the same rule in an authorization expression more than once. Also, you can use the same rule in the expression for the policy domain and in expressions for any of its policies.

Table 4–3 defines the four types of rules for a policy domain or a policy.

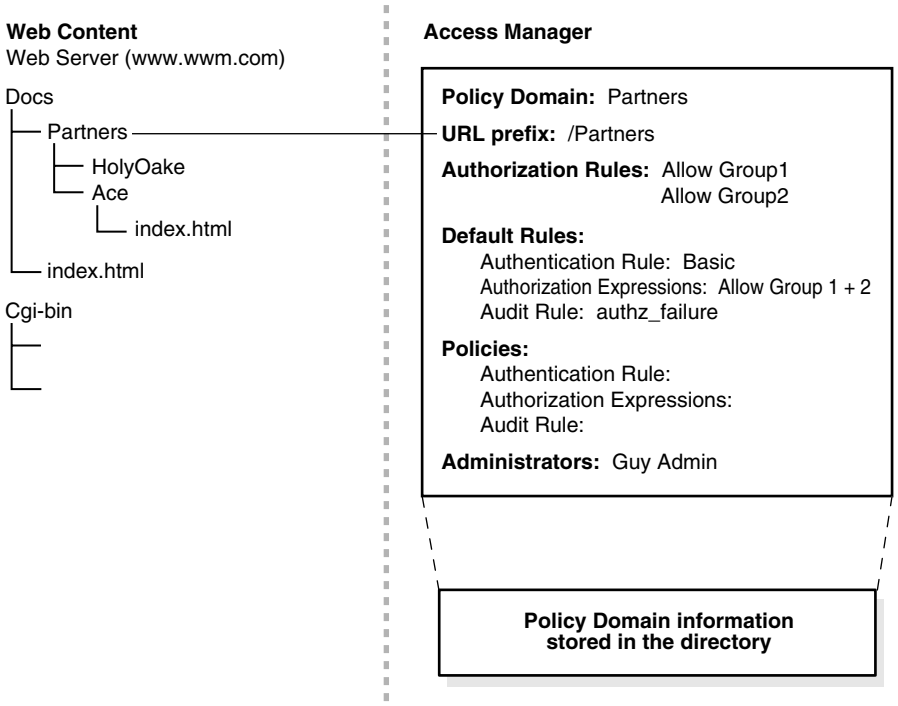
Table 4–3 Types of Rules

Rule	Description
Authentication Rule	Specifies the method used to challenge and authenticate users requesting access to protected resources. Can specify actions to be taken if authentication is successful or if it fails. **Note**: Only one default authentication rule can be included in a policy domain. Each of its policies, however, can have its own authentication rule.
Authorization Rule	Allows or denies a user access to requested resources within a policy domain or policy. Can specify actions to be taken if authentication is successful or if it fails. These rules are included in authorization expressions. **Note**: If multiple rules are included in an expression, the order of evaluation of the rules is determined by the logic you specify to form the expression. Can also specify conditions for access. **Note**: Only one default authentication rule can be included in a policy domain. Each of its policies, however, can have its own authentication rule.
Authorization Expression	Includes authorization rules. A policy domain must have one and only one authorization expression. **Note**: A policy within a domain can have a single authorization expression. If it does not include one, the resources of the policy are protected by the authorization expression of the policy domain.
Audit Rule	Captures attributes and information about specific events pertaining to the policy domain. ▪ It modifies and overrides events and information specified in the Master Audit Rule. ▪ If no specific audit rules are applied, the Master Audit Rule is enforced by default.

Note: Authentication rules are applied before authorization rules because a user's identity must be proven before he or she is granted access to a resource.

Figure 4–3 illustrates a policy domain containing a default set of rules and a default authorization expression applied to the domain's resources. For the resources defined by the policy, the default rules and expression are overridden by those of the policy.

Figure 4–3 Rules and Authorization Expression for a Policy Domain and Policy



Lessening or Increasing Controls with Rules

By default, the Access System allows access to a resource that is not explicitly protected by a policy domain rule or a policy. You can begin to create policy domains from this condition—all resources unprotected. You can take the opposite position and reverse the default state so that all resources are protected at the outset.

Beginning with All Resources Unprotected

If you begin to create policy domains from a position in which all resources are unprotected, you must apply access controls to those resources. You can do this at a broad level by creating policy domains with default rules which are more or less restrictive:

- If you use restrictive default rules to impose tight controls across all resources of a domain, you can use policies to remove or change restrictions for subgroups of resources.
- If you use lenient default rules as a starting point, you can use policies to provide tighter, specific controls on subgroups of resources within a domain.

Beginning with All Resources Protected

To start from a state in which all resources are protected, you set the parameter DenyOnNotProtected. If this switch is set to true, DenyOnNotProtected denies access to all resources not explicitly allowed by a policy domain's rules or policies.

If all resources are protected, you must create policy domains and policies to remove protection from those resources you want to make available to various users. In this sense, you are uncovering resources to a greater or lesser degree to make them available.

You can do this at a broad level by providing default rules for a policy domain:

- If you use lenient default rules to lessen controls across all resources of a policy domain, you can use policies to apply particular restrictions for subgroups of resources.
- If you use tighter default rules as a starting point—perhaps rules that are stringent but less so than the current default state of complete denial of access—then you can use policies to lessen access control for subgroups of resources in various ways.

> **Note:** If DenyOnNotProtected is set to false, this switch allows access to all resources not explicitly denied by a policy domain's rules or policies.

For information describing how to use the DenyOnNotProtected switch, see Chapter 3, "Configuring WebGates and Access Servers" on page 3-1.

Creating and Managing Policy Domains

This section describes how to create policy domains, enable or disable them, and manage their resources. It addresses the following set of tasks:

- Creating a Policy Domain
- Modifying a Policy Domain
- Deleting a Policy Domain
- Enabling and Disabling Policy Domains
- Searching for Policy Domains and Policies
- Viewing General Information about Policy Domains
- Adding Resources to Policy Domains
- Modifying a Resource's Description
- Deleting a Resource

Creating a Policy Domain

Both Master Access Administrators and Delegated Access Administrators can create policy domains. Master Access Administrators can create policy domains at any level. Delegated Access Administrators can create policy domains that are subordinate to any policy domains delegated to them for administration.

You use the Policy Manager to create policy domains, add resources to a domain, and protect the resources using authentication rules and authorization rules and expressions.

> **Note:** By default, a policy domain is not enabled. Do not enable a domain until you have added resources to it. Be aware that if you enable a policy domain that does not contain resources, the domain cannot be used.

See Also: "Task overview: Prerequisite tasks for a Master Administrator" on page 4-2

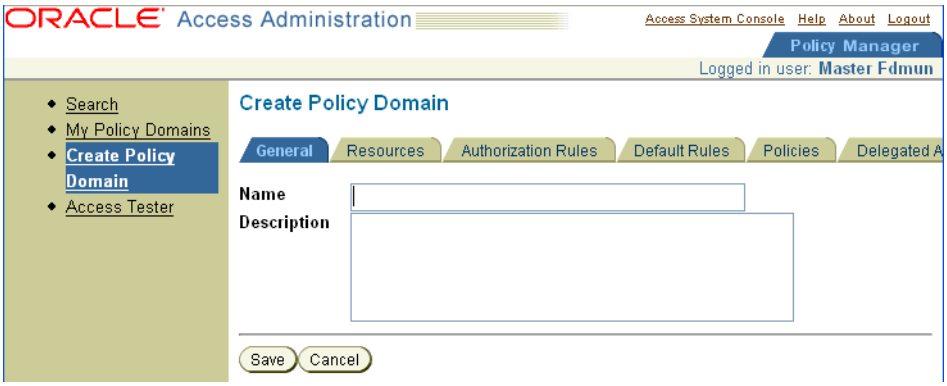
To create a policy domain

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

   If you are working with the Access System Console, click the link for the Policy Manager at the top of the page.

2. From the Policy Manager, click Create Policy Domain in the left navigation pane.

   The Create Policy Domain page appears, as illustrated in the following.



3. General: Fill in the name and optional description:

   - In the Name field, enter a short alphanumeric string identifying the domain.

     You can use spaces in this field.

   - In the Description field, type a brief description of this policy domain.

     The Name and Description appear in pages showing lists of policy domains. A description is optional.

4. Click Save.

   - To view currently defined information about your policy domain, click View as Page.
   - To return to the General page, click the name of your domain at the upper left part of the View as Page page.

5. Resources: To add a resource to this policy domain, see:

   - Configuring Resource Types
   - Configuring URLs for Resources
   - Adding Resources to Policy Domains

   See Also: "Defining a Resource Type" on page 4-13 if you need to add a new resource type to this policy domain. You must use the Access System Console to define a resource type, then return to the Policy Manager to add it to the policy domain.

6. Authorization Rules: From the Authorization Rules tab you can add and enable a rule to use in Authorization Expressions later. Each rule includes general

information, timing conditions, actions, and access (Allow or Deny) conditions. For more information, see "About Rules and Expressions" on page 4-23.

See Also: Chapter 6 for details about configuring authorization rules.

7. Default Rules: From this tab you add the authentication rule, authorization expression, and audit rule for this policy domain.

   Authentication Rule: A policy domain must have at least one authentication rule, which specifies one authentication scheme and authentication actions. For more information, see "Managing Authentication Rules" on page 5-48.

   See Also:

   - "Managing Authentication Actions" on page 5-53
   - "Creating an Authentication Rule for a Policy Domain" on page 5-49

   Authorization Expression: These include authorization rules and the operators used to combine them. For more information, see "About Authorization Expressions" on page 6-14.

   See Also: Chapter 6 for details about configuring authorization expressions.

   Audit Rule: If there is no Master Audit Rule defined, you are instructed to contact your Access System Administrator.

   See Also:

   - "About the Master Audit Rule" on page 4-34
   - "Creating an Audit Rule for a Policy Domain" on page 4-42

8. Policies: From the Policies tab you can add a policy to protect resources within the policy domain.

   Before setting up a policy, decide the level of access control needed for the URL you will protect. For each policy you create, you can assign a specific authentication rule, authorization expression, and auditing rule. If no rules are defined, the default rules for the policy domain remain in effect. You can create policies with granular URL patterns.

   See Also:

   - "Configuring URLs for Resources" on page 4-14
   - "How URL Patterns are Used" on page 4-19
   - "Adding a Policy" on page 4-39

9. Delegated Access Admins: "Delegating Policy Domain Administration" on page 4-46.
10. Enabling the Policy Domain: When you are ready to enable a new policy domain, see "Enabling and Disabling Policy Domains" on page 4-29.

Modifying a Policy Domain

You can modify a policy domain after creating it. Modifying a policy domain includes changing any aspect of it— adding or removing resources, and modifying, removing, or adding rules.

Be sure to disable the policy domain before you modify it. See "Enabling and Disabling Policy Domains" on page 4-29 for details about enabling and disabling policy domains.

To modify a policy domain

1. From the Policy Manager, click My Policy Domains in the left navigation pane.

   The My Policy domains page appears, displaying a list of policy domains.

2. Select the check box before the name of the policy domain you want to modify and click the domain's name.
3. On the general page, click Modify at the bottom of the page.
4. Change any values you want to modify, and click Save.

Deleting a Policy Domain

You can delete a policy domain entirely without first removing its resources and rules. Before you delete a domain, disable it. See "Enabling and Disabling Policy Domains" on page 4-29 for details.

To delete a policy domain

1. From the Policy Manager, select My Policy Domains.

   The My policy domains page appears, displaying a list of policy domains.

2. Select the check box before the name of the policy domain you want to delete.
3. Click Delete at the bottom of the page.

Enabling and Disabling Policy Domains

You must *enable* a policy domain before you can use it. You must *disable* a policy domain before you can modify its configuration.

The Access System provides two default policy domains that protect /identity and /access URLs. Both of these default policy domains must be enabled for the Access System to operate correctly. Only disable the default domains to modify them. Re-enable them after you have finished modifying them.

Important: Disable a domain before modifying its rules or policies.

To enable a policy domain

1. From the Policy Manager, select My Policy Domains.
2. In the My Policy Domains page, select the check box next to the domain you want to enable.
3. Click Enable.

   Yes appears in the Enabled column.

To disable a policy domain

1. From the Policy Manager, select My Policy Domains.
2. In the My Policy Domains page, select the check box next to the domain you want to disable.
3. Click Disable.

   No appears in the Enabled column.

Searching for Policy Domains and Policies

You can search for and display existing policy domains and policies. Master Access Administrators can search for and see all policy domains and policies. Delegated Access Administrators can see only the policy domains for which they have been delegated administrative rights. For their policy domains, they can also see the policies which they have defined along with those defined by a Master Access Administrator.

You use the Search function to search for policy domains and policies.

To search for existing policy domains or policies

1. From the Policy Manager, click Search in the left navigation pane.

   The Search window appears.
2. In the Search list at the top left of the page, select either Policy Domain Name or Policy.
3. Select an entry from the list of search criteria in the middle, then type a text string in the right column.

   To find all entries that match the selected search criterion, leave the right column blank.
4. Click Go.

   The results display on your page.

Viewing General Information about Policy Domains

You can display a list of policy domains and view configured information for an individual domain. The My Policy Domains page displays a list of domains for which you have administrative rights. Master Access Administrators can see information about all policy domains. Delegated Access Administrators can see only the policy domains for which they have been delegated management.

To view policy domains and configuration information

1. From the Policy Manager, click My Policy Domains in the left navigation pane.
2. Click the domain's link to view a domain's configuration settings.

   The General page displays the name and description of the policy domain and whether or not it is enabled. You can click other tabs to view configured information.

Adding Resources to Policy Domains

The Access System defines some resource types by default. A Master Access Administrator can define others. After a resource type is defined, both Master Access Administrators and Delegated Access Administrators can add resources of that type to

policy domains they administer. When a Delegated Access Administrator is granted administrative rights for a policy domain, that administrator can add resources to the domain.

Using Host Identifiers and Host Contexts

The Master Access Administrator defines host identifiers on the List All Host Identifiers page (Access System Console, Access System Configuration, Host Identifiers link). See "ObPERM cookie" on page 3-59 for details.

When you add a resource to a policy domain, you select the host identifier for the computer hosting the resource. If the Master Access Administrator has configured host identifiers for computers, you can select the appropriate one from the list labeled Host Identifiers on the Resource (add) page.

You can use the Host Identifiers feature to create a host context. A host context consists of multiple hosts identified in relation to a single name, a host context name. Instead of adding to a host identifier name the various ways to reference one host, the Master Access Administrator can add corresponding information for multiple hosts to create a context in which all of these hosts share.

A host context is useful if you want to add to a policy domain resources that have the same URL paths on different computers.You want to protect all of these resources in the same way in the same policy domain. In this case, the only variable that distinguishes one set of resources from another is identification of its host computer. Use of a host context provides an efficient way to add the resources for all hosts to the policy domain at once. From the Host Identifiers list, you select the host context name. The rest of the information you enter is the same for all of the sets of resources, so you need only specify it once.

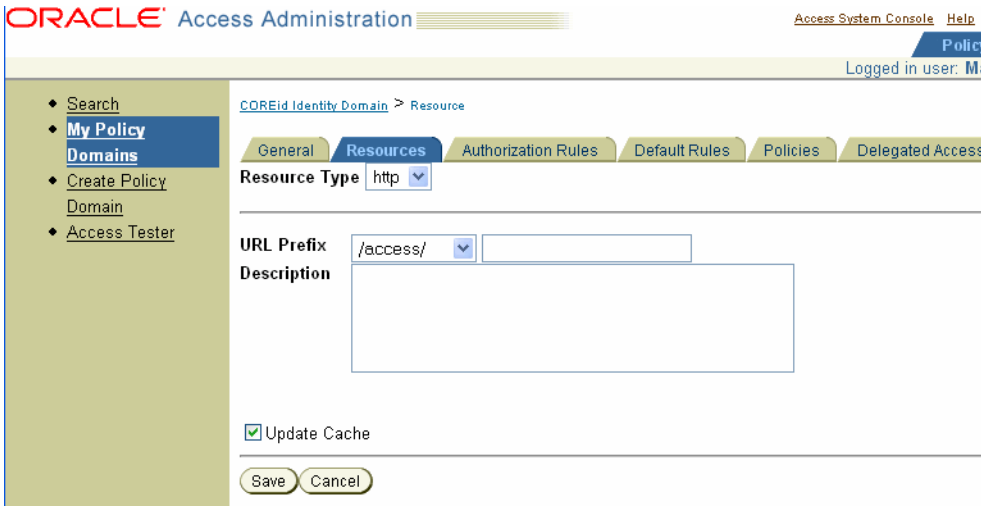
You use the Resources tab page to add resources to a policy domain after you create the domain.

To add resources to a policy domain

1. From the Policy Manager, click My Policy Domains in the left navigation pane, then click the policy domain link.
2. Click the Resources tab.

   If resources have been added to this domain, they are listed on this page, otherwise the message appears, "There Are No Resources Defined."
3. Click the Add button.

   The Resources page appears, as illustrated in the following.



4. In the Resource Type field, select an entry.

   The Access System provides two default resource types, HTTP and EJB. Others may be available if your administrator defined them through the System Console.

   Note: HTTP covers both HTTP and HTTPS resources.

   If host identifiers were created for individual servers, the Host Identifiers field appears. If no Host Identifiers were defined, this field does not appear on this page.

5. If a Host Identifiers field appears, select a Host Identifier for the resource.

   The Host Identifier enables the Access System to distinguish between otherwise identical URL prefixes for resources that might exist on multiple hosts.

6. Select an existing URL prefix to be the basis for the new URL.

   You can see the URL prefixes for existing resources only if you are a Master Access Administrator or if you are a Delegated Access Administrator with rights to view or manage the URL prefix.

7. You can optionally enter a more specific URL prefix string.

   Enter the URL prefix for the resource using an acceptable format. To add a specific resource, enter the remainder of the URL for that resource in the field, for example:

   - Directory (/marketing/.../)
   - Directory with wildcards (subfolder/*.html)
   - Specific file (marketing/subfolder/marketing.html)

8. In the next field, enter the name of a region to be appended to the URL prefix.

   For example, if the prefix you selected in the previous step was `/your_company`, you might enter `/sales` in this field.

   Note: You need to add the / in front of your entry unless you specified / as the policy root during setup.

You can later reuse the same prefix but add a different appended region, for example:

```
/your_company/marketing
```

After the newly defined region is saved, it appears in the URL Prefix field.

> **Note:** By default, The Access System reads URL prefixes and regions as case-insensitive. To change to case-sensitive, the Access Administrator should use the resource matching feature in the Common Information Configuration/Resource Type Definition function within the Access System Console. If you change this setting, you must restart the Access Servers and AccessGates.

9. In the Description field, enter a description of the protected region (whether a policy domain or a policy).

   Completing this field is optional.

10. Determine when you want Access Server caches to be updated:

    - **Immediately:** Select Update Cache to update all Access Server caches *immediately* with information about this new prefix.
    - **Later**: If you do not select Update Cache, the Access Server caches are updated when they time out and read new information from the directory server.

11. Click Save.

    The Resource page appears again and displays the name of the new resource.

12. Click OK to confirm your change.

13. Repeat these steps to add more resources to this policy domain.

Modifying a Resource's Description

Only the Master Access Administrator can modify a resource's description.

You can modify only the Description field of a resource. If you want to change the resource itself, you must delete it and create a new one.

To change a resource description

1. From the Policy Manager, select My Policy Domains.
2. In the My Policy Domains page, click a policy domain's link.

   The policy domain's General page appears.

3. Click the Resources tab.

   The Resource page appears with a list of resource types included in this policy domain.

4. Click a resource's link.

   The next page shows the type and prefix of the resource.

5. Click Modify.

   A new page appears.

6. Change the Description as needed.

7. Click Save.

Deleting a Resource

Only the Master Access Administrator can delete resources.

To delete a resource

1. From the Policy Manager, click My Policy Domains, and select a policy domain's link.

   The General page displays the Name, Description, and Enabled status of the domain.

2. Click Resources.

   The Resource page appears.

3. Select the check box for the resource you want to delete and click Delete.

   A message asks you to confirm your decision.

4. Select (or deselect) the Update Cache checkbox.

5. Click OK to delete the prefix (or click Cancel to exit the page without saving).

About the Master Audit Rule

The Access System enables you to capture and record user activities for protected resources, including user identity information and information about various authentication and authorization activities. You use auditing information to monitor activity for a specific policy domain.

The Access System provides a Master Audit Rule that can be configured by a Master Access Administrator. Delegated Access Administrators can use the Master Audit Rule to create their own audit rules for policy domains and policies. The Access System does not log any audit information to the audit log file until the Master Administrator or Master Access Administrator creates a Master Audit Rule.

The Master Audit Rule contains the following information:

- User identity attributes you want to audit (cn, uid, and so forth)
- Events to audit (authentication success, failure, and so forth)
- Selection of a date format
- Format and event mapping for audit log

Master Administrators and Master Access Administrators use the Access System Console to configure a Master Audit Rule, using the Add the Master Audit Rule page.

The rest of this section discusses the following topics:

- Configuring the Master Audit Rule
- Modifying the Master Audit Rule
- Deleting the Master Audit Rule

Note: Making most parameters unchangeable enforces common auditing parameters across all Access Servers.

Configuring the Master Audit Rule

The following procedure describes configuring the Master Audit rule for a server. A Master Access Administrator configures this rule.

To configure a server's Master Audit rule

1. From the Access System Console, click the Access System Configuration tab, then click Common Information Configuration in the left navigation pane.
2. Click the Master Audit Rule sub-tab.
3. Click the Add button on the No Master Audit Rule found page to create the master audit rule.

   The Add the Master Audit Rule page appears, as illustrated in the following.

4. In the Profile Attributes field, enter the identity profile attributes you want to capture.

   These attributes are written to the log file when the event happens. In most cases, cn is the best choice.

   Click the plus (+) and minus (–) icons to add or remove attribute fields.

   The Master Access Administrator can add attributes to this field, but cannot delete the ones you select.

5. In the Audit Events field, select the events you want to capture.

   Master Access Administrators and Delegated Access Administrators can add or delete events when configuring policy domains.

6. In the Audit Event Mapping field, enter the strings logged for each event.

   For example, Authentication Success maps to AUTHENT_SUCCESS.

7. In the Audit Date Type field, select the format in which dates are logged.

8. In the Audit Escape Character field, type a character that separates fields and ensures that logged information appears correctly in reports.

   If no escape character is specified, audit records will not be escaped.

9. In the Audit Record Format field, enter data types associated with authentication and authorization activities.

   Note: Supported data types for output to a file are shown in the following list. You may want to output to a database (using audit-2-db, for example). In this case, the format string for audit output must be replaced, as described in the auditing information in the *Oracle Access Manager Identity and Common Administration Guide*.

   - **ob_ip**: Corresponds to the IP address of the computer making the request.
   - **ob_datetime**: Corresponds to date and time. The date is logged in the format specified in the master audit policy. The time is logged as hh:mm:ss. The time is always the GMT time on the host that received the request, followed by the host's offset from GMT.
   - **ob_serverid**: Corresponds to the ID of the Access Server that is auditing this information.
   - **ob_url**: Request URL.
   - **ob_operation**: HTTP operation, such as GET, PUT, POST, or others.
   - **ob_event**: A string corresponding to the event that occurred. The event can be one of the following: Authentication Success, Authentication Failure, Authorization Success, or Authorization Failure.
   - **ob_userid**: Contains the user's distinguished name, if the user was successfully authenticated.

     If the user is authenticated and has an entry in the directory, in addition to the distinguished name, the log may contain other information that the authentication module of the Access Server is configured to audit. If the user does not exist in the directory, the only information that can be audited is the user name. If the user exists in the directory but enters an incorrect password, there is no way to confirm the user's identity. As a result, this information is not audited. Passwords are never written to the audit log for users who do not log in as Anonymous.
   - **ob_wgid**: ID of the AccessGate that received the request.
   - **ob_date**: Corresponds to date only. It does not include the time of the event unless the date format is ISO.
   - **ob_time**: Corresponds to the GMT time at which the event occurred on the host. Time is always logged as hh:mm:ss+/-<offset from GMT on host>.
   - **ob_time_no_offset**: Corresponds to the GMT time on the AccessGate, but no GMT offset is logged. Time is logged as hh:mm:ss. Master Access Administrators and Delegated Access Administrators cannot change these settings.

- **ob_reason**: Returns information for authentication success, authentication failure, authorization success, and authorization failure events. The overall reason is either ALLOW (for success) or DENY (for failure). However, in the case of DENY, any of the following reasons, which are given by the minor status code, can be the cause for denial of access. Also, a code indicating that there is no reason may be provided when the event is authentication success or authorization success.

  These reasons are returned to clarify the cause of denial, and they are represented by the following integers:

  - **40**: An invalid password was provided as input to the authentication process.
  - **68**: The overall result of evaluation of the authorization expression was inconclusive.
  - **2**: No reason is provided. This code is returned for authentication success and authorization success events.

10. Determine when you want Access Server caches to be updated.
    - **Immediately**: Select Update Cache to update all Access Server caches *immediately* with this auditing information.
    - **Later**: If you do not select Update Cache, the Access Server caches are updated when they time out and read the new auditing information from the directory server.
11. Click Save to implement your changes (or Cancel to leave this page without saving).

Modifying the Master Audit Rule

Master Administrators and Master Access Administrators use the Access System Console to modify a Master Audit Rule. You use the Modify the Master Audit Rule page to change the configuration of the Master Audit Rule.

To modify the Master Audit Rule

1. From the Access System Console, click the Access System Configuration tab, then click Common Information Configuration in the left navigation pane.
2. Click the Master Audit Rule tab.
3. Click the Modify button on the Master Audit Rule page.

   The Modify the Master Audit Rule page appears.
4. Change the parameters as necessary.
5. Click Save.

Deleting the Master Audit Rule

Master and Master Access Administrators can delete the existing Master Audit Rule from the Master Audit Rule page.

To delete the Master Audit Rule

1. From the Access System Console, select Access System Configuration, Common Information Configuration, Master Audit Rule

2. In the Master Audit Rule page, click Delete.

   You are prompted to confirm your decision.

3. Click OK to delete the rule (or Cancel to exit without saving).

Configuring Policies

Policies enable you to differentiate how subsets of resources in a domain are protected. You can use policies to establish more or less stringent protection for a subgroup of resources of a policy domain.

A policy can include:

- One or more resources.
- Allowed operations (request methods) for a resource type.
- URL patterns for a specific file, directory, query string pattern, or query string variable.

  See "About URL Prefixes" on page 4-16 for details.

- Authentication and auditing rules, and authorization rules and expressions different from the default ones.

If a resource is not covered by a policy, the default rules of the domain apply.

The following example of a policy domain includes two policies. Boggle Games, Inc. provides human resources information to three categories of personnel: regular employees, part-time employees, and contracted employees. The policy domain includes one URL: `/mycompany/HR`. Other details of the policies are:

- The company shares some information with all groups of users. All users know how and where to obtain a building access badge.

  Badge information resources reside in a subordinate badge directory, `/mycompany/HR/badges`.

  However, because resources in the badges directory are not protected with a policy, they fall under the protection of the policy domain's default rules.

- The company shares some information only with regular employees; regular employees can view information about holiday, vacation, and stock benefits.

  A policy is used to protect the resources for employee benefits, which reside in directories subordinate to `/mycompany/HR`.

- The company shares some information only with managers; managers can view lists of preferred vendors who provide contract personnel to Boggle Games, Inc.

The rest of this section discusses the following topics:

Policies with Overlapping Patterns

If you have multiple policies with overlapping patterns, the order of the policies within the policy domain becomes important. In this case, you should order the policies from the most granular to the least granular.

Adding a Policy

You use the Policies tab page to add a policy to resources of a policy domain. This section assumes that you have performed two prerequisite tasks:

- Created a policy domain, as explained in "To create a policy domain" on page 4-27
- Added a resource to the policy domain, as explained in "To add resources to a policy domain" on page 4-31

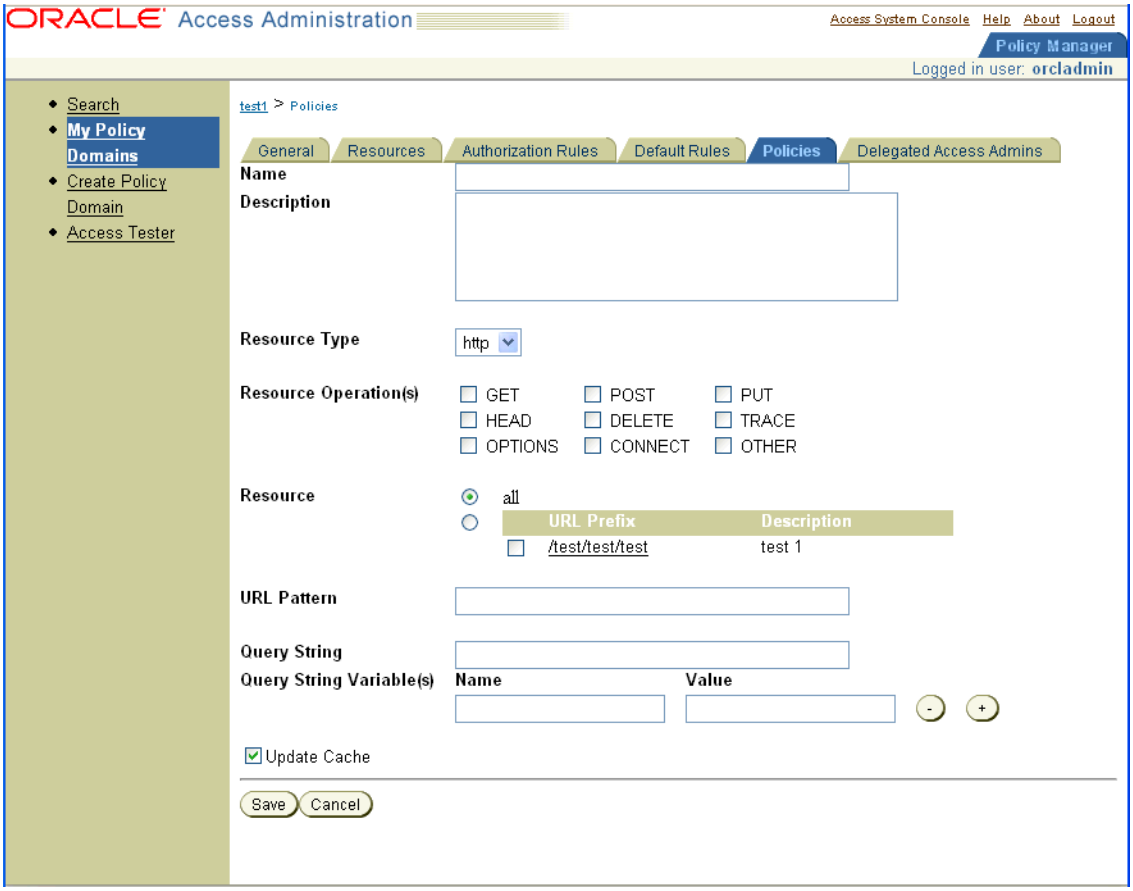
> **Note:** On some directory servers, adding a very large number of policies and resources may cause a size limit error. In lab conditions, this maximum has only been reached when multiple thousands of resources have been added to the policies.

Before adding URL patterns, see "Configuring URLs for Resources" on page 4-14.

To add a policy

1. From the Policy Manager, click My Policy Domains in the left navigation pane.
2. Click the policy domain that you want to add the policy to.
3. Select the Policies sub-tab and click Add.

   The policy configuration page appears.



4. Fill in information for the general information for the policy, and then click Save.
5. Click the name of the policy to display the sub-tabs for this policy.
6. Authentication Rule: Click this sub-tab and then click Add (or click View Default). Create (or edit) as authentication rule for this policy. The guidelines for this rule are the same as for the policy domain.
7. Authorization Expressions: Click this sub-tab and then click Add (or click View Default). Create (or edit) an authorization expression for this policy. The guidelines for this expression are the same as for the policy domain.
8. Audit Rule: Click this sub-tab. If there is no Master Audit Rule defined and you want to add an auditing rule to this policy, you are instructed to contact your Access System Administrator.

Modifying a Policy

You use the Policies page to modify a policy.

To modify a policy

1. From the Policy Manager, click My Policy Domains in the left navigation pane.
2. Click the link for the policy domain whose policy you want to modify.
3. Select the Policies tab, select the policy, and click Modify.
4. On the Policies tab modification page, change the policy information.

5. Click Save.

Setting the Order in which Policies Are Checked

If you create two or more policies, you can specify the order in which the Access Server checks them. By default, a new policy is checked last.

To set the order of policies within a domain

1. From the Policy Manager, select My Policy Domains, and click the link for the policy domain.
2. Select the Policies tab.
3. Click Order.
4. Select the name of the policy you want to move within the current order.

   Click the Up and Down arrows to relocate the policy.

   Repeat this process for each of the policies whose order you want to change.
5. Determine when you want Access Server caches to be updated.
   - **Immediately**: Select Update Cache to update all Access Server caches immediately with this auditing information.
   - **Later**: If you do not select Update Cache, the Access Server caches are updated when they time out and read the new auditing information from the directory server.
6. When you are satisfied with the order of the list of policies, click Save.

Deleting a Policy

You delete a policy directly from the list of policies for the policy domain it belongs to.

To delete a policy

1. From the Policy Manager tab, select My Policy Domains, click a link for a policy domain, then select the Policies tab.
2. Select the check box before the name of the policy you want to delete.
3. Click Delete.

Deploying a Policy into Production

After you have tested a policy domain that you administer, and you are satisfied that resource protection is enacted as planned, you can deploy the domain for production use. To deploy a policy domain, you enable it.

You must also enable a policy domain to test it. See "Using Access Tester" on page 4-44 for information describing how to test a policy domain.

Auditing User Activity for a Policy Domain

Auditing is the process of collecting information about users' activities in relation to the resources of a policy domain or its policies. The Access System automatically audits administrative events, such as clearing information from caches. Audit policies set in the Master Audit Rule and audit rules derived from it determine what is tracked.

You can configure audit policies for:

- Authentication and authorization success or failure
- Resource access
- Policy modification

You can customize audit output to include user profile attributes.You can use audit trails for reporting, history, or any purpose you see fit. For example, you can collect the cn and other attributes of user profiles to maintain detailed information about policy domain usage. This information can be searched and used to generate reports.

The rest of this section discusses the following topics:

Creating an Audit Rule for a Policy Domain

You can create audit rules for a policy domain. A policy domain's audit rule serves as the default rule for all resources of the domain unless you define an audit rule for any of the domain's policies.

You must derive this rule from a Master Audit Rule created by a Master Access Administrator. For details about creating a Master Audit Rule, see "About the Master Audit Rule" on page 4-34.

To create an audit rule for a policy domain

1. From the Access System Console, select Policy Manager, My Policy Domains, and click a link for a policy domain.

   The selected policy domain appears, with the General tab selected.

2. Click the Default Rules tab.

   The Default Rules tab, when selected, displays sub-tabs for Authentication Rule, Authorization Expression, and Audit Rule.

3. Click Audit Rule sub-tab.

   A page appears either showing the audit rule defined for the policy domain or reporting that there is no rule defined. If the page states that there is no Master Audit Rule defined, a Master Access Administrator must create one before you can define an audit rule for the policy domain.

4. Click Add to start the audit rule.

   The Audit Rule page appears. If the Master Audit Rule exists, its values are shown as defaults.

5. Select the events to be audited and the audit profile attributes.

6. Click Save.

Modifying an Audit Rule for a Policy Domain

For a policy domain, you can modify existing audit rules, which are derived from the Master Audit Rule.

To modify an audit rule for a policy domain

1. From the Access System Console, select Policy Manager, My Policy Domains, and click a link for a policy domain.

   The General page for the selected policy domain appears.

2. Select Default Rules, Audit Rule.

   The General page appears.

3. Click Default Rules, Audit Rule tab.

   The Audit Rule page appears.

4. Select the audit rule to be modified.

   A page with the rule's information appears.

5. Click Modify.

   The rule's page with editable text fields appears.

6. Modify the information and click Save.

Defining an Audit Rule for a Policy

If you define an audit rule for a policy, it overrides the default one defined for the policy domain. Before you can define a policy's audit rule, a Master Access Administrator must create a Master Audit Rule.

To define an audit rule for a policy

1. From the Access System Console, select Policy Manager, My Policy Domains, and click a link for a policy domain.

   The General page for the selected policy domain appears.

2. Click the Policies tab.

3. Select the policy for which you want to create an audit rule.

4. Click Audit Rule.

   A page appears either showing the audit rule defined for the policy domain or reporting that there is no rule defined. If the page states that there is no Master Audit Rule, a Master Access Administrator must create one before you can define an audit rule for the policy.

5. Click Add to start an audit rule.

   The Audit Rule page appears. Values for the Master Audit Rule, if one exists, are shown as defaults.

6. Select the events to be audited and the audit profile attributes.

7. Click Save.

Modifying an Audit Rule for a Policy

You can modify the audit rules for the policies of a policy domain. These rules are derived from the Master Audit Rule created by a Master Access Administrator.

To modify an audit rule for a policy

1. From the Access System Console, select Policy Manager, My Policy Domains, and click a link for a policy domain.

The General page for the selected policy domain appears.

2. Click the Policies tab.
3. Select the policy for which you want to create an audit rule.
4. Click Audit Rule.

   The Audit Rule page appears.
5. Select one of the audit rules.

   A page with the rule's information appears.
6. Click Modify.

   A page with the rule's information in editable text fields appears.
7. Modify the information, and click Save.

About the Audit Log File

An audit rule causes event-based data to be written to the audit log file. There is one audit log for each Access Server. You can configure the size of the audit log file and the rotation interval for each server. Depending on events recorded, the audit log may contain some duplicate audit entries.

Note: To audit to a database, by using audit-2-db for example, the format string used for audit output must be replaced, as documented in the *Oracle Access Manager Identity and Common Administration Guide*. Also, you need to have a supported database installed and specific configuration in the Access System.

Using Access Tester

Use the Access Tester to verify that the authentication and authorization rules and authorization expressions you created for a policy domain produce the results you expect. You should test the policy domain before you make it available for production. After you select various parameters for your rules and compare the results to what you expect, you may need to make adjustments to your rules.

Using the Access Tester when you have a custom authentication or authorization plug-ins configured requires that you first make certain files available. The following procedure describes using the Access Tester. You can skip Step 1 if you are not using a custom authentication or authorization plug-ins.

Note: You must enable the policy domain before you can test it. See "Enabling and Disabling Policy Domains" on page 4-29 for details.

To run Access Tester

1. Custom Authentication or Authorization Plug-ins: Copy the following items into the Web server /bin directory.

   For example if the Web server is Oracle HTTP Server, copy the following files to $ORACLE_HOME/bin.

   a. The custom plugin dll

b. The managed_plugin_interface.dll with which the custom plug-in was compiled

c. The managed_plugin_impl.dll from the following directory path:

PolicyManager_install_dir/webcomponent/access/oblix/apps/common/bin

2. From the Policy Manager, click the Access Tester link in the left navigation pane.
3. In the URL field of the Access Tester page, type the full path to the application or content you want to check.

   The path (with the addition of an http:// or https:// prefix) should bring up a landing page when entered in a browser.
4. In the Resource Type field, select an entry from the list.

   You can only select a type that has been defined in the Access System Console.
5. In the Resource Operation field, select the request methods you want to test for this URL.

   Note: The operations available in this field depend on the resource type you selected in the previous step.

   If you select none of these, they *all* are tested.
6. If you want to know if a particular computer can access the resource (URL), type the computer's IP address into the From this IP Address field.

   You must enter a complete IP address. Wildcards are not allowed in this field.
7. In the Date/Time of access list, do one of the following:
   - Click the button beside "any" to test this resource without timing restrictions.
   - Click the button beside "specific date and time" and fill in the following Date and Time fields.
8. In the Check access for the following user(s) field, do one of the following:
   - Click the button beside "all users."

     If you select all users, the Access Tester processes the authentication and authorization information for all users. If there are a great many user entries in your database, this could take a considerable amount of time.
   - Click the button beside "selected users," then click the Select User button to display the Sector page where you can select specific users.

     Note: Do *not* select groups. Access Tester can only test access control for individual users, not groups. Also, it will not resolve groups to the individual level.
9. In the Show Administrators field, select the number of end users you want to display at one time.
10. From the list, select the button beside the option that describes the appropriate user access:
    - show only users who are allowed
    - show only users who are

- show both

Policy and Rule options are also listed:

- show matching Policy—If this resource (URL) is protected by a policy, and you want it to display the policy, select the button beside show matching Policy.
- show matching Rule—If you want the authorization rule for this resource (URL) to be displayed, select the button beside show matching Rule.

11. Click Submit.

    The results appear.

Delegating Policy Domain Administration

When a Master Access Administrators creates a policy domain, he or she assumes the role of default Delegated Access Administrator. This default Delegated Access Administrator has all management rights within that domain, and can delegate administration of that domain to others who then become Delegated Access Administrators.

There are three levels of rights for Delegated Access Administrators:

- **Delegate**: Delegates grant or basic rights to other users.
- **Grant**: Delegates basic rights to other users.
- **Basic**: Performs delegated tasks, but cannot delegate this right to others.

Only Delegated Access Administrators who have rights to a specific domain—or the Master Access Administrator—can view a policy domain.

Delegated Access Administrators can manage policy domains that are delegated to them. They can also *create* policy domains for resources that fall under the URL prefixes of the policy domains that are delegated to them.

Table 4–4 summarizes the rights of the different types of administrator:

Table 4–4 Types of administrators and their rights

Type of Administrator	Policy Domain Rights
Master Administrator	■ Creates Master Access Administrators. ■ Creates the policy root. ■ Creates the policy base.
Master Access Administrator	■ Creates the first policy domain and adds resources to it. ■ Defines resource types. ■ Creates, deletes, and manages authentication and authorization schemes. ■ Creates the Master Audit Rule. ■ Delegates management of policy domains to Delegated Access Administrators. ■ Retains all rights delegated to other users.

Table 4–4 (Cont.) Types of administrators and their rights

Type of Administrator	Policy Domain Rights
Delegated Access Administrator with delegate rights	For that policy domain only, a Delegated Access Administrator with delegate rights can: ■ View the domain. ■ Create authentication and authorization rules. ■ Create an authorization expression for the policy domain and for any policies it contains. ■ Create audit rules based on the Master Audit Rule. ■ Define Delegated Access Administrators with grant or basic rights. ■ Enable or disable the policy domain. ■ Test the policy domain. **Important**: The Delegated Access Administrator cannot redefine the attributes of the Master Audit Rule.
Delegated Access Administrator with grant rights	Created by a Master Access Administrator or a Delegated Access Administrator with delegate rights. For that policy domain only, a Delegated Access Administrator with grant rights can: ■ View the domain. ■ Create and delete authorization rules. ■ Create or modify an authorization expression for the policy domain and for any policies it contains. ■ Create audit rules based on the Master Audit Rule, or change the events to be audited, removing existing events or including other ones. ■ Create and delete policies for resources in the policy domain. ■ Define Delegated Access Administrators with grant or basic rights. ■ Enable or disable the policy domain. ■ Test the policy domain.
Delegated Access Administrator with basic rights	Created by a Master Access Administrator or a Delegated Access Administrator with delegate or grant rights. A Delegated Access Administrator with basic rights cannot create or delete policy domains. For the specified policy domain, this administrator can: ■ View the domain. ■ Create or delete authentication and authorization rules. ■ Create or modify an authorization expression for the policy domain or any of its policies. ■ Create audit rules based on the Master Audit Rule. ■ Redefine the events to be audited, removing existing events or including other events. ■ Add new attributes to the Master Audit Rule. However, this administrator cannot redefine existing attributes. ■ Create and delete policies for resources. ■ Enable or disable the domain. ■ Using Access Tester, verify access to the resources protected by the policy domain.

Configuring Policy Domain Administrators

Both Master Access Administrators and Delegated Access Administrators can administer policy domains. For details about creating Master Access Administrators, see "Configuring Access Administrators" on page 2-1. To create and view Delegated Access Administrators for a policy domain and to modify delegated rights, see the following paragraphs.

To view Delegated Access Administrators for a policy domain

1. From the Policy Manager, select My Policy Domains and click the policy domain.
2. Select the Delegated Access Admins tab.
3. On the Delegated Access Admins page, in the Show Administrators with field, select the Delegate Rights, Grant Rights, or Basic Rights radio button.

   The page is refreshed to display the current users and groups with the selected administrative right for this policy domain. If no users have this right, you receive the message "There are no Delegated Access Admins with this right."
4. Click the administrator link to display the profile for the user or group.

To delegate rights for a policy domain

1. From the Policy Manager, select My Policy Domains and click the policy domain.
2. Select Delegated Access Admins.
3. Click the radio button for the kind of right that you want to grant.
4. Click the Modify button at the bottom of the Delegated Access Admin page.
5. Click Select User.
6. Use the Search process to display a list of users to select from, and click Done.
7. Click Save.

To modify policy domain rights

1. From the Policy Manager, select My Policy Domains.
2. Click the policy domain.
3. Select the Delegated Access Admins tab.
4. Click Modify.
5. Modify the field values for the rights you want to change.
6. Click Save.

5

Configuring User Authentication

Authentication is the process of proving that a user is who he or she claims to be. The Access System enables you to configure authentication rules in the policy domains and policies that protect your resources. Authentication rules, in turn, contain authentication schemes. Authentication schemes provide the methods for performing verification of the user's identity.

This chapter explains authentication schemes, authentication rules, and actions that you can associate with the possible outcomes for authentication rules.

This chapter discusses the following topics:

> **Note:** In addition to authentication schemes, policy domains and policies contain authorization rules, authorization expressions, and audit rules. See "Configuring User Authorization" on page 6-1 for details.
>
> After you define a policy domain, you can define its authentication rules, authorization rules and expressions, and audit rules in any order.

About Authentication and Authentication Schemes

You use the Access System to define authentication schemes and authentication rules. These schemes and rules establish ways to authenticate users who try to access resources in your policy domains. To authenticate a user, you obtain and process information about the user to verify that the user is who he or she claims to be.

The rest of this section discusses the following topics:

- Background Reading
- Basic Components of Authentication
- About Authentication Schemes
- Default Authentication Schemes

> **Note:** One advanced authentication method is to collect context-specific information before submitting the request to the Access Server. Context-specific information can be in the form of an external call for information. This topic is discussed in the chapter on form-based authentication. See "Using an External Call for Data in an Authentication Request" on page A-6 for details.

Background Reading

Before you read this chapter, read the following:

- Chapter 3, "Configuring WebGates and Access Servers" on page 3-1.

  This chapter describes how to configure AccessGates and Access Servers, which you must do before the policy domains you create can take effect.

- Chapter 4, "Protecting Resources with Policy Domains" on page 4-1.

  This chapter describes policy domains, policies, resources, and the Master Audit Rule. You must configure policy domains and resources along with authentication schemes.

- Chapter 6, "Configuring User Authorization" on page 6-1.

  This chapter describes creating authorization schemes, rules, and expressions. You should have at least a general idea of the type of authorization methods you want before configuring authentication schemes.

Basic Components of Authentication

To configure authentication, you create the following components:

- **Authentication Schemes**: An authentication scheme includes the method used to challenge the user for credentials.

  An authentication scheme also includes one or more steps. Each step contains one or more plug-ins that perform part of the authentication process.

- **Authentication Plug-Ins**: Plug-ins usually implement methods for challenging the user for credentials.

  The Access System provides default challenge plug-ins. You can use these plug-ins alone, you can replace them with custom ones, or you can use them in combination with custom ones.

  The Access System also provides a credential mapping plug-in that matches the credentials of the user who requests a resource to a user profile in an LDAP directory.

- **Authentication Rules**: After configuring an authentication scheme, you can include it in an authentication rule.

You provide a default authentication rule for each policy domain. You can also create an authentication rule for each policy within a policy domain.

To use the Access System for user authentication, your directory must be structured in one of the following ways:

- All of your user information is in one branch of a single directory.
- Your user information is in multiple directory servers but all of the directories use the same schema.
- You divide your user profile information logically across different branches of your directory, each with its own search base.

Searching a Single Directory: You can use authentication to search a single search base of a single directory. If the information is not found in that directory, the user cannot be authenticated and the search terminates.

Searching Two Directories of the Same Type: You can use chained authentication to search two or more directories of the same type managed by the same Oracle Access Manager instance.

- **Searching Two Directories Consecutively**: You can use two directories of the same type to store information about users. You may want each directory to be searched until information about a user is found. If the information is found in the first directory, you can end the search process. If the information is not found in the first directory, you can search the next directory or you can end the search, depending on the user's status.
- **Searching One Directory or Another Based on Conditions**: You may want to create another chained authentication scheme to search one or another directory. For example, the scheme can specify that one directory is to be searched if the user is an employee and that another directory is to be searched if the user is a vendor. For each condition, if the user is not found in the first directory, the scheme specifies that a third directory is to be searched before the authentication process is terminated.

Searching Different Branches of the Same Directory: You can use chained authentication to search different branches of the same directory for a user profile. You can store some user profiles in one branch, some in another, and some in yet another. For example, if the third branch of the directory contains legacy data, you can search the third branch for a user profile only if the information for the user cannot be found in the other two branches. For this purpose, you can configure an authentication scheme whose steps contain plug-ins to begin from the first search base, map the user's credentials to a user profile, and, if it is found, process the credentials, then terminate the search. If the user profile is not found in the first branch, the scheme's steps can direct the search to the next search base, and so on.

About Authentication Schemes

An authentication scheme specifies how authentication is to be performed for users who request access to a resource that is protected using a policy domain or a policy. The authentication scheme is part of an authentication rule. A simple authentication scheme contains a single step. For chained authentication, an authentication scheme contains multiple steps linked together to produce different behaviors depending on certain conditions.

Only Master Access Administrators can create authentication schemes. See "Delegating Policy Domain Administration" on page 4-46 for details.

Authentication schemes consist of the following main components:

- **General information**: General information for an authentication scheme includes data such as the method to use to challenge users for credentials when authenticating his or her identity, and the security level for the scheme.

  See "Defining and Managing Authentication Schemes" on page 5-7 for details.

- **Plug-Ins**: You add one or more plug-ins to an authentication scheme to process the user's credentials.

  Only the plug-ins you add to a scheme can be used for any of its steps. See "Plug-Ins for Authentication" on page 5-18 for details.

- **Steps**: An authentication scheme can include one or more steps, each of which must include at least one plug-in.

  A step contains a group of plug-ins that are run in order of their position in the step. See "About Authentication Steps" on page 5-34 for details.

- **Authentication flows**: These are the possible execution paths through the steps of an authentication scheme.

  See "Configuring Authentication Flows" on page 5-43 for details.

Default Authentication Schemes

The following authentication schemes are configured during Policy Manager setup:

- **Oracle Access and Identity Basic Over LDAP**: Used to protect Oracle Access Manager-related resources (URLs) for most directory types.
- **Oracle Access and Identity Basic Over LDAP for AD Forest**: This is only set up if you installed the Oracle Access Manager system in an Active Directory configuration.

  This scheme can be used to protect Oracle Access Manager-related resources (URLs) for Active Directory.

- **Anonymous**: Used to unprotect specific Oracle Access Manager URLs.

  The Anonymous authentication scheme allows users to access Oracle Access Manager-specific URLs that you do not want to protect with the Access System, for example, Web pages for self registration and lost password management. This scheme uses a challenge method of None, which means that users are not challenged and do not need to enter their credentials.

During setup you may also have configured two authentication schemes based on configuration information in your user directory:

- **Basic Over LDAP**: This built-in Web server challenge mechanism requires the user to enter their login ID and password.

  The credentials supplied are compared to the user's profile in the LDAP directory server.

- **Client Certificate**: This authentication scheme is a certificate-based user identification method.

  To use this method, a user certificate must be installed on your browser and the Web server must be SSL-enabled.

  See Also:

  - "Certificate Decode Plug-In" on page 5-25
  - "To view the certificate details" on page 5-27

About Challenge Methods

You must include a challenge method in every authentication scheme you define.You can use a challenge method provided with the Access System or provide a custom one.

> **Note:** If you are using the Access System-provided schemes, you must ensure the obMappingFilter of the plug-in parameter is set correctly for your directory and environment. For details, see Table 5–2.

Challenge Methods and Schemes Configured During Installation

If the Master Administrator selected a challenge method during installation of the Access System, the Access System configures authentication schemes automatically. See the *Oracle Access Manager Installation Guide* for information about configuration of these schemes during the installation process.

The Access System supports the following five challenge methods:

- **None**: Users are not prompted to provide any credential information. This method allows access to Identity System-specific resources (URLs) that you do not want to protect with the Access System, for example, Self Registration. This method is used by the Anonymous authentication scheme that can be configured during Policy Manager setup. No challenge parameters are used.
- **Basic**: Users must enter a user name and password in a window supplied by the Web server.

  A challenge parameter is required with this method, as described in "About Challenge Parameters" on page 5-6. This method can be redirected to SSL. See "About Basic and X.509Cert (Client Cert)" on page 5-5 for details.
- **Client Cert (X509Cert)**: The user's browser must supply an X.509 digital certificate over SSL to the Access Server through WebGate to complete authentication.

  A challenge parameter is not used. The user's organization can determine how to obtain a certificate. See "About Basic and X.509Cert (Client Cert)" on page 5-5 for details.
- **Form:** This method is similar to the basic challenge method, but users enter information in a custom HTML form.

  You can choose the information users must provide in the form that you create. A challenge parameter is used, as described in "About Challenge Parameters" on page 5-6. For details about form-based authentication for redirecting users to another site, see "Form-Based Authentication" on page A-1.
- **Ext**: An external challenge method (outside Oracle Access Manager) is used.

  This method enables you to use your own authentication challenge method.

  If you use Ext, you must provide the challenge parameter `creds`. This parameter is a space-separated list of server variables set by the external challenge method. See the *Oracle Access Manager Customization Guide* for more information.

About Basic and X.509Cert (Client Cert)

Oracle Access Manager supports client certificate authentication using public key encryption cryptography and X.509 certificates. The client certificate challenge method uses the Secure Sockets Layer version 3 (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.

For both Basic and X.509Cert, you can configure an AccessGate to handle unauthenticated requests received over a non-SSL connection.

For the client certificate challenge method, you configure the AccessGate to redirect the user's browser to another server to establish an SSL connection, as mentioned previously. After the AccessGate authenticates the certificate, it redirects the user's browser back to the original URL.

Note: Basic authentication fails with non-ASCII login credentials. Non-ASCII user credentials are supported in only form-based (Form) authentication, as described in "Form-Based Authentication" on page A-1.

About Challenge Parameters

A challenge parameter is a short text string that helps the end user remember which username and password should be entered.

Here is an example of the text string for an LDAP directory:

```
realm:LDAP username + password
```

The Basic, Form, and Ext challenge methods require a challenge parameter. Challenge parameters are not used for None or X509Cert challenge methods.

See Also:

- "About the sensitivecreds Challenge Parameter" on page 5-6
- "About Plug-Ins for Challenge Methods" on page 5-7
- "Defining a New Authentication Scheme" on page 5-9

About the sensitivecreds Challenge Parameter

A new challenge parameter for Basic and Form challenge methods is provided to suppress sensitive information when WebGate encounters an authentication error and displays an error page. The new challenge parameter is:

```
sensitivecreds
```

The authentication scheme must contain the challenge parameter `sensitivecreds` followed by a space-separated list of credential values that represent the names of fields to be suppressed on the error page. For example:

```
sensitivecreds:password ssn
sensitivecreds:password creditcardnumber secret-code
sensitivecreds:password creditcardnumber secret\\ code
```

The space character is used only as a delimeter between credential values; the space character cannot be used alone within a credential value. To define a space within a credential value (secret code, for example), you must use the escape character `\\` as part of the first element in the value (`secret\\`, for example). The result is that the following element is interpretted as part of the first element and not as a separate field. Therefore, `secret\\ code` becomes `secret code` on the page. Without the escape character `\\`, a space within a value (`code`) is considered to be a delimeter between two separate fields.

Following is a description of the example values:

- In the first example, `password` is the name of the first field to be sanitized and `ssn` is the name of a second field to be sanitized. There are no spaces within the credential value.
- In the second example, `password` is the name of the first field to be sanitized, `creditcardnumber` is the name of a second field to be sanitized, and secret-code is the name of the third field to be sanitized.
- In the third example, `password` is the name of the first field to be sanitized, `creditcardnumber` is the name of a second field to be sanitized, and secret`\\` code is the name of the third field to be sanitized. Here secret`\\` code represents a value that includes a space: `secret code`.

Note: When using `sensitivecreds`, any plug-in that accepts the password to be suppressed from the error page should be included in the authentication scheme.

When using sensitivecreds, there are no special requirements or constraints for the following, which can be defined as usual:

- Challenge Redirect
- Steps
- AuthN Flow

See Also:

- "About Plug-Ins for Challenge Methods" on page 5-7
- "Defining a New Authentication Scheme" on page 5-9

About Plug-Ins for Challenge Methods

Plug-ins are the engines of an authentication scheme. They implement challenge methods, map user credentials to user profile entries in a directory, process user credentials, perform custom tasks related to the authentication process, and so on. Authentication schemes contain the following two types of plug-ins:

- Access System-Provided Plug-Ins
- Custom plug-ins

After defining basic information about the authentication scheme, you can add plug-ins for challenge methods. See "Access System Plug-Ins for Authentication Challenge Methods" on page 5-21 for details.

Defining and Managing Authentication Schemes

Every authentication scheme must contain a challenge method and a way to map the credentials provided by the user to the corresponding user profile stored in the directory. Creating an authentication scheme includes defining how the scheme challenges the user for credentials, maps the information, verifies it, and so forth. For example, a scheme's challenge method may require users to provide passwords or it may require users to provide certificates attesting to their identity.

An authentication scheme can also contain plug-ins that do additional processing. For example, a plug-in can search multiple directories based on conditions and perform

tasks based on the outcome of other processes. After you create an authentication scheme, you can add plug-ins to the scheme and then configure the scheme's steps and their execution order.

Task overview: Defining and managing authentication schemes

1. View existing authentication schemes to see if the one you want to create is already defined.

   See "Listing and Viewing Authentication Schemes" on page 5-8.

2. If necessary, create a new authentication scheme.

   See "Defining a New Authentication Scheme" on page 5-9.

3. Enable or disable the scheme.

   See "Enabling and Disabling Authentication Schemes" on page 5-15.

4. Modify the scheme as needed.

   See "Modifying an Authentication Scheme" on page 5-13.

5. Remove obsolete authentication schemes.

   See "Deleting a Authentication Scheme" on page 5-18.

6. Set up a scheme when using multiple searchbases (also known as disjoint domains or realms).

   See "Configuring an Authentication Scheme When Using Multiple Searchbases" on page 5-14.

The rest of this section discusses the following topics:

Listing and Viewing Authentication Schemes

You can list existing authentication schemes to ensure that the one you want to create is not already defined. You can also view the details of a scheme.

Note: After you modify a pre-existing scheme, it cannot be used for systems prior to version 6.5.

To list and view the details of an authentication scheme

1. From the landing page for the Access System, click the Access System Console link.

If you are working with the Policy Manager, click the Access System Console link at the top of the page.

2. Click the Access System Configuration tab.
3. Click the Authentication Management link in the left navigation pane.

   The Authentication Management: List All Authentication Schemes page appears.
4. Click the name of the authentication scheme you want to see.

   The Details for an Authentication Scheme page appears.

Defining a New Authentication Scheme

Before you define an authentication scheme, determine the following:

- A unique name for the scheme and a brief description of what it does.

  Delegated Access Administrators who create authentication rules that contain the scheme select the scheme from a list. Provide a description of each scheme to simplify their task.
- The security level of the authentication scheme.

  The security level of the scheme reflects the challenge method and degree of security used to protect transport of credentials from the user. The security level is expressed as an integer.

  After a user is authenticated for a resource at a specified level, the user is automatically authenticated for other resources in the same policy domain or in different policy domains, if the resources have the same or a lower security level as the original resource. See "Changing the Security Level of an Authentication Scheme" on page 5-20 and "Configuring Single Sign-On" on page 7-1 for details.
- The type of challenge and the parameters to use to obtain the user's credentials

  The challenge method specifies how authentication is to be performed and the information required to authenticate the user. Each authentication scheme can have only one challenge method. Authentication is successful if the user credentials obtained in response to a challenge match one and only one DN in the directory.

  Usually a challenge parameter provides WebGate with additional information to perform an authentication, often used to prompt the user for information. Challenge parameters are entered in *name*:*value* format.
- Whether users must be authenticated using a server enabled for Secure Sockets Layer (SSL).
- The URL of a server specified as the Challenge Redirect, if you want user requests to be redirected to another server for processing.

  Authentication schemes may require redirection of the request to another URL to properly carry out the authentication. For example, redirection is used when an authentication request for a resource is made over HTTP but the authentication scheme requires the authentication to be made over HTTPS (secure HTTP). WebGate sends the redirect to the user's browser telling it to request a URL defined by the authentication scheme. After authentication is completed, WebGate redirects the browser back to the original requested resource.

  Also, redirection is required to perform multi-domain single sign-on (SSO). For information describing how challenge redirects are used for multi-domain single sign-on, see "Multi-Domain Single Sign-On" on page 7-9.

- Whether the scheme should be enabled.

  This page includes a radio button that you can set to enable or disable an authentication scheme. For details about enabling and disabling a scheme, see "Enabling and Disabling Authentication Schemes" on page 5-15.

- Whether the Access Server's cache should be updated automatically with new information and changes you make to the scheme

  This page includes a checkbox that you can select to specify that the cache should be updated.

  See Also: "Configuring an Authentication Scheme When Using Multiple Searchbases" on page 5-14

To create an authentication scheme and specify general information

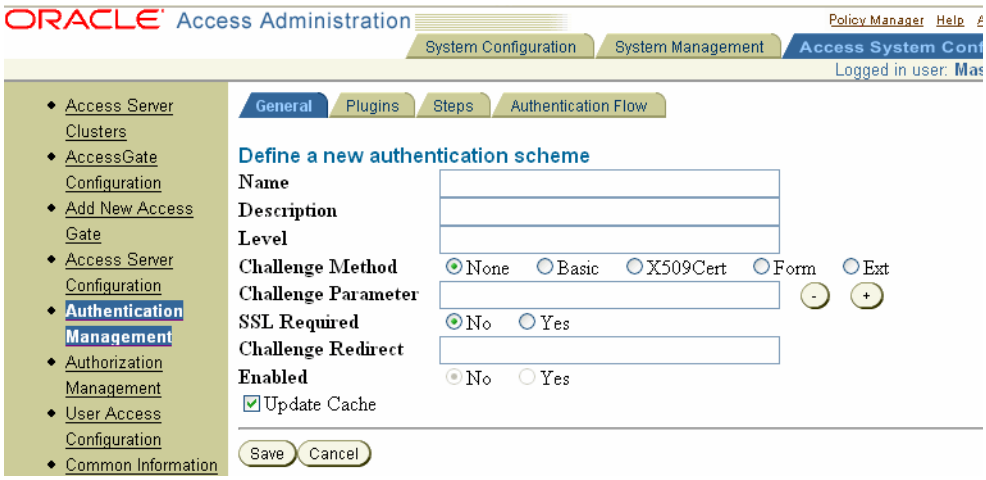
1. From the landing page from the Access System, click the Access System Console link.

   If you are working with the Policy Manager, click the Access System Console link at the top of the page.

2. From the Access System Console, click the Access System Configuration tab, then click the Authentication Management link in the left navigation pane.

3. On the List All Authentication Schemes page, click Add.

   The Define a new authentication scheme page appears with the General tab selected, as illustrated in the following screen shot.



4. In the Name field, specify a name for the authentication scheme.

   Each authentication scheme must have a unique name.

5. In the Description field, provide a brief description of the scheme.

   For instance, you might explain the purpose of the scheme and its behavior.

6. In the Level field, enter an integer corresponding to the level of security of the scheme.

7. In the Challenge Method field, click the radio button for the authentication scheme challenge method you want to use (each authentication scheme can have only one challenge method see "About Challenge Methods" on page 5-5:

- None
- Basic
- X.509Cert
- Form
- Ext

8. If you selected Form, Basic, or Ext for the challenge method, specify a Challenge Parameter.

   - For the Basic Challenge Method, type a short text string to be used as a hint to help end users remember their user names and passwords for the requested resource.

     Here is an example of the text string for an LDAP directory:

     ```
     realm:LDAP username + password
     ```

     See Also: "About the sensitivecreds Challenge Parameter" on page 5-6 if you are using the Basic challenge method and want to suppress sensitive information from the error page that appears when WebGate encounters an authentication error

   - If you selected the Form challenge method, you are required to provide the following parameters in the Challenge Parameter fields.

     See Also: "About the sensitivecreds Challenge Parameter" on page 5-6 if you are using the Form challenge method

Challenge Parameter	Description
form:	Indicates where the HTML form is located relative to the host's document directory. `For example, form:/login.html.`
creds:	Lists all fields used for login in the HTML form. The parameter creds is a space-separated list. For example: `creds:`*`username password`* Where *username* and *password* are the names of the relevant fields in the login form. **Note:** You can specify the creds parameter for the other challenge method types. WebGate uses the named credentials from the creds parameter to find matching parameters in the user's request. For Basic schemes, WebGate always uses user ID and password. For Cert schemes, WebGate always uses certificate. These do not need to be declared in the creds parameter.
action:	The URL that the HTML form is posting to.

Challenge Parameter	Description
passthrough:	The passthrough: parameter is optional. It determines whether the WebGate redirects the browser back to the originally requested resource or passes the login credentials on to another program. The Access System assumes that the URL given for the form in the authentication scheme is on the same computer as WebGate. Possible values are yes or no. Accept the default value, no, if you want WebGate to redirect the browser back to the originally requested resource. Specify yes if you want to pass the login credentials through to a post-processing program. **Note:** This parameter does not work on a Domino Web server when using the POST method for form-based login.
maxpostdata bytes:	The value of this optional parameter is the maximum number of data bytes that an end user can post for authentication to a Web server that uses a WebGate. If the POST data exceeds the threshold set by maxpostdatabytes in the form-based authentication scheme, user receives an error and a log entry is added at the DEBUG3 log level in the oblog.log file. Example: maxpostdatabytes:100 If you omit this parameter, the end user can post unlimited length strings for authentication to a Web server that is protected by a WebGate. Very long strings can cause the WebGate or Web server to crash, denial of service, or another fatal error.

9. Determine whether you want the end user authenticated through an SSL-enabled server.

   If you click Yes, the request is routed to the HTTPS server you specify in the Challenge Redirect field.

10. In the Challenge Redirect field, enter the URL of another server to which you want to redirect this request if authentication does not take place on the resource Web server.

    Use the host URL of the designated primary authentication server. For example:

    `https://www.`*`yourcompany`*`.com`

    The host name that you enter in the Challenge Redirect field must be available in the Host Identifiers list when adding or modifying an authentication scheme. See "Adding a Host Identifier" on page 3-48 for details.

11. Select the radio button to enable or disable the authentication scheme.

12. Click Save (or Cancel):

    - If you click Save, the Details for an Authentication Scheme display page appears. This page displays the information you entered for the new authentication scheme.
    - If you click Cancel, the configuration is not saved and the page listing all authentication schemes is displayed again.

13. Proceed with the following tasks, as needed, to complete your authentications scheme definition:

    - Adding a Plug-In to an Authentication Scheme

- Adding a Step to an Authentication Scheme
- Configuring and Modifying the Flows of an Authentication Chain
- Verifying and Correcting Cycles in an Authentication Flow
- Enabling and Disabling Authentication Schemes
- Securing the ObSSOCookie in an Authentication Scheme
- Retaining the ObSSOCookie Over Multiple Sessions
- Configuring Browser Cookies as Credentials in an Authentication Scheme

Modifying an Authentication Scheme

You can modify the content of an existing authentication scheme. To modify a new authentication scheme as you define it, select the tab and modify the information on its pages.

In pre-10.4.1 versions of Oracle Access Manager, before you modified a scheme, you needed to ensure that it was not included in the authentication rules of any active policy domains, and you needed to disable the scheme. These actions are now unnecessary.

Authentication schemes must be enabled to be available for use in a rule. If a disabled scheme is used in action domains or policies, the resource is not protected.

The following procedure describes how to modify an existing scheme.

Note: Existing authentication schemes are compatible with prior releases of the Access System. However, if you modify an older authentication scheme, it will run on version 6.5 and later Access Servers but not on earlier versions of the Access Server.

To modify the contents of an authentication scheme

1. Ensure that the scheme is not included in the authentication rules of any active policy domains.

   See "Enabling and Disabling Authentication Schemes" on page 5-15 for details.

2. Go to the landing page for the Access System and click the Access System Console link.

   If you are working with the Policy Manager, click the link for the Access System Console at the top of the page.

3. From the Access System Console, click the Access System Configuration tab.

4. Click the Authentication Management link in the left navigation pane.

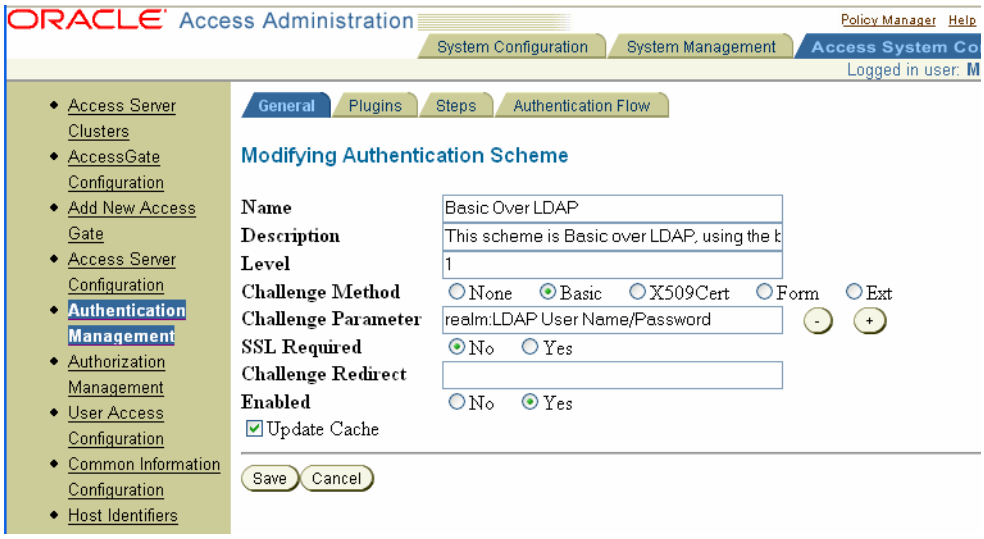
   The Authentication Management: List All Authentication Schemes page appears.

5. Click the name of the authentication scheme you want to modify.

   The Details for Authentication Scheme page appears. From this page, you can select other tabs, such as Plugins, Steps, Authentication Flow.

6. Click Modify.

   The Modifying Authentication Scheme page appears, as illustrated in the following screen shot. You can modify the scheme's general information from this page.



If you are modifying the Oracle Access and Identity Basic Over LDAP authentication scheme, you must enter ASCII characters in the Challenge Parameter field.

7. To modify other parts of the scheme
   - Select the tab for that part.
   - Click Modify on the page which appears.
   - Follow the configuration process for that page.
8. Click Save.
9. If needed, enable the scheme.

   Authentication schemes must be enabled to be available for use in a rule. If a disabled scheme is used in action domains or policies, the resource is not protected.

Configuring an Authentication Scheme When Using Multiple Searchbases

If you have multiple searchbases in your directory, also called disjoint domains or multiple realms, depending on your directory type, you need to configure an authentication scheme that enables searches for users with identical user IDs who reside in the separate searchbases (the disjoint domains).

For additional information on multiple directory searchbases (or disjoint domains, or realms), see the *Oracle Access Manager Identity and Common Administration Guide*.

To configure an authentication scheme for multiple searchbases (also known as disjoint domains or realms)

1. On Active Directory, add the plug-in for Oracle Access and Identity Basic Over LDAP for AD Forest to your authentication scheme.

   See "Adding a Plug-In to an Authentication Scheme" on page 5-30 for details.

   For other platforms, create a custom authentication scheme similar to the following:

   ```
   obMappingBase="%domain%",obMappingFilter="(&(& (objectclass=objectclassname)(
   loginattribute=%userid%))(|(!
   ```

```
(obuseraccountcontrol=*))(obuseraccountcontrol=ACTIVATED)))",obdomain="domain"
```

Where:

- *objectclassname* is the name of the Person object class, for example gensiteorgperson
- *loginattribute* is the name of the login attribute for the Person object class, for example, genuserid

The %domain% and %userid% elements extract the user domain and user ID.

For example:

```
credential_mapping
obMappingBase="%domain%",obMappingFilter="(&(&
(objectclass=gensiteorgperson)(genuserid=%userid%))(|(!
(obuseraccountcontrol=*))(obuseraccountcontrol=ACTIVATED)))",obdomain="domain"
```

2. Modify this plug-in:
   - Change the object class to your user object class.
   - Change the genuserid to your login attribute configured on your user object class.
3. In the authentication action that you define upon successful authentication using this scheme, you need to set the following values:

   Type: HEADERVAR

   > **Note:** This must be done for both the default identity and access policy domains.

   Name: HTTP_OBLIX_UID

   Return Attribute: obuniqueid

   See "Setting Authentication Actions" on page 5-58 for details.

   > **Note:** This must be done for both the default identity and access policy domains.

4. In addition, you need to make the following configuration file changes:
   - Locate the globalparams.xml file in the following path:

     PolicyManager_install_dir/access/oblix/apps/common/bin/globalparams.xml
   - Change the value of whichAttrIsLogin to ObUniqueID.
5. Make the same change in the following Identity Server file:

   IdentityServer_install_dir/identity/oblix/apps/common/bin/globalparams.xml

Enabling and Disabling Authentication Schemes

When you create an authentication scheme, the scheme is disabled until you enable it. It is good practice to enable an authentication scheme only after you complete its configuration.

Before you disable a scheme, determine if the scheme is used in authentication rules of any active policy domains. If a scheme is disabled:

- It is not available for use in authentication rules.
- Resources previously protected by the scheme are no longer available to users requesting access to them.

The following error message is reported when an attempt is made to access resources protected by an authentication rule containing a disabled authentication scheme:

```
The authentication scheme SchemeID is invalid or has been disabled
```

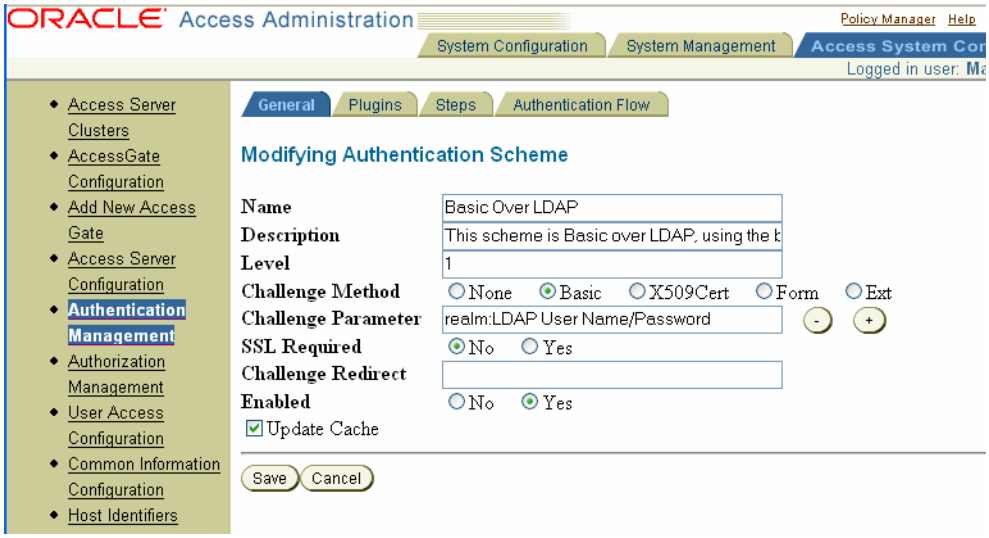
To enable or disable an authentication scheme

1. From the Access System Console, click the Access System Configuration tab.
2. Click the Authentication Management link in the left navigation pane.

   The Authentication Management: List All Authentication Schemes page appears.
3. In the List All Authentication Schemes page, click the scheme you want to enable or disable.

   The Details for Authentication Scheme page appears.
4. Click Modify.

   The Modifying Authentication Scheme page appears, as follows:



5. Select the appropriate radio button to enable or disable the authentication scheme.
6. Click Save.

Securing the ObSSOCookie in an Authentication Scheme

The Access System implements single sign-on through an encrypted cookie called the ObSSOCookie. See "Configuring Single Sign-On" on page 7-1 for details.

You can specify a challenge parameter that ensures the ObSSOCookie is only sent over an SSL connection and prevents the cookie from being sent back to a non-secure Web server.

To secure the ObSSOCookie

1. Create an authentication scheme.

   See "Defining and Managing Authentication Schemes" on page 5-7 for details.

2. In the Challenge Parameter field, add another field and specify the following to secure the ObSSOCookie:

   ```
   ssoCookie:httponly
   ```

   Note: The challenge parameter is case-sensitive. Be sure to enter an uppercase C in ssoCookie, and uppercase S in Secure.

3. In the SSL Required field, click Yes to ensure the end user is authenticated through an SSL-enabled server.

Retaining the ObSSOCookie Over Multiple Sessions

You can configure an authentication scheme that allows the user to log in for a time period rather than a single session by adding the challenge parameter `ssoCookie:max-age` in the authentication scheme. This creates a persistent cookie in some browsers, rather than one that lasts for a single session. The persistent cookie functionality works with Internet Explorer and Mozilla browsers.

To define a persistent cookie in the authentication scheme

1. Define an authentication scheme, as described in "Defining a New Authentication Scheme" on page 5-9.

2. In the challenge parameter for this scheme, add the following:

   `ssoCookie:max-age=`*`time-in seconds`*

   Where *time-in-seconds* represents the time interval when the cookie expires. For example, a setting of `ssoCookie:max-age=2592000` sets the cookie to expire in 30 days (2592000 seconds).

Configuring Browser Cookies as Credentials in an Authentication Scheme

A cookie is a text string that an application can set in an HTTP request or response. A cookie helps a Web site retrieve information about the user, for example, previously visited pages. You can include a cookie name in the creds challenge parameter of an authentication scheme. If a user attempts to access a resource that is protected by this type of scheme and is required to log in, the WebGate retrieves the cookie value if it exists, and sends it to the Access Server with the other credentials. An authentication plug-in can obtain the cookie value using the same methods it uses to extract other credentials.

To include a browser cookie as a credential in an authentication scheme

1. From the landing page from the Access System, click the Access System Console link.

   If you are working with the Policy Manager, click the Access System Console link at the top of the page.

2. From the Access System Console, click the Access System Configuration tab, then click the Authentication Management link in the left navigation pane.

3. To create an authentication scheme, on the List All Authentication Schemes page, click Add.

   You can also modify an existing authentication scheme, for example, Basic Over LDAP.

4. In the authentication scheme, include the name of the cookie as the challenge parameter, for example:

   ```
   creds:MyDomain
   ```

5. Use the cookie credential value in an Oracle-provided or custom plug-in in the authentication scheme.

   For example, a credential_mapping plug-in can compare the cookie credential value to a user attribute using an ObMappingFilter parameter similar to the following:

   ```
   obMappingBase="o=Company,c=US",obMappingFilter="(&(objectclass=
   gensiteorgperson)(uid=%userid%)(domain=%MyDomain%))"
   ```

   The preceding example compares the value of the MyDomain cookie to a user attribute named domain. The authentication only succeeds if the user supplies a cookie with the expected value.

Deleting a Authentication Scheme

An authentication scheme cannot be deleted if it is being used by a policy. Remove it from the policy before deleting it.

To delete an authentication scheme

1. Launch the Access System, select Access System Console, then Access System Configuration.
2. Click the Authentication Management link in the left navigation pane.

   The Authentication Management: List All Authentication Schemes page appears.
3. Select the check box for the authentication scheme to delete.

   To delete multiple schemes, select the check box for each scheme.
4. Click Delete.

Plug-Ins for Authentication

An authentication plug-in is an executable shared library that participates in the user authentication process. Plug-ins are the engines of an authentication scheme. They implement challenge methods, map user credentials to user profile entries in a directory, process user credentials, perform custom tasks related to the authentication process, and so on.

The steps of an authentication scheme include one or more plug-ins. Before you can add plug-ins to a step, you must add them to the authentication scheme. You must add to the authentication scheme all of the plug-ins to be used for any of its steps.

The rest of this section discusses the following topics:

- About Access System-Provided Plug-Ins
- About Custom Plug-Ins

About Access System-Provided Plug-Ins

The Access System provides plug-ins to implement the challenge methods it supports by default, plus a credential mapping plug-in. Every authentication scheme must include a credential mapping plug-in to match user credentials with a user profile in the directory. You can use the Access System-provided plug-in for this purpose, or you can replace it with a custom one that implements the same behavior. See "Access System Plug-Ins for Authentication Challenge Methods" on page 5-21 for details about these plug-ins and their parameters.

You include plug-ins in a step. If execution of a plug-in provided by the Access System fails, the step that contains the plug-in fails. For details about steps and plug-ins, see "About Authentication Steps" on page 5-34.

About Custom Plug-Ins

You can create custom plug-ins to serve purposes related to your authentication process. For example, you can create custom plug-ins to be used to search each of several directories, or each of several branches of a directory. Or, if you store user profiles for one department in one branch of a directory and user profiles for another department in another branch of the same directory, you may want to search the branches consecutively depending on certain conditions.

The Access System supports writing authentication plug-ins in C and any language supported by the Microsoft .NET framework, including C, C++, and Visual Basic.

If execution of a custom plug-in fails, the outcome depends on the step to be executed next as determined by the authentication flow of the authentication scheme and the return code returned by the plug-in.

For information describing how to create plug-ins to be used for authentication, see the chapter on the authentication plug-in API in the *Oracle Access Manager Developer Guide*.

For information about authentication flows, see "Configuring Authentication Flows" on page 5-43.

Note on Managed Code and Plug-Ins

For details about managed code for authentication plug-ins, see the *Oracle Access Manager Developer Guide*. A managed code authentication plug-in must include the following two plug-in parameters:

- `obtype="managed"`
- `obnamespace="`*`namespace`*`"`

  Here, *namespace* is the namespace of the managed code.

Return Codes for Plug-Ins

If you create a custom plug-in, the Access Server expects your custom plug-in to return one of the following four status codes:

- ObAnPluginStatusContinue
- ObAnPluginStatusAllowed
- ObAnPluginStatusDenied
- ObAnPluginStatusAbort

For details explaining what these return codes means and how the Access Server interprets them and responds to them, see the chapter on the authentication plug-in API in the *Oracle Access Manager Developer Guide.*

About Reusing Plug-Ins Across Authentication Schemes

You can use custom plug-ins in any number of authentication schemes. You must rename the plug-in so that its name is unique across authentication schemes. When you add a plug-in to an authentication scheme, the Access Server transparently assigns that plug-in an identifier. The Access System manages these numbers internally. You cannot change them or delete them.

Because the Access System uses identifiers to keep track of plug-ins, the execution order of plug-ins is not dependent on their position exclusively, and a single plug-in can be reused in the following ways:

- It can be used in combination with other plug-ins to form a step.
- It can be used more than once within a step.

  A step can contain multiple instances of the plug-in with different parameters.
- It can be used for different steps of the same authentication scheme.

Changing the Security Level of an Authentication Scheme

You can write a custom plug-in to change the security level of an authentication scheme. In some cases, you may want to increase the security level of an authentication scheme depending on certain conditions. You may want the security level of an authentication scheme to depend on the application the user logged in from. For example, if Active Directory and a reverse proxy are among the sources your users can log in from, you may want to set one authentication security level to be used for users who log in from Active Directory and another security level to be used for users who log in from the reverse proxy.

Your code could determine the source from which the user logged in, and it could set the authentication scheme security level accordingly. It could check the current value of the ObAuthentSchemeLevel variable maintained by the Access Server in the credential list for the scheme. Your plug-in could change the security level, setting the variable value to a security level that depends on the requirements you have established for login from the application. To set the security level, you modify the value of ObAuthentSchemeLevel variable. If you do not change this value, the Access

Server uses the security level already set for the authentication scheme through the user interface.

You can use the following code in your plug-in to open the credentials list file, check the ObAuthentSchemeLevel variable value, and set it to the security level you want to use for an application.

```
schemeLevel = pFnBlock->GetCredFn(pInfo->Creds,
"ObAuthentSchemeLevel");

if (schemeLevel ! = NULL) {
schemeLevelAsInt = atoi(schemeLevel);
schemeLevelAsInt +=10
iota(schemeLevelAsInt, buff, 10);

pFnBlock->SetCredFn(pInfo->Creds,
"ObAuthentSchemeLevel", buff);
```

Access System Plug-Ins for Authentication Challenge Methods

Table 5–1 shows the predefined challenge methods and the plug-ins that support them. For challenge methods with multiple plug-ins, the order in which the plug-ins are executed is identified.

You can use these plug-ins in order, as shown in Table 5–1, in one or more steps of your authentication scheme. You can use them with custom plug-ins that replace the Oracle-provided plug-ins. Or, you can provide all of your own custom plug-ins to implement the required ones for the authentication schemes.

Table 5–1 Access System Predefined Challenge Methods and Plug-Ins

Challenge Method	Plug-Ins and Order of Execution
None	credential_mapping
Basic	1. credential_mapping 2. validate_password
Client Cert (X509Cert)	1. cert_decode 2. credential_mapping The following plug-in is optional: 3. selection_filter
Form	1. credential_mapping 2. validate_password

The Access System provides the following plug-ins.

- **credential_mapping**: This plug-in maps the user ID to a valid distinguished name (DN) in the directory.

  You can configure the attribute to which the user ID is mapped. The most common attribute it is mapped to is uid. You can also map the user ID to a profile attribute other than uid by changing the obMappingFilter parameter.

  A credential mapping plug-in is required for every authentication scheme. You can use the credential_mapping plug-in provided by the Access System for an LDAP directory server for this purpose, or you can provide your own plug-in. For more information, see "Credential Mapping Plug-In" on page 5-22.

- **validate_password**: This plug-in is used to validate the user's password against the LDAP data source. It addresses the Form and Basic challenge methods. See "Validate Password Plug-In" on page 5-24 for details.
- **selection_filter**: This plug-in further validates the authentication credentials with some criteria. It addresses credentials provided by the user and does not use backend data sources. It addresses all of the challenge methods.
- **cert_decode**: The plug-in validates the certificate and does not use a data source. It addresses the Client Certificate (Cert) challenge method. See "Certificate Decode Plug-In" on page 5-25 for details.
- **NT/Win2000**: This plug-in addresses Form and Basic challenge methods for Microsoft Windows 2000 systems. See "Plug-Ins for Windows and SecurID" on page 5-28 for details.
- **SecurID**: 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. This addresses the Form challenge method for SecurID. For details, see the *Oracle Access Manager Integration Guide*.

For each of the Access System-provided plug-ins described in this section, a table is provided which includes information about the plug-in, its parameters, and how it is used.

The following explanations apply to these tables:

1. Parameters for all plug-ins are case-sensitive. You must enter them exactly as they are shown in the tables.
2. Parameters not labeled as mandatory in the tables are optional.

Note: Oracle recommends that all authentication schemes use the credential_mapping plug-in even if you select None as the challenge method. However, this is not a requirement. See "Credential Mapping Plug-In" on page 5-22 for required parameters.

Credential Mapping Plug-In

Your authentication scheme must provide the functionality implemented by the credential_mapping plug-in. It must map the user's credentials to information in the LDAP directory server.

If you do not use the Access System-provided credential_mapping plug-in, you must create a custom plug-in that performs the same task. Table 5–2 gives the parameters you use for the credential mapping plug-in. Both the obMappingBase and obMappingFilter parameter values are used to map the extracted user credentials to a Distinguished Name (DN) in the directory. Additional details follow this table.

Table 5–2 Credential Mapping Parameters

Characteristic	Description	Details
Plugin Name	credential_mapping	Maps user-provided information to a valid DN in the directory. If one DN (not zero and not more than one) matches the specified criteria, authentication continues. The obMappingBase and obMappingFilter parameters are added to the list of credentials and the internal uid is set to the DN. The plug-in fails if zero or more than one DN is returned.
Plugin Parameters	obMappingBase	Base DN in the LDAP search.
	obMappingFilter	Filter in the LDAP search: ■ This parameter is mandatory. ■ The value specified for this parameter is used to filter for categories of end users.
	obDomain	Used only to authenticate a user against an Active Directory Forest when the challenge method is Basic.
	obEnableCredentialCache	Turns the credential mapping cache on or off in the credential_mapping plug-in. By default, the credential mapping cache is turned off. Oracle recommends that you accept the default for the credential mapping cache.

There are two parameters that are critical to credential mapping, which you must support if you provide your own plug-ins. Both are required:

- obMappingBase: The search base in the directory where the search for user credentials begins.

  For example, the default Anonymous authentication scheme uses the following search base in its plug-in:

  ```
  obMappingBase="o=company,c=us"
  ```

- obMappingFilter: The search criteria for the filter.

  For example, the default Basic Over LDAP authentication scheme uses the following search criteria in its plug-in:

  ```
  obMappingFilter="(&(objectclass=inetorgperson)(uid=%userid%))
  "
  ```

  Where the %userid% is extracted from the user's credentials, for example, a field in the login form for form-based authentication.

The credential_mapping plugin in the Client Certificate authentication scheme in Oracle Access Manager should match the user certificate installed in the Web browser. For example, if the user certificate has cn and mail fields, the following are valid credential mapping parameters for a Client Certificate authentication scheme:

CN

```
obMappingBase="dc=us,dc=oracle,dc=com",obMappingFilter="(&(objectclass=inetorgpers
```

```
on)(cn=%certSubject.CN%))"
```

Mail

```
obMappingBase="dc=us,dc=oracle,dc=com",obMappingFilter="(&(objectclass=inetorgpers
on)(mail=%certSubject.E%))"
```

Filtering Inactive Users

You can add the obuseraccountcontrol parameter to the obMappingFilter parameter used for the credential mapping plug-in. This makes it possible to filter two categories of users:

Users who have been added to your directory server, but who have not been activated in the Identity System.

Users who have been deactivated from the Identity System, but who are still in your directory server.

Here is an example of an obuseraccountcontrol term to filter out the two categories of users:

```
(|(!(obuseraccountcontrol=*))
(obuseraccountcontrol=ACTIVATED))
```

If obuseraccountcontrol is ACTIVATED, or there is no value, then inactive users are filtered out. The obuseraccountcontrol parameter must be used with the obMappingFilter parameter. It cannot be specified without obMappingFilter.

Validate Password Plug-In

The validate_password plug-in validates the user's password against the specified directory server for the authentication scheme. For validate_password, the Access Server uses the same directory server against which it performed the credential_mapping plug-in with a successful outcome.

Here is an example of settings for the validate_password plug-in:

```
validate_password
obCredentialPassword="password", obAnonUser="cn=anonymous, o=Company, c=US"
```

Table 5–3 describes the validate password plug-in.

Table 5–3 Validate Password Plug-In

Name	validate_password
Purpose	Validates the user-provided password against the user's password in the directory.
Result	If the user-entered password matches the password in that user's directory entry, authentication continues. If not, the plug-in fails.

Table 5–4 describes the parameters for the Validate Password plug-in.

Table 5–4 Validate Password Plug-ins

Parameter	Description
obCredential Password	Specifies the name of the password field. This parameter is mandatory, and it must be listed first.

Table 5–4 (Cont.) Validate Password Plug-ins

Parameter	Description
obAnonUser	Specifies a DN that is authenticated with any password. This DN must map to a user profile, preferably with restricted access. There may be multiple obAnonUser parameters for a single plug-in. Examples: guest, anonymous.
obCredValidationByAs	When set to true, the Access Server validates passwords using its cache. A user's initial attempt is validated by the directory server. The Access Server caches an MD5 hash of the password and checks the password when subsequent requests are made. If the given and cached password match, the password is considered valid.
obPwdHashTTL	This setting controls the interval during which the Access Server validates passwords by comparing them with a cached password. After the interval, the Access Server returns to the directory server to validate each password. The default value is 1800 seconds (30 minutes).
obReadPasswdMode and obWritePasswdMode	Values can be LDAP or CACHE: ■ If the value is LDAP, the user's entry is obtained from the directory server for each authentication. ■ If the value is CACHE, the first authentication is made from the directory server, and afterward from the cache. Values for both parameters (obReadPasswdMode and obWritePasswdMode) should be the same. That is, both parameters should be either LDAP or CACHE. Enables password management for the authentication scheme in the Access Server.
obWebPassURLprefix	This parameter works with Lost Password Management, which is described in the section on configuring password policies in the *Oracle Access Manager Identity and Common Administration Guide*. When a WebGate and a WebPass are installed on different servers, and a user accesses a resource that is protected by the WebGate, this parameter is required to point to the challenge response page. The value of this plug-in is as follows: http://*webpasshost*:*webpassport*

Certificate Decode Plug-In

The certificate decode plug-in extracts the components of the certificate subject's and issuer's Distinguished Name (DN). For each component, the plug-in inserts a credential with a certSubject or certIssuer prefix. For instance, if your certificates have a subject name such as givenName=somename, the plug-in adds the credential certSubject.givenName=somename to the credential list.

Table 5–5 describes the certificate decode plug-in.

Table 5–5 Certificate Decode Plug-in

Name	cert_decode
Purpose	Decodes the certificate and extracts the elements of the certificate's subject and issuer DN. This plug-in can be used with the X.509Cert challenge method.
Result	If the decoding is successful, the elements of the certificate's subject and issuer DN are added to the list of credentials. If not, authentication fails.
Parameters	None

If your certificate is stored in the browser, you can view the certificate details. For details, see "To view the certificate details" on page 5-27.

The following table lists the OIDs of the attributes that are supported by the Access Server with the corresponding suffix used to retrieve the attribute.

Table 5–6 OIDs Supported by the Access Server

OID	Component lookup name
2.4.5.3	CN
2.5.4.4	SN
2.5.4.5	Serial Number
2.5.4.6	C
2.5.4.7	L
2.5.4.8	ST
2.5.4.9	Street Address
2.5.4.10	O
2.5.4.11	OU
2.5.4.12	Title
2.5.4.13	Description
2.5.4.14	Search Guide
2.5.4.15	Business Category
2.5.4.16	Postal Address
2.5.4.17	Postal Code
2.5.4.18	Post OfficeBox
2.5.4.19	Physical Delivery Office Name
2.5.4.20	Telephone Number
2.5.4.21	Telex Number
2.5.4.22	Telex Terminal Identifier
2.5.4.23	Facsimile Telephone Number
2.5.4.24	x121 Address
2.5.4.25	International ISDN Number
2.5.4.26	Registered Address

Table 5–6 (Cont.) OIDs Supported by the Access Server

OID	Component lookup name
2.5.4.27	Destination Indicator
2.5.4.28	Preferred Delivery Method
2.5.4.29	Presentation Address
2.5.4.30	Supported Application Context
2.5.4.31	Member
2.5.4.32	Owner
2.5.4.33	Role Occupant
2.5.4.34	See Also
2.5.4.35	User Password
2.5.4.36	User Certificate
2.5.4.37	CA Certificate
2.5.4.38	Authority Revocation List
2.5.4.39	Certificate Revocation List
2.5.4.40	Cross Certificate Pair
2.5.4.41	Name
2.5.4.42	Given Name
2.5.4.43	Initials
2.5.4.44	Generation Qualifier
2.5.4.45	Unique Identifier
2.5.4.46	DN Qualifier
2.5.4.47	Enhanced Search Code
2.5.4.48	Protocol Information
2.5.4.49	Distinguished Name
2.5.4.50	Unique Member
2.5.4.51	House Identifier
2.5.4.52	Supported Algorithms
2.5.4.53	Delta Revocation List
2.5.4.58	Certificate Attribute
2.5.4.65	Pseudonym
1.2.840.113549.1.9.1	E
0.9.2342.19200300.100.1.1	UID

Notice that most of the names are space separated. The following is an excerpt of code used to retrieve these values from an authentication plug-in:

```
sn = pFnBlock->GetCredFn(pInfo->Creds, "cerSubject.Serial Number");
```

To view the certificate details

1. Open up an IE browser.

2. In Tools menu click Internet Options.
3. Click the Content Tab.
4. Click the Certificates button.
5. Double-click your certificate.
6. Click the Details tab.
7. Click the Subject line.

Caching Validated Passwords to Increase Performance

By default, the directory server validates user passwords. To increase performance, you can use the Access Server to validate passwords after the first time they are validated by the directory server.

For this purpose, you must:

- Include the validate_password plug-in in an authentication scheme
- Set the plug-in's obCredValidationByAS parameter to true

When the obCredValidationByAS parameter is set to true, the Access Server caches an MD5 hash of a user's password after it is validated by the directory server.

The next time the user attempts to access a resource within the same policy domain, the user's password is compared with the cached password. If the two match, the given password is validated and the user is granted access to the requested resource.

Another parameter, obPwdHashTTL, controls the length of time the Access Server validates passwords. The default is 1800 seconds (30 minutes). You can change this value. When the specified length of time elapses, the password validation function returns to the directory server.

Here is an example of settings for these parameters that allow the Access Server to validate passwords for 100 seconds:

```
validate_password obCredentialPassword="password",, obCredValidationByAS="true",
obAnonUser="cn=anonymous, o=Company, c=US", obPwdHashTTL="100"
```

Plug-Ins for Windows and SecurID

The Access System offers plug-ins that allow you to authenticate users whose information is stored on Windows or in the SecureID security product. These plug-ins are installed automatically when you install the Access System. This section describes the plug-ins for Windows

For information about the SecureID plug-in, see the *Oracle Access Manager Developer Guide.*

Table 5–7 describes the Windows plug-in used to authenticate against a Windows domain.

Table 5–7 Windows Plug-in

Name	authn_windows
Purpose	Authenticates user name and password against a Windows domain.
Result	■ If authn_windows returns success, authentication continues. ■ If not, authentication fails.

Table 5–7 (Cont.) Windows Plug-in

Name	authn_windows
Parameters	■ ntusername—Name of the field containing the user name. This parameter is mandatory. ■ ntpwd—Name of the field containing the password. This parameter is mandatory. ■ ntdomain—Name of the field containing the domain.

Adding and Managing Plug-Ins

The steps of an authentication scheme include one or more plug-ins. Before you can add plug-ins to a step, you must add to the authentication scheme all of the plug-ins to be used for any of its steps.

For information about defining plug-ins, see "Plug-Ins for Authentication" on page 5-18.

The first time you add plug-ins to an authentication scheme, the Access System creates a default step that includes all of them. If you are using an authentication scheme from a release prior to the version 6.5 Access System, the Access Server creates a default step for the authentication scheme containing all of its plug-ins.

The rest of this section discusses the following topics:

- Viewing Plug-Ins for an Authentication Scheme
- Adding a Plug-In to an Authentication Scheme
- Deleting Plug-Ins from an Authentication Scheme

Viewing Plug-Ins for an Authentication Scheme

You can list an authentication scheme's plug-ins at any time. For example, you may want to list the plug-ins to see ones already added to that scheme before you add others. The plug-ins list displays the names and parameters of the plug-ins already added to the authentication scheme. The list may include any Access System-provided and custom plug-ins previously added to the scheme.

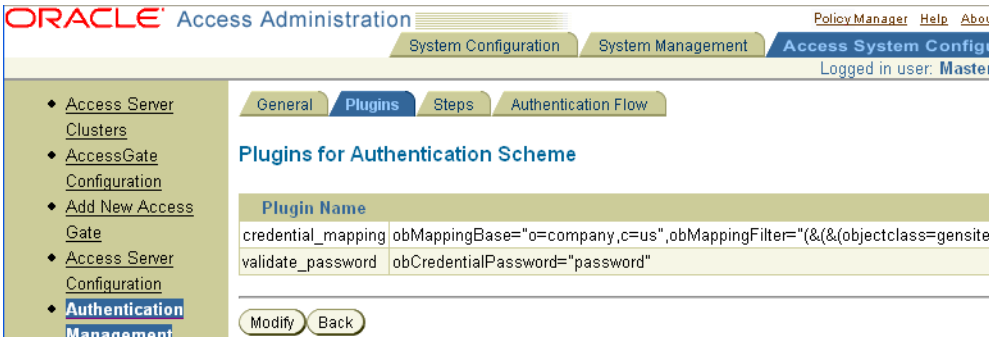
Note: It is possible to have multiple credential_mapping plug-ins within a scheme.

To view the list of plug-ins for an authentication scheme

1. From the Access System Console, click the Access System Configuration tab.
2. Click the Authentication Management link in the left navigation pane.

   The List All Authentication Schemes page appears.
3. In the List All Authentication Schemes page, click the scheme for which you want to display a list of plug-ins.
4. Select the Plugins tab.

   The plug-ins for an Authentication Scheme page appears, as illustrated in the following screen.



Adding a Plug-In to an Authentication Scheme

When you add a plug-in to an authentication scheme, you specify the name of the plug-in and its parameters. You can add the same plug-in more than once to an authentication scheme if each instance of the plug-in has different parameters. Each instance of a plug-in with unique parameters appears as a separate plug-in in the list.

Use the following task to add a plug-in to an authentication scheme, whether you are adding it to a new scheme or an existing one.

The authentication schemes that use the plug-ins must be enabled to be available for use in a rule. If a disabled scheme is used in action domains or policies, the resource is not protected.

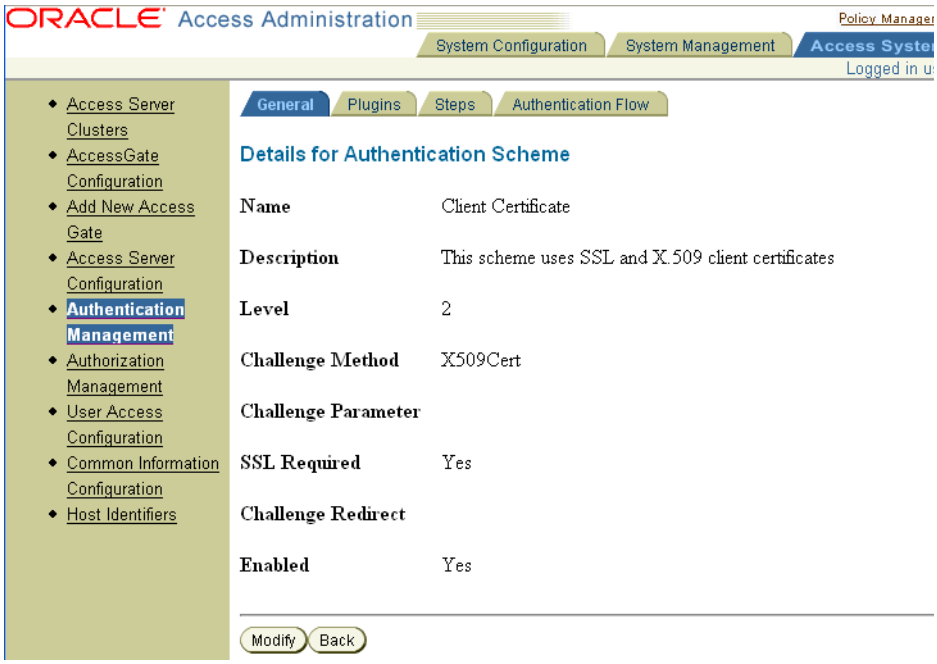
Note: When adding the credential mapping plug-in to an authentication scheme, ensure that the credential_mapping plug-in is placed before the validate_password plug-in. The authentication scheme must process the plug-in that validates the attribute for the login ID before validating the password.

To add plug-ins to an authentication scheme

1. From the Access System Console, click the Access System Configuration tab.
2. Click the Authentication Management link in the left navigation pane.

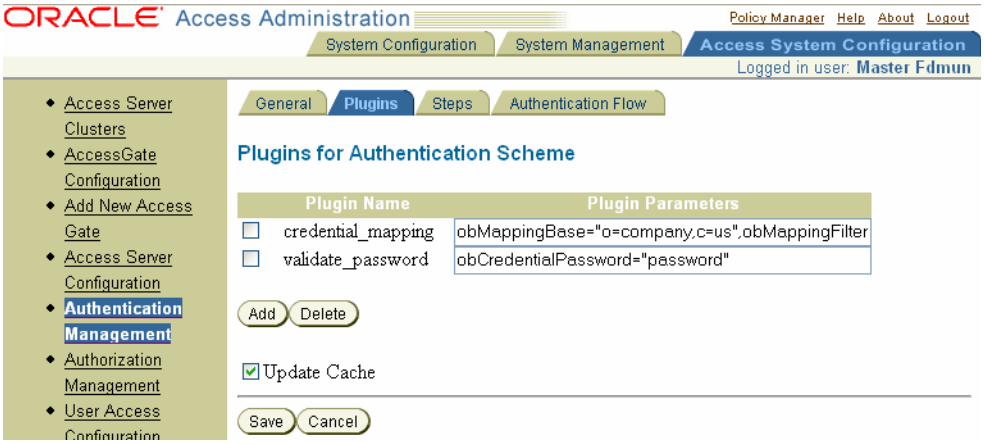
   The Authentication Management: List All Authentication Schemes page appears.
3. Click the link for an authentication scheme.

   The Details for Authentication Scheme page appears, as shown in the following screen shot.



4. Click Modify to display the Modifying Authentication Scheme page.
5. Select the Plugins tab to display the plug-ins for this authentication scheme.
6. Click Modify

   The Plugins for Authentication Scheme page changes to include the Add and Delete buttons as well as the Update Cache checkbox, as illustrated in the following screen shot.



7. Click Add.

   The page changes to include a list of options and a text box for selecting and defining the plug-in to be added. You either select a Access System-provided plug-in or enter the name of the custom plug-in in the Plugin Name box.

   A credential mapping plug-in is required for every authentication scheme. The credential mapping plug-in must precede a validate password plug-in, if you include one, because the user's login ID must be identified before his or her

password can be validated. You can select the Access System-provided plug-in or a custom one that implements the same behavior.

Select a plug-in from the right-most Plugin Name text box or enter the name of a custom plug-in in the text box. For details on this plug-in, see "Credential Mapping Plug-In" on page 5-22.

Each parameter can have multiple values.

To add more plug-ins, click the Add button after you finish adding the previous one. Repeat this step for each plug-in that you want to add.

8. Click Save.

Deleting Plug-Ins from an Authentication Scheme

You can remove a plug-in from an authentication scheme, but you must first remove the plug-in from any steps of the scheme that include it.

To delete plug-ins from an authentication scheme

1. From the Access System Console, click the Access System Configuration tab.
2. Click the Authentication Management link in the left navigation pane.

   The Authentication Management: List All Authentication Schemes page appears.
3. Select the name of the authentication scheme whose plug-in you want to delete.

   The Define an Authentication Scheme page appears.
4. Click Modify.
5. Select the Plugins tab to display the plug-ins for this authentication scheme.
6. Click Modify.

   The Plugins for Authentication Scheme page appears.
7. Select the plug-in that you want to delete by checking the box before the name of the plug-in. To delete multiple plug-ins, select each of them.
8. Click the Delete button at the bottom of the list to delete the selected plug-ins from the authentication scheme.
9. Click Save.

About Chained Authentication

When a user requests access to a resource protected by an authentication rule, the rule's authentication scheme determines the way in which authentication is to be performed. For chained authentication schemes, this process includes obtaining user credentials and mapping those credentials to a user profile. A chained authentication scheme can be designed to do this and more. For example, instead of limiting the search for a user profile to one directory, a chained authentication scheme can support attempts to map the credentials to a user profile in one directory, another directory, or yet another directory consecutively, until the information is found. It can also include additional processes indirectly related to the authentication process.

Process overview: A simple chained authentication scheme

Step 1: Plug-ins for Directory A map user credentials to one directory server and verify those credentials. If either plug-in of Directory A fails, the step specifies that Step 2 is to be executed.

Step 2: Plug-ins for Directory B map user credentials to another directory server and verify the credentials.
Step 3: If either plug-in of Directory B fails, this step specifies that the plug-ins for Issue Message and Quit are to be executed.

The rest of this section discusses the following topics:

About Creating an Authentication Rule Using Chained Authentication

Here is an overview of the process you use to set up chained authentication for a policy domain. This process assumes that the policy domain exists and that the plug-ins to be used have already been defined.

You use the Access System Console for all of the following steps except the last one—creating an authentication rule. To create an authentication rule for a policy domain, you use the Policy Manager.

Task overview: Defining and using a chained authentication scheme

1. Define a chained authentication scheme, as described in "Defining and Managing Authentication Schemes" on page 5-7.

   Before you can create an authentication scheme containing one or more steps, you must first define the scheme. This process includes specifying the challenge method to be used.

2. Add to the chained authentication scheme all of the plug-ins to be used for its steps, as described in "Adding and Managing Plug-Ins" on page 5-29.

   For example:

   a. Select a plug-in to be added from among the ones the Access System provides by default, or specify existing custom plug-ins.

   b. Specify the parameters for each plug-in as you add it to the scheme. You can add multiple instances of a plug-in to a scheme, each with its own set of parameters.

   c. Repeat this process for as many plug-ins as you want to add to the scheme.

   Note: When you add plug-ins to an authentication scheme, the Policy Manager creates a default step and adds them to it.

3. Add the steps of the authentication scheme, as described in "Configuring and Managing Steps" on page 5-38.

   Before you create a step, consider its purpose within the authentication scheme and in relation to other steps of the scheme.

   Planning for the steps of a scheme and the scheme's flows are interdependent processes. You can use steps to isolate plug-ins into groups. You can then connect those groups of plug-ins—that is, connect their steps—in different ways, creating different authentication flows.

Here is how to add a step to an authentication scheme:

a. Give the step a meaningful name. Well-chosen names are helpful if you rearrange the steps of a scheme.

b. Add plug-ins to the step.

c. Arrange the plug-ins in the order in which you want them executed.

Take into account how the result of a plug-in affects the result of a step for:

- The Access System

  If any Access System-provided plug-in fails, the step fails.

- Custom plug-ins

For details, see "Return Codes for Plug-Ins" on page 5-20.

Add as many steps as are necessary to complete the authentication process for your environment.

4. Create the authentication flows of the chained authentication scheme, as described in "Configuring Authentication Flows" on page 5-43.

   Plan the authentication flows of a scheme. Before you configure a scheme's authentication flows, take the time to plot the actions you want to occur for each step.

   Here is how to create a scheme's authentication flows:

   a. Determine which step you want to be executed first. Mark it the initiating step.

   b. Configure the links for the step:

   - Determine the next step to be executed if the plug-ins of a step cause the step to fail.
   - Determine the next step to be executed if the plug-ins of a step cause the step to succeed.

5. Verify the flows of the chained authentication scheme, as described in "Configuring Authentication Flows" on page 5-43.

   Test the way you configured the flows—that is, the connections between the steps creating flows—to ensure that there are no cycles.

6. Correct the flows of the chained authentication scheme, if necessary, as described in "Configuring Authentication Flows" on page 5-43.

7. Enable the authentication scheme after you are satisfied with its configuration, as described in "Enabling and Disabling Authentication Schemes" on page 5-15.

8. Create an authentication rule which includes the chained authentication scheme for the policy domain, as described in "Managing Authentication Rules" on page 5-48.

9. Specify actions for the authentication rule to be taken if authentication fails or if it succeeds based on the rule. For details, see "Managing Authentication Actions" on page 5-53.

About Authentication Steps

A step contains a group of plug-ins that are run in order of their position in the step. To connect steps in a chained authentication scheme, you specify the step to be

executed next, depending on the outcome of the present step. A different step may be executed next if the present step fails or if it succeeds. You can repeat a step in an authentication scheme. You can stop the authentication process after a step.

An authentication scheme includes one or more steps whose execution order is determined dynamically. Execution of the steps of an authentication scheme begins with the one chosen as the starting, or initiating, one. From the starting step and for each succeeding step, the step to be executed next is determined by the result of the preceding step.

Each step of an authentication scheme contains one or more plug-ins. Plug-ins within a step are executed in the order in which you position them.

At any time, you can change

- The connections between the steps of an authentication scheme.
- The order of a step's plug-ins.

Within a step, if any Access System-provided plug-in fails, the Access Server treats the step as if it failed and stops execution of the step at that point.

For information describing how the Access Server responds to a step containing a custom plug-in based on the execution result of that plug-in, see "Return Codes for Plug-Ins" on page 5-20.

Figure 5–1 illustrates a prototype for a sample authentication scheme.

Figure 5–1 Sample Authentication Scheme Prototype

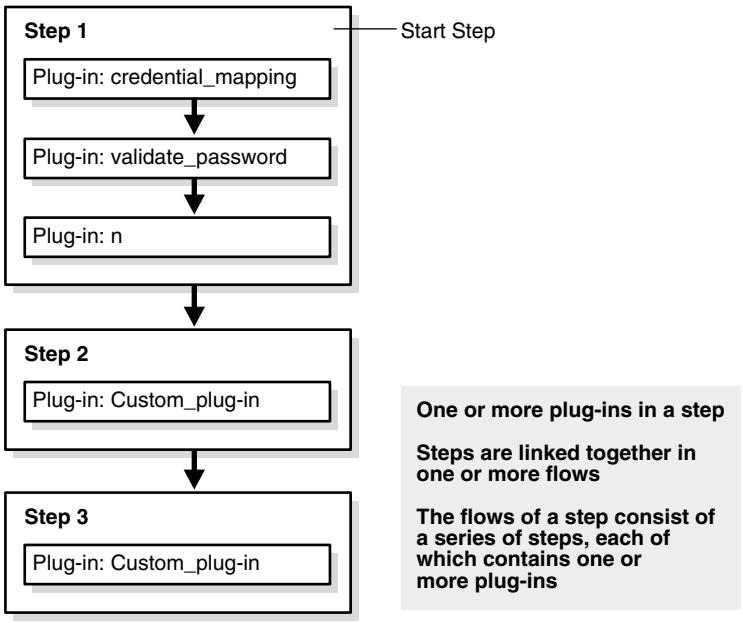


Table 5–8 summarizes the step components.

Table 5–8 Aspects of a Step

Component	Summary
Step Name	A step is a discrete entity. Each step must have a unique name.

Table 5–8 (Cont.) Aspects of a Step

Component	Summary
Plug-Ins for a Step	A plug-in provides an authentication scheme's functionality. A step can contain one or more plug-ins, but it must contain at least one. The parameters a plug-in can take are specified when the plug-in is added to the scheme, not when it is added to a step.
Number of Steps	An authentication scheme can contain any number of steps, but it must contain at least one.
Connections Between Steps	Steps are connected to form one or more flows of an authentication chain. Because steps are discrete, they can be combined in any order. Connections between steps are established by defining possible authentication flows—or flows of execution—through the authentication chain. See "Configuring Authentication Flows" on page 5-43 for details.
Execution of Steps	Steps are executed in the order in which they occur in a flow of the authentication chain. The plug-ins of steps are executed in the order in which they are positioned in a step's list of plug-ins. The order of plug-ins in a list can be changed. Execution of one step's plug-ins is followed by execution of those of the next step in the authentication flow.

About Single-Step Authentication Schemes

Many authentication schemes are simple enough to require only a single step. In such a case, the step must contain all of the plug-ins required to transact the purpose of the scheme. Because it is the only step in the authentication scheme, the authentication scheme's flow consists of execution of the step's plug-ins.

You can use the Policy Manager's authentication feature to create a single step that provides all of the functionality you may require to obtain user credentials, map them to an entry in the directory server, authenticate the user, and so forth.

It is easy to create and manage an authentication scheme with a single step, and it makes good sense to include all plug-ins in a single step in many cases. For example, a single-step authentication scheme is useful if a group of plug-ins are meant to be executed consecutively and, in the event of failure, you do not care which plug-in causes the step to fail.

Why Separate Plug-Ins Into Steps?

You may want to separate plug-ins into steps because the plug-ins form a set meant to be executed together. Also, combining the plug-ins in a step enables you to use that step in a scheme more than once. You may want to configure it as the next step to be executed for one step if that step fails, and as the next step to be executed for another step if that step succeeds.

You may also find it necessary to separate plug-ins into discrete steps even if two plug-ins form a couple logically. For example, you may want to take this approach if you must know which of two plug-ins caused the authentication process to fail.

There are many cases for which you may want to separate closely related plug-ins into discrete steps. Use of the password management feature offers an example of one case. An organization uses password management to control user access. Based on the

number of attempts specified in the password policy, it gives a user a certain number of opportunities to enter the correct password. If authentication fails, the administrator must know why. The administrator must be able to distinguish between the following two events:

- Whether authentication fails because there is no entry for the user in the directories checked.
- Whether authentication fails because the user entered the wrong password each time for the three allowed attempts.

For example, an organization uses two different parts of its directory to store user profile information for its human resources department and for its marketing department. The organization wants to be able to search across both branches of the directory for user profile information to authenticate users. The organization wants the search to begin with the search base for the human resources information and if the user profile information is not found there, continue with the search base for the marketing department.

- Search Base A
  - Includes entries for all human resources department members.
  - Examples:
    * cn=Maurice Breton
    * cn=Alice Smith
- Search Base B
  - Includes entries for all marketing department members.
  - Example:
    * cn=Sonal Kalra
    * cn=Robert Jang

The organization defines the following chained authentication steps:

- Step 1: Credential mapping
  - Success: Execute Step 2
  - Failure: Execute Step 3
- Step 2: Validate password
  - Success: Execute Step 4
  - Failure: Stop

The validate password plug-in gives the user three attempts to enter a valid password, which is based on a setting in the password policy, before it fails Step 2.

- Step 3: Custom plug-in to check Search Base B
  - Success: Execute Step 2.
  - Failure: Stop
- Step 4: Custom plug-in to do some additional processing
  - Success: Stop, return result.
  - Failure: Stop, return result

Sonal Kalra requests access to a resource protected by this authentication scheme. She enters her user name. Following is the process that occurs.

Process overview: User requests access and enters user ID

1. The Access Server searches Search Base A for an entry for Sonal Kalra.
   - There is no entry.
   - Evaluation of Step 1: failure. On failure, go to Step 3.
2. The Access Server searches Search Base B for an entry for Sonal Kalra
   - An entry with cn=Sonal Kalra is found.
   - Evaluation Step 3: success. On success, go to Step 2.
3. Sonal Kalra is prompted for her password
   - She enters the wrong password the first time. Step 2: validate password prompts her for her password three times before returning a failure.
   - At the second prompt, she enters the correct password.
   - Evaluation of Step 2: On success, go to Step 4.
4. Some additional processing is done, which completes successfully (Step 4: On success: Stop, return result).

If Sonal Kalra entered the wrong password for each of the three attempts, Step 2: validate password, would return a result of failure, and the authentication process would stop. The Delegated Access Administrator would know why the authentication process failed—not because no user entry was found for Sonal Kalra, but because she entered the wrong password three times.

About the Default Step

The first time you add plug-ins to an authentication scheme, the Policy Manager defines a default step that contains all of the plug-ins. You can modify the default step if you want to use it, or you can delete it after you add one or more additional steps to the scheme. An authentication scheme must include at least one step.

Configuring and Managing Steps

After you define an authentication scheme and add plug-ins to it, you can configure its steps. You can modify the steps of an authentication scheme at any time, but you must first ensure that the scheme is not used by any active policy domains. You can add plug-ins to a step or remove them from one, or you can delete the step.

The rest of this section discusses the following topics:

Viewing the Steps of an Authentication Scheme

You can view a list of the currently configured steps of an authentication scheme.

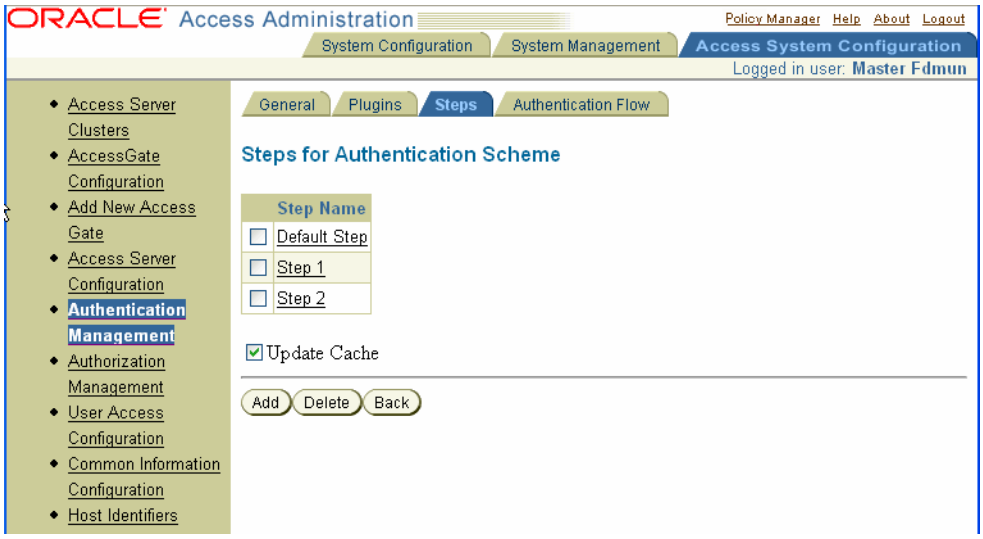
To view the steps of an authentication scheme

1. From the Access System Console, click the Access System Configuration tab.
2. Click the Authentication Management link in the left navigation pane.

   The Authentication Management: List All Authentication Schemes page appears.
3. Click the name of the authentication scheme whose steps you want to see on the Authentication Management: List All Authentication Schemes page.

   The Details for Authentication Scheme page appears. By default, the General page is displayed.
4. Select the Steps tab.

   The Steps for Authentication Scheme page appears, as illustrated in the following screen shot. This page displays the names of all the steps configured for the scheme. Each step's name is a link, which you can click to display details about the step.



If you are creating an authentication scheme and have not yet added any steps to it, or if the scheme contains only a single step, this page shows only a step called Default Step. See "About the Default Step" on page 5-38 for details.

Viewing the Configuration Details for a Step

You can view the details of the current configuration of a step for an authentication scheme any time after it is created.

To view the details for a step

1. From the Access System Console, click the Access System Configuration tab.
2. Click the Authentication Management link in the left navigation pane.

   The Authentication Management: List All Authentication Schemes page appears.
3. Click the name of the authentication scheme containing the step whose configuration details you want to see.

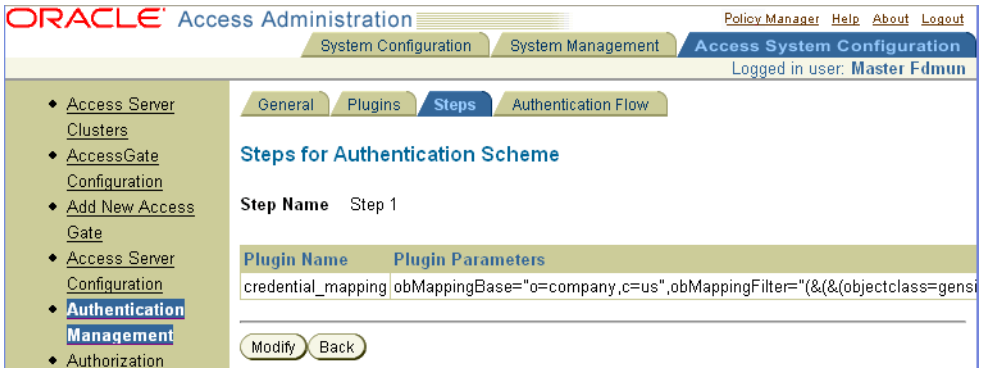
   The Details for Authentication Scheme page appears. By default, the General page is displayed.

4. Select the Steps tab.

   The Steps for Authentication Scheme page appears. This page displays the names of all the steps configured for the scheme.

5. Click the name of the step whose configuration you want to see.

   The Steps for Authentication Scheme page appears again, as illustrated in the following screen shot, this time showing the details for the selected step.



Adding a Step to an Authentication Scheme

To add a step to a scheme, you name the step and add to it the plug-ins that provide the step's functions. For steps with multiple plug-ins, the order in which you position the plug-ins in the step determines their execution order. The highest order plug-in—the one at the top of the list—is executed first.

When you add a plug-in to a step, it is placed at the bottom of the list of active plug-ins. You can rearrange the order of plug-ins in a step.

To add a step to an authentication scheme

1. From the Access System Console, click the Access System Configuration tab.
2. Click the Authentication Management link in the left navigation pane.

   The Authentication Management: List All Authentication Schemes page appears.

3. Click the name of the authentication scheme on the Authentication Management: List All Authentication Schemes page.

   The Details for Authentication Scheme page appears. By default, the General page is displayed.

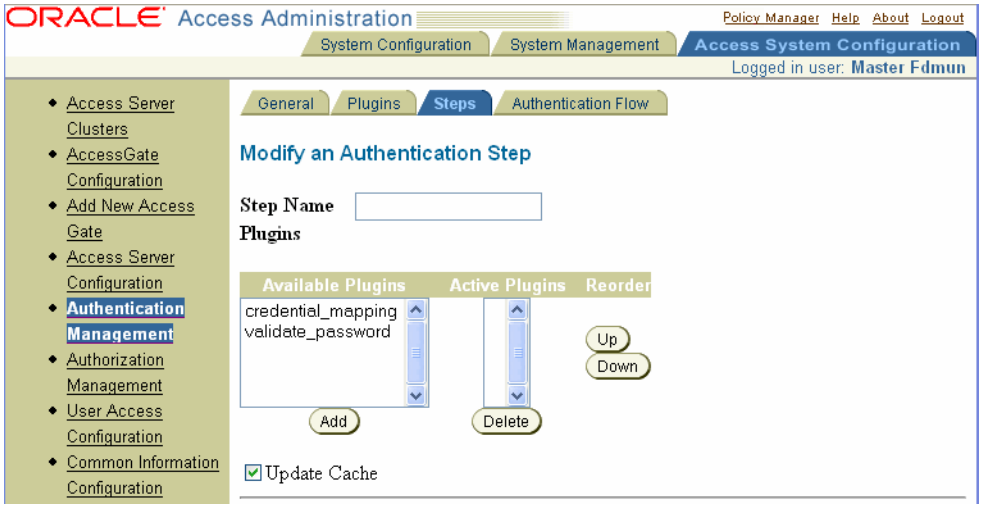
4. Select the Steps tab.

   The Steps for Authentication Scheme page appears.

   If you are creating an authentication scheme and have not yet added any steps to it, this page shows only a step called Default Step. See "About the Default Step" on page 5-38 for details.

5. Click Add.

   The Modify an Authentication Step page appears, as illustrated in the following screen shot. Note that although this page is titled Modify, it is used to add a step as well as to modify the content of an existing one.



6. Enter a unique name for the step in the Step Name text box.
7. From the list of Available Plug-ins, click the plug-in to be added to the step and click Add.

   The name of the plug-in appears in the Active Plugins scroll box.
8. Repeat Step 7 to add as many plug-ins as you want to include in the step.
9. To reposition plug-ins within the step, select the plug-in from the list of Active Plug-ins, and click the Up or Down button to move the plug-in up or down in the list.
10. Click Save.

Modifying a Step

You can modify existing authentication steps. For example, you may want to upgrade a step's plug-ins, replacing one with another, or you may want to add new plug-ins to a step to extend or change its function. You may also want to remove plug-ins which are no longer used.

To add, remove, or re-order plug-ins in an existing step

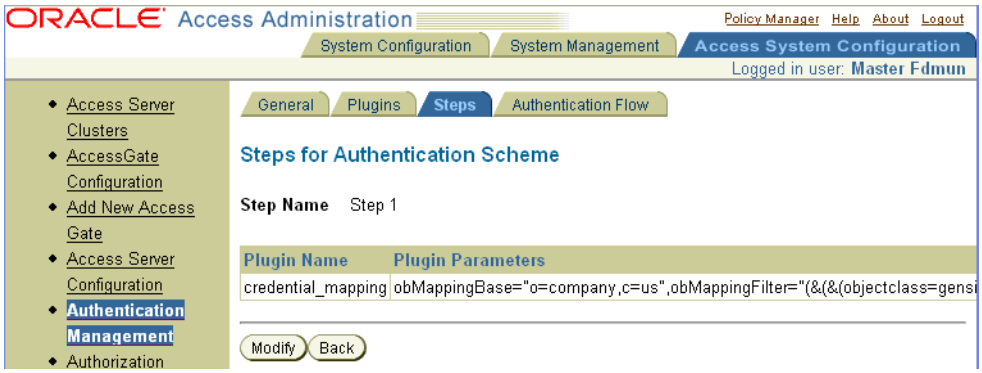
1. From the Access System Console, click the Access System Configuration tab.
2. Click the Authentication Management link in the left navigation pane.

   The Authentication Management: List All Authentication Schemes page appears.
3. Click the name of the authentication scheme whose step you want to change.

   The Details for Authentication Scheme page for that scheme appears.
4. Select the Steps tab.

   The Steps for Authentication Scheme page appears.
5. Click the name of the step that you want to modify.

   The Steps for Authentication Scheme page appears, showing the plug-ins and parameters for the step.



6. Click Modify.

   The Modify an Authentication Step page appears.

7. Change the plug-ins in the step in any of the following ways:

   - To add a plug-in to the Active Plugins list, select the plug-in from the Available Plugins list and click Add.
   - To remove a plug-in from the active list, select the plug-in from the Active Plugins list, and click Delete.
   - To change the order of plug-ins in the Active Plugins list, select the plug-in you want to move and click the Up or Down button to move the plug-in up or down in the list.

8. Click Save to save the step after you are satisfied with the changes.

Deleting a Step

You can delete one or more steps from a scheme. An authentication scheme must have at least one step.

To delete a step from an authentication scheme

1. From the Access System Console, click the Access System Configuration tab.
2. Click the Authentication Management link in the left navigation pane.

   The Authentication Management: List All Authentication Schemes page appears.

3. Click the name of the authentication scheme whose step you want to delete.

   The Details for an Authentication Scheme page for that scheme appears.

4. Select the Steps tab.

   The Steps for Authentication Scheme page appears.

5. Select the step that you want to delete.

   Select the check box for each step that you want to delete, if you want to delete more than one.

6. Click Delete.

Configuring Authentication Flows

An authentication flow is a path of execution through steps of an authentication scheme. For a single-step scheme, the authentication flow consists of running the plug-ins in the step.

Either of the following kinds of authentication schemes has an authentication flow:

- A single-step authentication scheme

  The authentication flow of an authentication scheme containing a single step consists of the flow of execution through that step's plug-ins in the order in which they appear in the step. For a description of single-step schemes, see "About Single-Step Authentication Schemes" on page 5-36.

- A chained authentication scheme

  The authentication flows of a chained authentication scheme consist of the execution of the plug-ins of one step after another in a flow. A chained authentication scheme can have one flow or many flows. The execution order of the authentication scheme's steps can vary to create different possible authentication flows, depending on the outcome of each step in a flow.

  For each step of an authentication scheme, you configure the next step to be executed based on the result of the current one. If the current step fails, the step you configured for that step's failure result is executed next. If the current step succeeds, the step you configured for that step's success result is executed next. The plug-ins of any of the steps of an authentication flow are executed in the order in which they appear in the step.

  You use the following means to configure the steps of an authentication scheme to produce various possible flows:

- Mark a step as the initiating step of the chained authentication scheme.

  All possible flows of a chained authentication scheme begin with the same step. Each authentication scheme can have only one step designated as the initiating step.

- Specify the next step to be executed if the present step fails or if it succeeds.

  This mechanism enables you to configure different flows of a chain, each of which is determined by the result of the current step.

- Use the Stop terminator.

  Any step of a chain may be followed by the Stop terminator. You can specify that execution is to stop if a step fails or if it succeeds. For either case, you set the failure condition or the success condition of the step to Stop. You can terminate execution after a step absolutely by setting both conditions of the step's result to Stop.

  You may want execution to terminate under multiple conditions for the flows of a chained authentication scheme, depending on the possible flows. Stop indicates that a flow has ended and no other steps of the authentication chain are executed.

The rest of this section discusses the following topics:

- Authentication Flows Example
- Viewing the Flows of an Authentication Scheme
- Configuring and Modifying the Flows of an Authentication Chain
- Verifying and Correcting Cycles in an Authentication Flow

Authentication Flows Example

An administrator for a company wants to organize the plug-ins used for authentication into steps so that she can more easily control the order in which they are executed. The administrator wants the result of execution of one plug-in to determine the next plug-in to be executed. If the plug-ins were to be executed in order, it would not be necessary to separate them into steps. The company uses the four plug-ins identified in Table 5–9 for its authentication process.

Table 5–9 Plug-Ins for Authentication Flow Example

Plug-In	Use
Plug-in 1: credential_mapping	Access System-provided credential mapping plug-in
Plug-in 2: validate_password	Access System-provided password validation plug-in
Plug-in 3: custom_pluginA	do_what_I_want: A
Plug-in 4: custom_pluginB	do_what_I_want: B

The administrator has determined that she wants to combine her authentication plug-ins into steps that allow her to define the following authentication flows:

- If plug-in 1 is successful (credentials mapped to user entry)

  Execute plug-in 2 (validate the user's password)

- If plug-ins 1 and 2 are successful (user's credentials map and user's password is valid)

  Execute plug-in 4 (do_what_I_want:B)

- If plug-in 1 or 2 fails (either the user's credentials cannot be mapped to an entry or the user's password is invalid)

  Execute plug-in 3 (do_what_I_want:A)

- If plug-in 3 succeeds (do_what_I_want: B),

  Execute plug-in 4 (do_what_I_want:B)

The administrator creates the three steps identified in Table 5–10.

Table 5–10 Steps for Authentication Flow Example

Step	Plug-Ins Used	Step Result
Step 1	Plug-in 1 and Plug-in 2	Succeeds if both Plug-in 1 and Plug-in 2 succeed. Fails if either of the plug-ins fails.
Step 2	Plug-in 3	Succeeds if Plug-in 3 succeeds. Fails if Plug-in 3 fails.
Step 3	Plug-in 4	Succeeds if Plug-in 4 succeeds. Fails if Plug-in 4 fails.

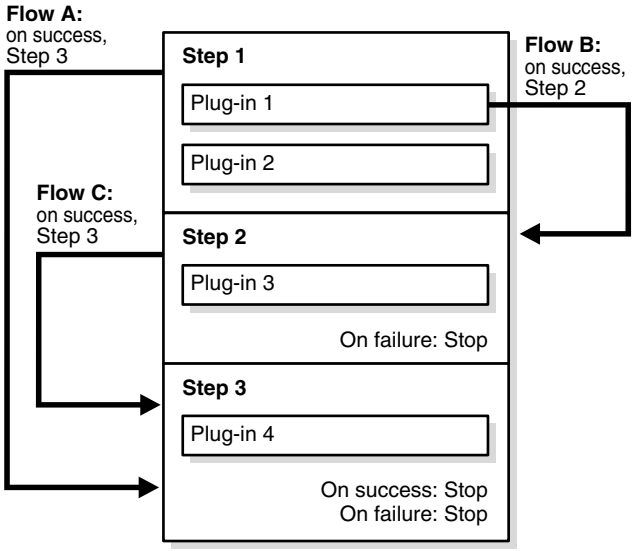
She combines the plug-ins in Table 5–9 with the steps in Table 5–10 to create the desired authentication flows. Table 5–11 shows the steps of the authentication scheme and which step is executed next if the step succeeds or if it fails.

Table 5–11 Outcome of Steps for Authentication Flow Example

Step	On success	On failure, executes
Step 1	Step 3	Step 2
Step 2	Step 3	Stop
Step 3	Stop	Stop

Figure 5–2 provides a diagram of the authentication flow table.

Figure 5–2 Illustration of Authentication Flow in Table 5–11



Viewing the Flows of an Authentication Scheme

At any time after you configure the authentication flows for an authentication scheme, you can look at the configuration by selecting the Authentication Flow tab. The Flow of the Authentication Scheme page shows the current configuration.

After you add a step to an authentication scheme, by default the Policy Manager assigns the Stop terminator to the On Success Next Step and On Failure Next Step result conditions of each step. The Flow of the Authentication Scheme page shows this default configuration until you modify it.

To view the configuration of an authentication flow

1. From the Access System Console, click the Access System Configuration tab.
2. Click the Authentication Management link in the left navigation pane.

   The Authentication Management: List All Authentication Schemes page appears.
3. Select the name of the authentication scheme whose authentication flows you want to view.

   The Details for Authentication Scheme page for the authentication scheme appears.
4. Select the Authentication Flow tab.

   The Flow of the Authentication Scheme page appears.

Configuring and Modifying the Flows of an Authentication Chain

After you add steps to an authentication scheme, you can configure the possible flows of execution through the steps. You use the Flow of the Authentication Scheme page to configure the On Success Next Step and On Failure Next Step result conditions for each step.

At any time, you can use the same page to modify the flows of the authentication scheme. You can change the links between steps in a chain to correct cycles or to redirect flows.

To configure the flows of an authentication scheme

1. From the Access System Console, click the Access System Configuration tab.
2. Click the Authentication Management link in the left navigation pane.

   The Authentication Management: List All Authentication Schemes page appears.
3. Select the name of the authentication scheme whose authentication flows you want to configure.

   The Details for Authentication Scheme page for the authentication scheme appears.
4. Select the Authentication Flow tab.

   The Flow of the Authentication Scheme page appears. For existing steps, this page shows the connections between steps of the chain. If there is only one step for the scheme, it appears here.
5. Click Modify

   The Flow of the Authentication Scheme page with modifiable entries appears. The page shows the names of the scheme's steps. For each step, the page includes lists from which to choose the next step to be executed if the current one succeeds or if it fails.
6. Choose the step to be used as the initiating step by selecting the radio button for the step in the Initiating Step column.

   Only one step can be configured as the Initiating step.
7. For each step in the Step Name column, complete the following:

   a. In the list under the On Success Next Step column, select the next step to be executed if the present one succeeds.

   b. In the list under the On Failure Next Step, select the next step to be executed if the present one fails.

      If you want execution to terminate after a step is completed, select the Stop terminator. You can use Stop for success of a step or for failure of a step.

   Both selection lists show the names of all steps configured for the chained authentication scheme.
8. After you are satisfied with the configuration, click Verify Flow to determine if it contains cycles.

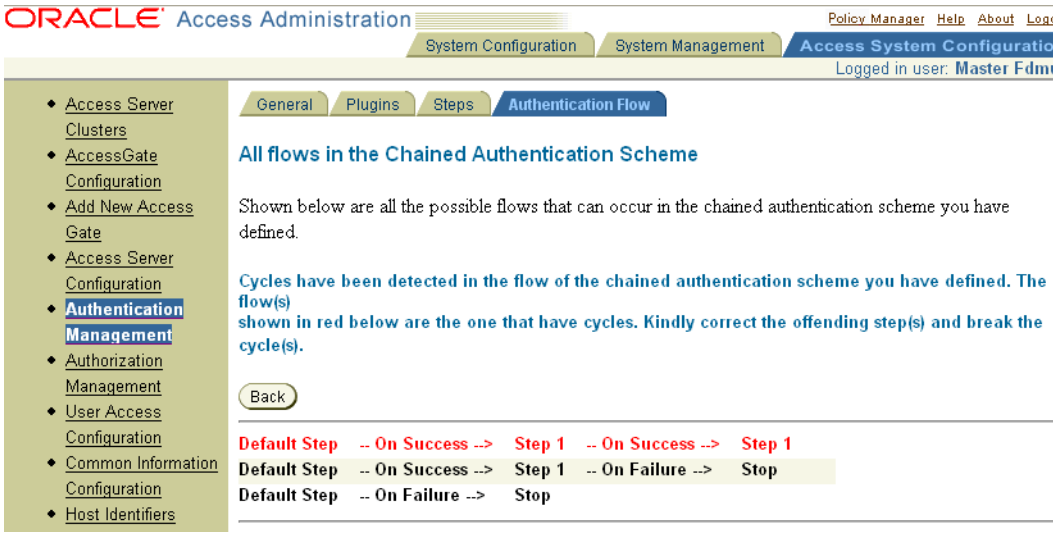
   See "Verifying and Correcting Cycles in an Authentication Flow" on page 5-47 for details.
9. Click Save after you have determined that there are no cycles in the flows.

Verifying and Correcting Cycles in an Authentication Flow

Because the flows of an authentication chain can be complex, it is possible for a chain to include cycles.

After you define how the steps of a scheme are connected, you can click the Verify Flows button to check the configuration for cycles before you save it.

The Verify Flows button appears when you click the Authentication flow sub-tab, then click Modify, as illustrated in the following screen shot.



If the authentication flow's configuration contains cycles, the Policy Manager identifies the offending flow on the All Flows in the Chained Authentication Scheme page. You cannot save the configuration until you correct the cycles. If you attempt to save an authentication flow's configuration without having verified it first, the Policy Manager automatically checks the configuration to ensure that none of its flows contain cycles.

Although the Policy Manager verifies the authentication flows to check for cycles, it is good practice to plot the flows of a complex authentication scheme well before you configure them. To correct flows containing cycles after they are reported on the All Flows in the Chained Authentication Scheme page, you use the Flow of the Authentication Scheme page.

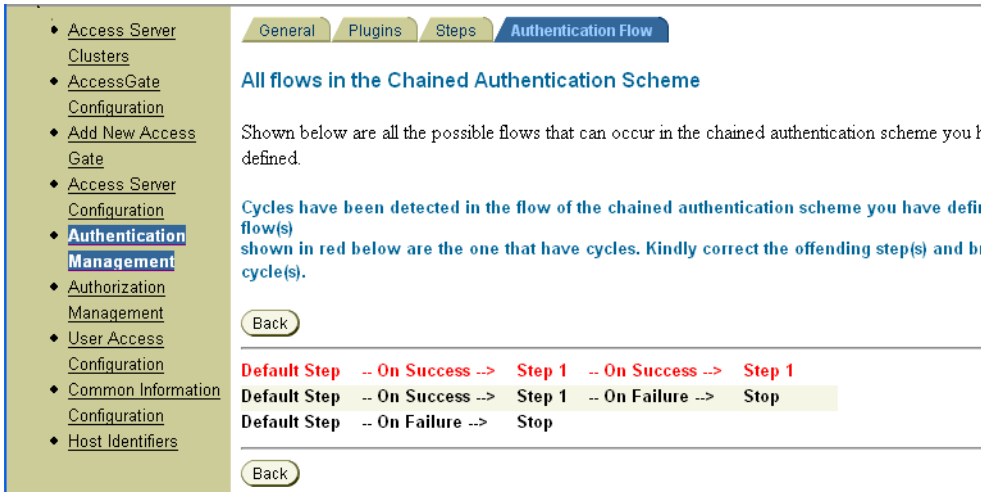
The All Flows in the Chained Authentication Scheme page shows all of the configured flows, depicting them in the following way:

- Flows without cycles are shown in black.
- Flows with cycles are shown in red.

To correct an authentication flow containing a cycle

1. From the Access System Console, click Access System Configuration tab, then click Authentication Management, in the left navigation pane.
2. Click the link for the multi-step authentication scheme.
3. Click the Authentication Flow sub-tab.
4. Click Modify.
5. Click Verify Flow.

6. If there is an error in the flow of the steps, the reported flow and its offending step in the flows display of the All Flows in the Chained Authentication Scheme page, as illustrated in the example of a flow with cycles in the following screen shot.



If the verification process reports multiple flows containing cycles, note and correct all of them.

7. Click Back on the All Flows in the Chained Authentication Scheme page, which reports the offending flow.

   The Flow of the Authentication Scheme page appears.

8. Correct the problem within the flow that contains the cycle.

   "Configuring and Modifying the Flows of an Authentication Chain" on page 5-46 describes the process to use to create authentication flows. Follow this process to modify the connections between the offending steps.

9. Click Verify Flow.

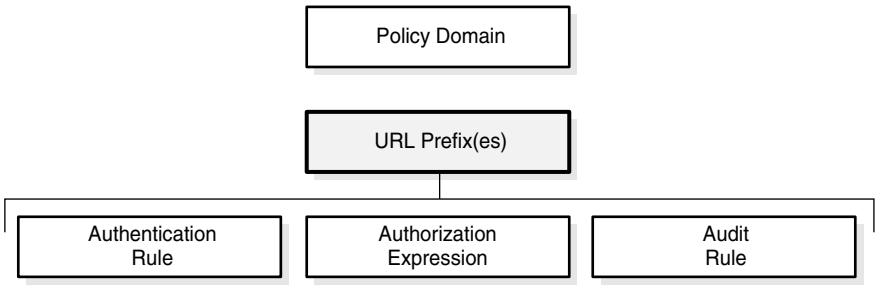
   If the verification results show more flows with cycles, continue to correct the flow.

10. After all problems causing the cycle are resolved, click Save.

Managing Authentication Rules

Each policy domain must include a single default authentication rule, and each policy in a domain can include an authentication rule specific to the policy. If a policy does not include an authentication rule, it inherits protection by the default authentication rule established for the entire policy domain. Figure 5–3 illustrates conceptually the set of default rules for a policy domain, among which is an authentication rule. In this example, no policies have been created yet for the policy domain.

Figure 5–3 Default Rules for a Policy Domain



An authentication rule includes an authentication scheme that specifies the kind of authentication required to verify a user's identity, the directory server to be checked for user information, and so on. See "About Authentication Schemes" on page 5-3 for details.

Whenever a user requests access to a resource protected by an authentication rule, the user must authenticate using the challenge method specified by the rule's scheme.

Delegated Access Administrators can create authentication rules for the policy domains and their policies for which they have administrative rights.

The rest of this section discusses the following topics:

- Creating an Authentication Rule for a Policy Domain
- Modifying an Authentication Rule for a Policy Domain
- Deleting a Policy Domain's Authentication Rule
- Creating an Authentication Rule for a Policy
- Modifying an Authentication Rule for a Policy
- Deleting an Authentication Rule for a Policy

Creating an Authentication Rule for a Policy Domain

For each policy domain, you must define a single default authentication rule.

To create a default authentication rule for a policy domain

1. From the landing page for the Access System, select the Policy Manager link.

   If you are working in the Access System Console, click the link for the Policy Manager at the top of the page.

2. Click My Policy Domains in the left navigation pane.

   A list of policy domains appears.

3. Click the link for the policy domain that you want to view.

   The General page for the selected policy domain appears.

4. For the selected policy domain, select the Default Rules page.

   If there is an authentication rule already configured for the policy domain, the Authentication Rule page appears showing the definition of the rule.

   There can be only one default authentication rule for a policy domain. If there is an existing default authentication rule, you must delete it before you can add a new

one. For details, see "Deleting a Policy Domain's Authentication Rule" on page 5-51.

5. Click the Add button on the Authentication Rule page.

   The General page for the Authentication Rule appears.

6. Enter a Name for the default authentication rule.

7. Enter a Description for the default authentication rule.

8. Select an authentication scheme.

   The list shows enabled authentication schemes created by the Master Access Administrator. To add new schemes, see "Defining and Managing Authentication Schemes" on page 5-7. Authentication schemes that are disabled do not appear in the list.

9. Click Save.

Modifying an Authentication Rule for a Policy Domain

You can modify the authentication rule for any policy domain for which you have administrative rights, including any policy domain that you have created.

To modify a policy domain's authentication rule

1. From the landing page for the Access System, select the Policy Manager link.

   If you are working in the Access System Console, click the link for the Policy Manager at the top of the page.

2. Click My Policy Domains in the left navigation pane.

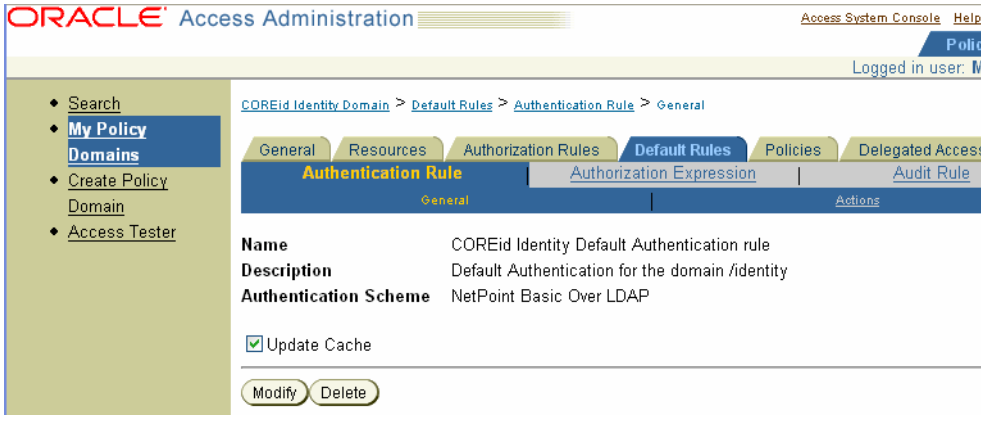
   A list of policy domains appears.

3. Click the link for the policy domain that you want to view.

   The General page for the selected policy domain appears.

4. Click Default Rules.

   The General page for the Authentication Rule tab appears. It shows the current configuration for the rule.



5. Click Modify.

   The General page, whose fields you can modify, appears.

6. Change the Name, Description, and Authentication Rule as necessary.

7. Click Save to save your changes or click Cancel to exit the page without saving.

Deleting a Policy Domain's Authentication Rule

Because a policy domain can have only one authentication rule, you must delete the existing rule before you can add a new one.

To delete a policy domain's authentication rule

1. From the landing page for the Access System, select the Policy Manager link.

   If you are working in the Access System Console, click the link for the Policy Manager at the top of the page.

2. Click My Policy Domains in the left navigation pane.

   A list of policy domains appears.

3. Click the link for the policy domain that you want to view.

   The General page for the selected policy domain appears.

4. Click Default Roles

   The General page for the Authentication Rule tab appears showing the currently configured rule.

5. Click Delete.

   Answer Yes to the prompt, to confirm the deletion.

Creating an Authentication Rule for a Policy

For any policy domain, you can create special policies for groups of resources within the domain. All resources of a policy domain are protected by its default authentication rule unless the resource is covered by a policy containing a different authentication rule. You define an authentication rule for a policy just as you would for a policy domain, but you define the rule in association with the policy.

If an authentication rule exists for the policy and you want to replace it, you must delete the rule before you can create a new one. See "Deleting an Authentication Rule for a Policy" on page 5-53 for details.

To create an authentication rule for a policy

1. From the landing page for the Access System, select the Policy Manager link.

   If you are working in the Access System Console, click the link for the Policy Manager at the top of the page.

2. Click My Policy Domains in the left navigation pane.

3. Click the link for a policy domain.

   The General page for the selected policy domain appears.

4. Click the Policies tab.

   The Policies page appears listing all of the existing policies, if any.

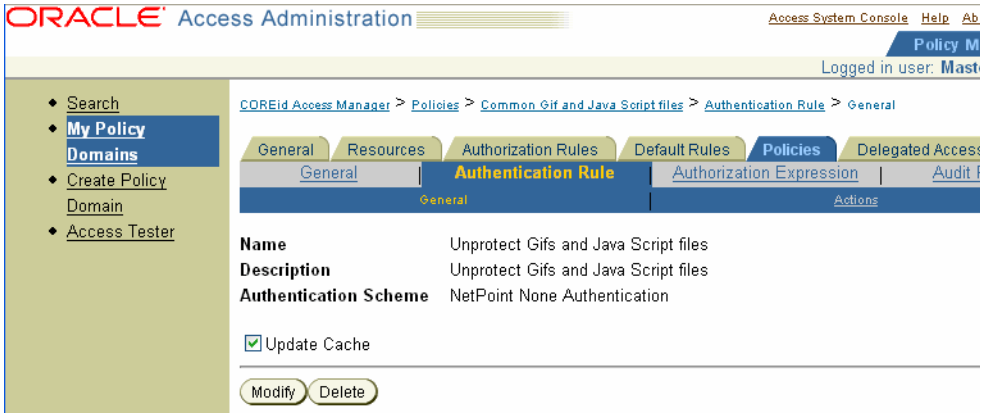
5. Click the link for the Policy for which you want to add an authentication rule.

   The General page showing the configuration for the policy appears.

6. Click the Authentication Rule sub-tab for the policy.

7. Click Add.

   The General page for defining an authentication rule appears.



8. Enter a Name for the default authentication rule.
9. Enter a Description for the default authentication rule.
10. Select an authentication scheme.

    The list shows the authentication schemes created by the Master Access Administrator. To add new schemes, if required, see "Defining and Managing Authentication Schemes" on page 5-7.

11. Click Save.

Modifying an Authentication Rule for a Policy

You can modify the authentication rule for a policy within a policy domain for which you are granted administrative rights and for a policy within a policy domain that you have created.

To modify a policy's authentication rule

1. From the landing page for the Access System, select the Policy Manager link.

   If you are working in the Access System Console, click the link for the Policy Manager at the top of the page.

2. Click My Policy Domains in the left navigation pane.
3. Select the policy domain whose authentication rule you want to modify.

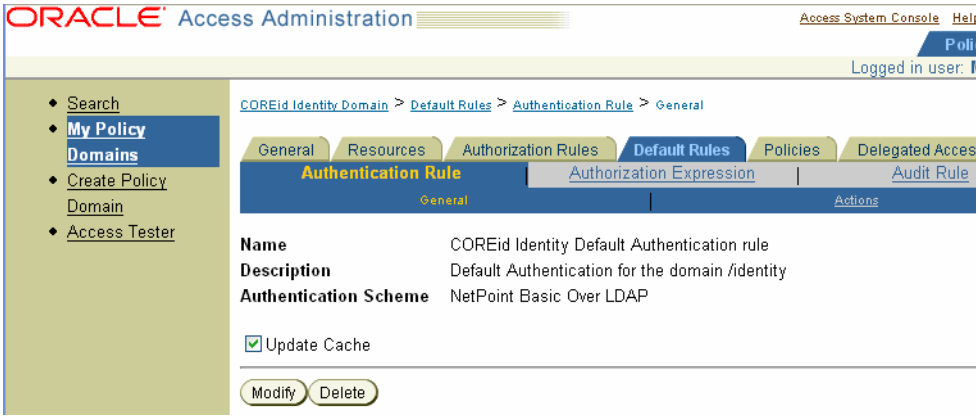
   The General page for the selected policy domain appears.

4. Select the Policies tab to display a page listing all existing policies.
5. From the list of policy names, select the Policy whose authentication rule you want to modify.

   The Policies General page appears showing the configuration for the policy.

6. Select the Authentication Rule tab.

   The Authentication Rule General page appears, listing the definition of the authentication rule, as illustrated in the following screen shot.



7. Click Modify.

   The Authentication Rule General page form appears enabling you to edit the information using text boxes and a list.

8. Modify the definition of the policy's authentication rule as necessary, changing its name, description, or the authentication scheme it includes.

9. Click Save.

Deleting an Authentication Rule for a Policy

You can delete the authentication rule for a policy within a policy domain if you are granted administrative rights, and for a policy within a policy domain that you have created.

To delete a policy's authentication rule

1. From the landing page for the Access System, select the Policy Manager link.

   If you are working in the Access System Console, click the link for the Policy Manager at the top of the page.

2. Click My Policy Domains in the left navigation pane.

   The General page for the selected policy domain appears.

3. For the selected policy domain, select the Policies tab.

   The Policies page appears listing all of the existing policies.

4. Select the Policy whose authentication rule you want to delete.

   The General page showing the configuration for the policy appears.

5. Select Authentication Rule.

   The General page showing the definition of the authentication rule appears.

6. Click Delete.

   Answer Yes to the confirmation prompt.

Managing Authentication Actions

You can optionally configure an authentication rule to perform actions based on the outcome of the rule, or by the success or failure of the authentication.

Actions allow you to pass user profile information for the user who requested the resource to other applications, or to redirect the user's browser to another site.

Actions are used in the following ways:

- If an Allow result is returned, the actions of the rule that determined the allow result are taken.
- If a Deny result is returned, the actions of the rule that determined the deny result are taken.

The rest of this section discusses the following topics:

Note: When configuring actions for an Active Directory forest using ADSI, ensure the administrative account is set to AD Domain/administrator in the Windows Directory Security: Authentication and Access Control manager.

About Kinds of Actions

Actions allow you to:

- Redirect the user's browser to another URL.

  You can redirect URLs from the Access Server to an AccessGate or a WebGate.

- Pass information about the user to downstream applications in the same policy domain or a different one.

  Using HTTP header variables or cookies, you can use actions to pass the following kinds of information:

  – User profile information

  – A user's DN

  – Static text strings

See "About the Use of HTTP Header Variables and Cookies" on page 5-55 for details about using header variables to pass information to downstream applications.

Note: Redirection and use of header variables are mutually exclusive.

About the Use of HTTP Header Variables and Cookies

The following are guidelines for HTTP header variables and cookies.

- Non-ASCII characters are not permitted in HTTP headers or cookies.

  For this reason, non-ASCII characters are not supported in the header variable Name and Return Attribute fields when you define authentication rule actions.

- Consider the 4K size limit of the HTTP header when you use HTTP header variables and cookies to pass information to downstream applications.

  This size limit includes all cookies, server variables, and environment variables—that is, all of the content of the HTTP header. There is no constraint on the number of individual elements an HTTP header can contain if the content does not exceed the 4K limit. When assessing the available space in the HTTP header, take into account the byte size of the data used by the applications. For example, if the Identity System and other applications combined use 1K in the HTTP header, you have 3K for your data.

Passing Information Using Actions

You can use actions for many purposes. The following table provides some examples of how to use actions.

Table 5–12 Examples of how to use actions

Task	Example
Personalizing the end user's interaction with the receiving application	You can use an authentication action to send the user's name to a downstream application. The application could use the name to greet the user with a personalized message when the user logs in.
Passing information in a header variable	You can use a header variable: ■ To pass membership information ■ To pass information about a user for purposes of single sign-on For single sign-on to work, the target application must be able to use the variable.
Redirecting users to a specific URL upon failure or success of the attempt to authenticate	You can use redirection to send the user to another location. For example, you can redirect a user to your portal page following authentication through your custom form.

Important: Redirection and use of header variables are mutually exclusive.

Actions and Header Variables

You can use HTTP header variables as vehicles for passing static values or identifying attributes. Authentication actions occur once during a user's session—when the user logs in. Header variables passed as authentication actions are not persistent during a user's session. A header variable is limited to 4KB. This size includes cookies, the server, and environment variables.

You can redirect header variables for authentication success and authentication failure to only those Web servers that are known or protected by the Access System. Header variables are not redirected outside of Oracle Access Manager.

For example, you could enter `HTTP_HELLO` in the Name field and `cn` in the Return Attribute field. In this case, the Access System sends a value to the Web server in the HTTP header called `HTTP_HELLO`, including the user's common name for the `cn` attribute. An application could then examine this HTTP header variable and display the value using application code to personalize an interface to include the user's name.

If the attribute contains multiple entries, such as phone numbers, the Access System returns them as a single string in colon-separated format. End users must parse the individual values themselves.

The `obmygroups` parameter populates a header variable with the names of groups that the user is a member of. You can use the syntax `obmygroups:`*`ldap_url`* to apply a filter to the groups that a user is a member of. Separate multiple entries in the filter with a colon (":"). For example, to return all of the groups in the DN that the user is a member of and that have the group_type set to role, you would enter the following in the Return Attribute field:

```
obmygroups:ldap:///o=company,c=us??sub?(group_type=role)
```

Note: For performance implications of using obmygroups, and tuning tips, see the *Oracle Access Manager Deployment Guide*. You may find that using an IdentityXML call, userGroupsProfile, is a less resource-intensive method for returning the groups that are associated with a user. See the chapter on configuring IdentityXML parameters in the *Oracle Access Manager Developer Guide* for details.

How Caching Header Variables Affects their Availability

If a header variable's value is changed, the new value is not available until the Access Server cache is refreshed.

There are two cache timeout parameters that affect header variables:

- **User Cache Timeout**: When an attribute in the header variable is obtained from the directory, it is placed in the user cache. If the value of this attribute changes and there is no user cache flush request for that user, the Access Server does not know about this change until the user cache timeout occurs. At this point, the Access Server retrieves the data again from the directory.
- **Policy Cache Timeou**t: For policy data, if a user changes the return attribute in an action, and this change does not reach the Access Server (for instance, if a cache flush failed), the Access Server does not know about this action until the policy cache timeout limit is reached.

Ways Different Web servers Handle Header Variables

Web servers process header variables differently. This variability affects how you must implement header variables in your applications.

Here are some examples:

- Netscape/iPlanet Web servers precede Access System variables with the string, HTTP:
  - If you define a variable called HTTP_CN, Netscape/iPlanet produces a variable called HTTP_HTTP_CN.

  - When you write an application that must read a header variable, the application must look for a variable called HTTP_HTTP_CN and not HTTP_CN.

- Microsoft IIS expects header variables to be defined with a dash, not an underscore. You would enter HTTP–CN, not HTTP_CN.

  The receiving application must read the variable as if it had an underscore. It looks for HTTP_CN, not HTTP–CN.

- The Lotus Domino Web server cannot pass Access System header variables.

  For information about how to use header variables for various servers, refer to your Web server's documentation.

Using Actions for Redirection

You can use actions to redirect a user from the target page to a different one. You can use form-based authentication to send users to another page when authentication succeeds, rather than to the originally requested URL. This is a popular use of redirection.

For example, a user might request www.dirac.com/spin/index.htm.You could create a custom form to be used to challenge the user. After the user is authenticated based on information they enter in the custom form, you might redirect the user to your main portal page. You could redirect the user's browser instead of sending the user to the resource requested initially. To do so, you enter the portal page URL in the redirect field when you configure the action.

You may want to redirect a user upon authentication failure if you want them to see a more informative Web page than the standard HTTP-404-Page Not Found.

Note: If you redirect a user upon authentication success or failure, the Access System does not pass the header variables. Oracle considers passing header variables on redirection a security risk. Thus, on the page where you are configuring the Authorization Action for Success, Failed, and Inconclusive, configuring to return HeaderVars for Authorization Failure and Authorization Inconclusive are not feasible.

Using Form-Based Authentication Instead of a Plug-In

Instead of implementing a plug-in to prompt your users for two levels of authentication information, you may want to use two consecutive form-based authentication screens.

You can design two HTML forms, each of which has text fields for users to enter credentials. You define credential mapping for each login form. You present the user consecutively with the two HTML form-based screens. When the user clicks on the form's submit button, the form data is intercepted and processed by WebGate before it is posted to the Web server. The WebGate searches the directory for profiles with attributes matching the form credentials.

For example, Arete Airlines provides employees with personal flight benefits accrued over time. The IT department of the airline has implemented a form-based authentication system to present two consecutive HTML form-based screens to the user. Each form requests a different kind of information for user authentication, and each form has its own security level:

- **First screen**: Prompts the user for Employment Area and Organization Number.

  This form-based authentication method may have a low security level, such as 1, because many people know the information.

- **Second screen**: Prompts for the user's Personal Information Number (PIN).

  This form-based authentication method may have a high security level, such as 3, because the information is private, identifying the user exclusively.

Process overview: Form-based authentication from the user's perspective

1. The user clicks a link on the company human resources site for employee flight benefits.
2. The application presents the user with the first form-based HTML page, prompting the user for department information.
   - If authentication succeeds for the first screen input, the user is presented with the second form-based HTML page.
   - If the user's PIN is authenticated, the user is granted access to the resource.

For more information about form-based authentication, see "Form-Based Authentication" on page A-1.

Custom Actions

If you want to customize the action taken in response to an authentication result, you can create your own actions.

To implement custom actions, you create a plug-in to be called in response to the authentication result.

You can design your external code to execute any number of actions. Some examples are:

- Accessing a relational database using required parameters
- Passing the user name of the user who has successfully been authorized for a resource
- Adding optional parameters that define a user's access

Setting Authentication Actions

You use the Actions page of the authentication rule page to create authentication actions. Actions are optional. You can specify them for authentication failure, authentication success, or both.

Task overview: Setting authentication actions for a policy domain includes procedures

1. To set authentication actions for a policy domain
2. To complete the authentication actions for a policy domain

To set authentication actions for a policy domain

1. From the landing page for the Access System, select the Policy Manager link.

   If you are working in the Access System Console, click the link for the Policy Manager at the top of the page.

2. Click My Policy Domains in the left navigation pane, and then click the name of the desired policy domain in the Name column of the table.

   The General page for the selected policy domain appears.

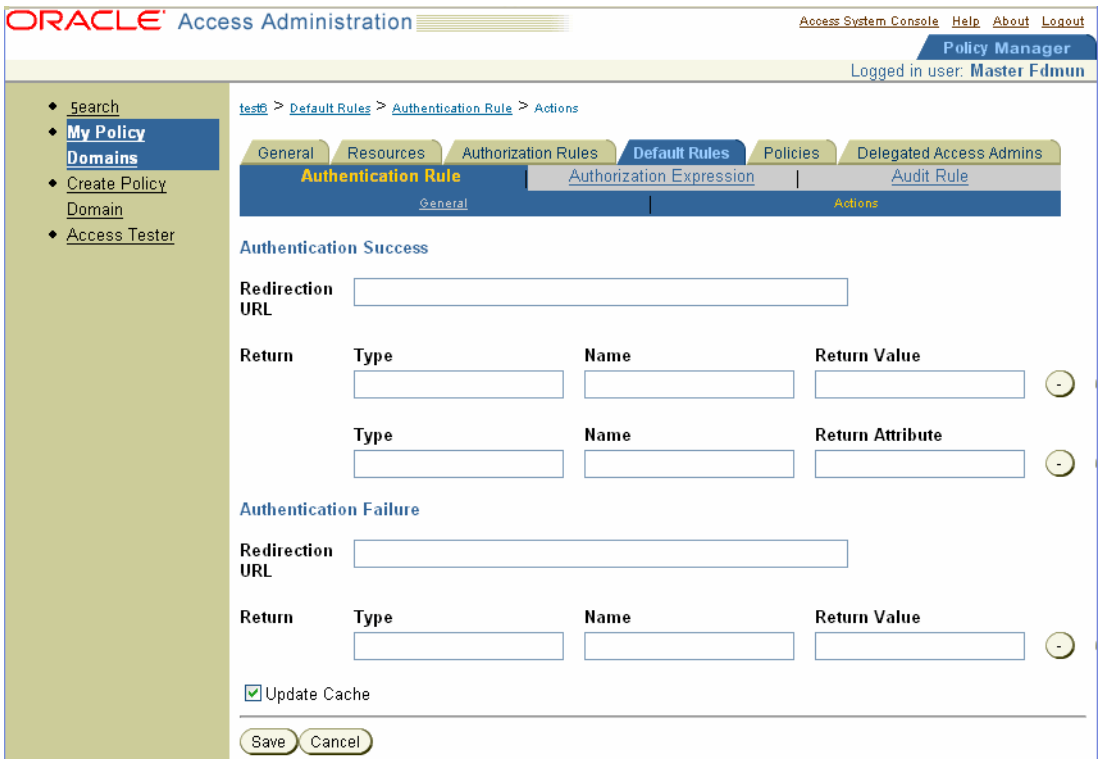
3. Click the Default Rules tab.

   The Authentication Rule sub-tab appears with the General panel selected. This page also has an Actions panel.

4. Click the Actions panel.

5. If no actions have been created, click Add (if actions have been created, click Modify).

   The Actions page for the authentication rule appears, as illustrated in the following screen shot.



6. Specify the actions to be taken in response to successful authentication of the user in the Authentication Success text boxes.

   For details, see the next procedure.

7. Specify in the Authentication Failure text boxes the actions to be taken if authentication of the user fails.

   For details, see the next procedure.

8. Determine when you want Access Server caches to be updated.

   - Select Update Cache if you want all Access Server caches to be updated immediately with information about this new prefix.
   - If you do not select Update Cache, the Access Server caches are updated when they time out and read new information from the directory server.

9. Click Save to save your input and return to the previous page (or click Cancel to return to the previous page without saving).
10. See the following procedure to complete the authentication actions for the policy domain.

To complete the authentication actions for a policy domain

1. From the Default Rules tab for the selected policy domain, click the Actions sub-tab and then click Modify.

   Header variables for both Authentication Success and Authentication Failure can be redirected only to Web servers known to or protected by the Access System. Header variables are not redirected outside of Oracle Access Manager.

2. In the Redirection URL field for both Authentication Success and Authentication Failure, type the complete path to a URL where the end user's browser is sent after the request is received.For example:

   - For authentication success, you could use the following URL to redirect the end user to a portal index page.

     `http://mycompany.com/authnsuccess.htm`

   - For authentication failure, you could use the following URL to redirect the end user's request to an error page or a self-registration script.

     `http://mycompany.com/authnfail.htm`

3. In the Return Type field of both, specify the method the Access System uses to send the value to the AccessGate.

   The method you specify must be recognized by your AccessGate. An AccessGate can use these types of methods:

   - `headervar`
   - `cookie`

   If you are using a client written with the Access Manager API, you can pass any alphanumeric string as the type and the client can interpret it.

   For details about HTTP header variables, see "About the Use of HTTP Header Variables and Cookies" on page 5-55 and "Actions and Header Variables" on page 5-55.

   Note: If you leave the Type field blank, and then click + to add another field (or click Save), the Access System uses headervar as the default.

4. In the Name field, enter a variable name that defines your return value or return attributes, such as REMOTE-USER to return the UID.

   Your applications must be configured in advance to accept the variables you enter in these fields. Non-ASCII characters are not permitted in the header variable Name field.

5. In the Return Value field, enter the value that must be assigned to the associated Name variable when the user is authenticated.
6. In the Return Attribute field, enter the LDAP attributes included in the response from the requesting user's Profile.

Click the + or – icons to add or remove fields as needed.

Note: If the returned value contains a special character (such as \ or :), these characters are escaped with a backslash (\). The obUniqueid special attribute returns the DN. Non-ASCII characters are not permitted in the header variable Return Attribute field.

7. After you define the actions, return to "To complete the authentication actions for a policy domain" on page 5-60 to complete configuration of actions for the policy domain.

Defining Actions for a Policy's Authentication Rule

For every policy, you can define actions for that policy's authentication rule to be taken in response to successful authentication of a user or failure to authenticate a user. Actions are optional. You can specify them for authentication failure and authentication success, or both.

Task overview: Defining actions for a policy's authentication rule includes procedures

1. To set authentication actions for a policy
2. To define actions for a policy

To set authentication actions for a policy

1. From the landing page for the Access System, select the Policy Manager link.

   If you are working in the Access System Console, click the link for the Policy Manager at the top of the page.
2. Click My Policy Domains in the left navigation pane.
3. From the My Policy Domains page, select the name of the policy domain where you want to set authentication actions.

   The General page for the selected policy domain appears.
4. For the selected policy domain, select the Policies tab.

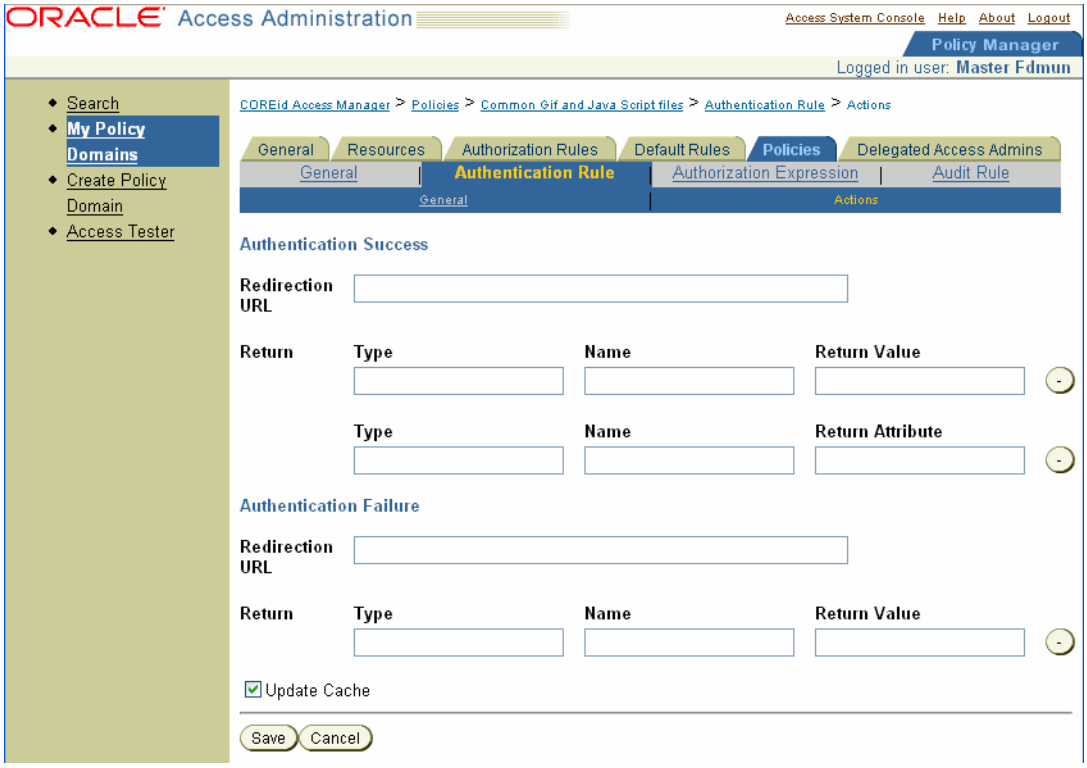
   The Policies tab lists all of the existing policies.
5. Select the policy for which you want to define authentication rule actions.

   The policy page appears, showing the General panel for the policy. There are several other panels on this page, including Authentication Rule, Authorization Expression, and Audit Rule.
6. Select the Authentication Rule panel.

   The Authentication Rule page appears, with a General panel selected. The page also contains an Actions panel.

   If you are defining the rule, see "About Creating an Authentication Rule Using Chained Authentication" on page 5-33.
7. Click the Actions panel.
8. Click Add.

   An entry form for defining the authentication rule's actions appears.



9. Specify in the Authentication Success and Authentication Failure text boxes the actions to be taken in response to successful authentication of the user.

   For details, see "To define actions for a policy" on page 5-62.

10. Determine when you want Access Server caches to be updated.
    - Select Update Cache if you want all Access Server caches to be updated *immediately* with information about this new prefix.
    - If you do not select Update Cache, the Access Server caches are updated when they time out and read new information from the directory server.

11. Click Save, and then define actions for the policy as described next.

To define actions for a policy

1. From the Policies tab for the selected policy domain, click Authentication Rules, and then click Actions, if needed.
2. Click the Modify button on the Actions page.
3. In the Redirection URL fields, type the complete path to a URL where the end user's browser is to be sent after the request is received. For example:
   - For authentication success, use this field to redirect the end user to a portal index page.

     `http://mycompany.com/authnsuccess.htm`
   - For authentication failure, use this field to redirect the end user's request to an error page or a self-registration script.

     `http://mycompany.com/authnfail.htm`

4. In the Return Type field, specify the method the Access System uses to send the value to the AccessGate.

   The method you specify must be recognized by your AccessGate. An AccessGate can use these types of methods:

   - headervar
   - cookie

   If you are using a client written with the Access Server API, you can pass any alphanumeric string as the type and the client can interpret it.

Triggering Authentication Actions After the ObSSOCookie is Set

After a user successfully authenticates, but before the user is served a requested resource, the Access System performs any authentication actions that were defined in the authentication rule that protects the requested resource. Authentication actions may include passing attributes to the protected application to customize the user's experience, perform redirections, and so on. In addition to performing authentication actions, a cookie known as ObSSOCookie is set for the user. The ObSSOCookie allows the user to access the resource repeatedly without re-authenticating. See "Configuring Single Sign-On" on page 7-1 for details.

Under some circumstances, the ObSSOCookie may be set before the authentication actions have been triggered. For example, suppose the user requests a resource that is protected by a form-based authentication scheme that redirects the user to a form with several options for logging in. When the user selects a login method on the form, he or she is again redirected, this time to a form containing a certificate-based authentication scheme. In this scenario, when the user authenticates and is redirected to the requested resource, the ObSSOCookie will have already been set and any authentication actions that exist for the target resource are bypassed.

The Access System provides a key named ObTriggerAuthentication (OTA) that triggers authentication success actions for a target resource after the ObSSOCookie has been set. To make use of this trigger, you configure parameter named OTA in an authentication scheme and define corresponding authorization success actions for each domain where you want the OTA scheme to work.

About the OTA Authentication Scheme

To trigger the authentication actions that are defined for a target resource, you usually pass information such as a header variable. The presence of the ObSSOCookie usually indicates that authentication actions have already been performed and should be bypassed. The OTA:true parameter in an authentication scheme acts as a trigger for authentication actions. When an authentication scheme contains the OTA:true parameter, the scheme sets a value of OTA=true in the ObSSOCookie. When the user is redirected back to the requested resource, the value of OTA=true in the ObSSOCookie forces execution of the authentication actions.

To configure the OTA authentication scheme, you must also define a cookie named NoExecuteOTA in the policy domain that protects the resource. The NoExecuteOTA cookie ensures that only specific policy domains can make use of the OTA:true trigger. If you implement single- or multi-domain single sign-on, the NoExecuteOTA cookie ensures that authentication actions are only triggered for designated policy domains, even when the ObSSOCookie contains the OTA parameter. This prevents the OTA parameter from working for targets with authentication actions that are not supposed to be triggered when the ObSSOCookie is present.

The NoExecuteOTA cookie set to true along with OTA set to true in a policy domain means that the authentication actions will not be performed for the resource protected by the policy domain.

The NoExecuteOTA cookie set to false along with OTA key set to true means that the authentication actions will be performed for the resource and the OBSSOCookie will be reset.

By default NoExecuteOTA is set to false.

Configuring the OTA Authentication Scheme and Authorization Action

The following procedures describe how to configure this scheme and actions associated with the scheme.

To create an OTA authentication scheme

1. From the Access System Console, click Access System Configuration.
2. Click Authentication Management in the left navigation pane.
3. Click the link for an authentication scheme for which you want to add the OTA:true parameter.

   The Details for Authentication Scheme page appears, with the General tab selected.
4. Click Modify.
5. In the Challenge Parameter field, add OTA:true.

   If necessary, click the plus symbol ("+") to add a new field for entering this parameter.
6. Click Save.

To create an associated authorization action

1. From the landing page for the Access System, select the Policy Manager link.

   If you are working in the Access System Console, click the link for the Policy Manager at the top of the page.
2. Click My Policy Domains in the left navigation pane.

   The General page for the selected policy domain appears.
3. Click the link for the policy domain for which you want to add the NoExecuteOTA cookie.
4. Click the Authorization Rules tab for this policy domain.
5. Provide a name for this authorization rule and click Save.
6. Click the Actions tab.
7. Click Add.
8. In the Return field, add the following:

   The type is cookie.

   The Name is NoExecuteOTA.

   The Return Value is true.

Auditing Authentication Events

An audit rule causes event-based data to be written to the audit log file. As a Master Access Administrator, you must create a Master Audit Rule in the Access System Console. As a Delegated Access Administrator, you can derive audit rules from the Master Audit Rule for your policy domains and policies, but you cannot create an alternative Master Audit Rule.

There is one audit log for an Access Server. You can configure the size of the audit log file and the rotation interval for a server. Depending on events, the audit log may contain some duplicate audit entries.

Note: You may direct audit details to a database, as described in the *Oracle Access Manager Identity and Common Administration Guide*.

Information Logged on Success or Failure

Different information is written to the audit log depending on the outcome of events. A log entry for authentication of a user differs depending on whether the user's identity was established.

Authentication failure can occur if there is no entry in the directory for a user or if a user's credentials are invalid. For example, if there is an entry for the user in the directory, but the user entered an incorrect password (authentication failure), the value for the cn attribute is logged based on the DN in the directory. However, because the entry for the user cannot be confirmed as the correct one, attributes such as givenname are not retrieved from the directory.

About Creating a Master Audit Rule and Derived Rules

You can define audit rules for a policy domain and its policies. Any audit rules you define must be derived from a Master Audit Rule. A Master Audit Rule must be created by a Master Access Administrator. Delegated Access Administrators can derive access rules from the Master Audit Rule, but they cannot create them.

Because you create audit rules for the policy domain and its policies, this chapter does not describe them. For details explaining how to create and define audit rules, see the following sections in the policy domain chapter:

- Auditing User Activity for a Policy Domain
- About Creating a Master Audit Rule and Derived Rules
- Creating an Audit Rule for a Policy Domain
- Creating an Authentication Rule for a Policy

6

Configuring User Authorization

The Access System enables you to protect resources, for example, Web pages and applications. These resources are defined as belonging to policy domains and individual policies. You use the Access System to define the resources, and to specify who is and is not authorized to use the resources, and under what conditions.

This chapter explains authorization and how to configure authorization rules and authorization expressions to meet the requirements for your policy domains and their policies. You combine authorization rules to create authorization expressions. A policy domain must include an authorization expression among the set of default rules that specify how its resources are protected.

This chapter discusses the following topics:

About Authorization

Authorization is the process of determining if a user may access a resource. To protect resources, you define authorization *rules*. These rules contain one or more *conditions*. You configure authorization *expressions* using one or more authorization rules. A policy domain and a policy can each contain only one authorization expression.

After you create your policy domains, you can define their rules and expressions. You can create the authentication rules, authorization rules and expressions, and audit rules for a policy domain in any order.

Background Reading

Before you read this chapter, read the following chapters:

- Chapter 3, "Configuring WebGates and Access Servers" on page 3-1 describes the configuration of AccessGates and Access Servers, which you must do before the policy domains you create can take effect.
- Chapter 4, "Protecting Resources with Policy Domains" on page 4-1 describes how to create and test policy domains and policies, how to define resource types, and how to define audit rules.
- Chapter 5, "Configuring User Authentication" on page 5-1 describes how to create and use authentication schemes and rules.

Introduction to Authorization Rules and Expressions

An authorization rule can contain the following:

- A condition that specifies who is authorized to access a protected resource.

  This condition is referred to as the Allow Access condition of the rule.

- A condition that specifies explicitly who is denied access to the protected resource.

  This condition is referred to as the Deny Access condition of the rule.

If Allow Access conditions, Deny Access conditions, or both are specified and they do not apply to a user, the user is not qualified by the rule, and by default the user is denied access to the requested resource.

To specify who is allowed or denied access to the resource, the rule can do the following:

- Identify the users by user name, role, or an LDAP filter whose criteria the user must satisfy.
- Stipulate the computers where users can access resources.
- Set a time period when the rule applies.

You can also set actions to take when the rule is evaluated to allow the user to access the resource, or if the rule is evaluated to deny access.

You protect resources in a policy domain by configuring an authorization expression containing one or more authorization rules.

Authorization expressions include the following:

- Authorization rules that you select from those that are defined for the policy domain.
- Operators that you use to combine rules to provide the kind of authorization protection that you want for the policy domain.

An authorization expression may consist of a single rule or a group of rules combined to express more complex conditions. For example, you can create an expression that requires a user to meet the Allow Access conditions of two rules to be granted access to the resource. You use the Policy Manager user interface to combine rules in expressions.

This chapter describes the Policy Manager authorization component, and it explains how it works. It also provides the procedures you use to protect your resources with authorization expressions.

The following is an overview of the steps you follow to create authorization expressions for your policy domains and their policies:

Task overview: Creating authorization expressions

1. Create your policy domain, as discussed in Chapter 4, "Protecting Resources with Policy Domains" on page 4-1.
2. Determine who is authorized to use the resources of the policy domain, and under what conditions, using the "Guidelines for Classifying Users" on page 6-3.

   See also "About Authorization Rules" on page 6-4.

   You can give specific users access to the resources and deny specific users access to the resources. It is not necessary for you to create rules that apply to all of your users—whether to allow them access or to expressly deny them access.

   Some users may not qualify for the conditions of a rule. They may qualify for other rules of the expression, or they may not qualify for the conditions of any rules. If a user does not qualify for the conditions of any of the rules of an expression, by default the user is denied access to the resource.
3. Create all of the authorization rules you need to protect the resources of the policy domain and any of its policies.

   See "Configuring Authorization Rules" on page 6-7 for details.

   You create all of these rules at the level of the policy domain. When you create a rule, you include an authorization scheme in it. If you do not plan to use the authorization scheme provided by the Access System, you must configure one or more custom ones. In this case, you must provide custom plug-ins. See "About Authorization Schemes for Custom Plug-Ins" on page 6-45 for details.
4. Create the authorization expression for the policy domain.

   A policy domain can have only one authorization expression. See "About Authorization Expressions" on page 6-14 for details.
5. Create an authorization expression for each of the policy domain's policies.

   See "About Authorization Expressions" on page 6-14 for details.

> **Note:** You must configure an authorization expression to determine if users are permitted to access resources. If no authorization expression is defined, access is denied to the target resources.

Guidelines for Classifying Users

Observe the following guidelines when classifying users:

- Divide the users and groups of users into groups for whom different conditions apply.

  For example, conditions can determine when the users can access the resources, the computers from which they must make their requests, and so on. See "About the Contents of an Authorization Rule" on page 6-6 for details.

  If some users fall into multiple categories, for example, a user in the marketing group belongs to the Teleon project group, or a user in the human resources group also belongs to the Teleon group, put the user in both categories. You can require that the user meet the conditions of two rules.

> **Note:** You do not need to be concerned about users who are denied access to the resources of the policy domain under any conditions. They are denied access by default if none of the rules of an expression qualify them.

- For each category for which you want to create a separate rule, consider the actions that you want to occur depending on whether the user is authorized to use the resource or if the user is not authorized to use it because of the rule.

  For example, you may want the system to return user profile information and pass that information to a downstream application, as follows:

  - If the user is authorized, you may want to pass the user's cn (common name) to another application so that the application can present a customized greeting to the user.
  - If the user is not authorized, you may also want to return information about the user for security purposes.

For information about actions, see "About Authorization Actions" on page 6-37.

Do this analysis for users for whom you want to grant authorization to use the policy domain's resources, and for users and groups for whom you want to explicitly deny authorization to use the resources.

To create policies for subsets of resources in a policy domain and protect them with different authorization rules, consider the same information for the policies: who can access the resources of the policy and under what condition, for whom, and under what conditions you want explicitly to deny access to the resources.

About Authorization Rules

An authorization rule identifies who can access a resource and who is explicitly denied access to the resource. You can include one or more authorization rules in an authorization expression for a policy domain or policy.

When a user requests access to a resource that is protected by an authorization rule in an authorization expression, information about the user is checked against the rule. If the rule stipulates other information, such as time period or time of day the rule applies, that, too, is checked. This process is referred to as *evaluation of the rule*.

The result of evaluation of an authorization rule, in conjunction with other authorization rules, if more than one is included in the authorization expression, determines if a user is granted access to the requested resource.

At the policy domain level, you create all of the authorization rules to be used for a policy domain or any of its policies. You combine these rules to create authorization expressions. For details about authorization expressions, see "About Authorization Expressions" on page 6-14.

This section describes authorization rules, and how to create and manage them. It includes the following topics:

About Allow Access and Deny Access Conditions

An authorization rule specifies the following two types of primary conditions:

- A condition referred to as Allow Access that grants the user access to the resource.
- A condition referred to as Deny Access that denies the user access to it.

When a user is said to qualify for an authorization rule, it does not mean that the user is authorized to use the resource protected by the rule. A user is said to qualify for a rule if the user meets a condition of the rule:

- If the user meets the Allow Access condition, the user qualifies for the Allow Access part of the rule.
- If the user meets the Deny Access condition, the user qualifies for the Deny Access part of the rule.
- If the user satisfies neither the Allow Access nor the Deny Access conditions, the rule is said to be unqualified for that user. You can also think of this as the user not qualifying for the rule. If evaluation of a rule results in an unqualified user, the user is denied access to the resource based on that rule.

For authorization expressions that contain multiple rules, a user may qualify for none of the expression's rules, one of the rules, or for the conditions of multiple rules. In any case, it is the result of evaluation of the expression—all of its rules and how they are combined—not any one rule, that determines whether a user is allowed or denied access to a resource.

Reuse of Authorization Rules

A policy domain can have only one authorization expression, which can include all of the authorization rules necessary to express the protection requirements for its resources. Each of the policies a policy domain contains can have its own authorization expression.

Any of the authorization rules you define for a policy domain can be used for the policy domain and for any of the policies it contains in the following ways:

- It can appear in multiple authorization expressions.
- It can appear in a single authorization expression more than once.

For information about authorization expressions, see "About Authorization Expressions" on page 6-14.

About the Contents of an Authorization Rule

An authorization rule contains the following information:

- **General Information**: An authorization rule has a name and a description, and it can be enabled or disabled. See "Configuring Authorization Rules" on page 6-7 for details.
- **Allow Access:** The Allow Access condition of an authorization rule specifies the end users and groups of users who are allowed access to a resource protected by the rule. See "Setting Allow Access" on page 6-9 for details.
- **Deny Access:** The Deny Access condition of an authorization rule specifies the end users and groups of users who are explicitly denied access to a resource protected by the rule. See "Setting Deny Access" on page 6-10 for details.
- **Timing Conditions:** An authorization rule can be configured to include a value that restricts access to a resource within a time period, such as 9:00 a.m. to 5:00 p.m. on week days for one group of users and 10:00 a.m. to 4:00 p.m. for another group of users. See "Setting Timing Conditions" on page 6-11for details.
- **Actions:** For either result of an authorization rule—whether its evaluation results in authorization success or authorization failure for a user requesting access to a protected resource—an associated set of actions can be specified to be taken in response to the result. For example, the Access System can return a header variable to be passed to a downstream application. The following list describes the kinds of actions you can specify:
  - Redirection of the user's browser to another URL.
  - Static values and user profile identity values passed in HTTP header variables or cookies.

  See "About Authorization Actions" on page 6-37 for information about actions.

About Authorization Rule Evaluation

When information about a user requesting access to a protected resource is checked against the conditions of an authorization rule, and the user qualifies for one of the conditions of the rule, that rule is evaluated to produce one of the following results:

- Authorization Success

  In this case, the user succeeds in gaining access to the requested resource. This result is associated with the Allow Access condition of the rule.
- Authorization Failure

  In this case, the user fails to gain access to the requested resource. This result is associated with the Deny Access condition of the rule.

Evaluation of a rule can produce neither result if the user requesting access to the protected resource is not mentioned in the Allow Access or the Deny Access conditions of the rule. In this case, the evaluation of the rule is said to be inconclusive, and the user is denied access to the requested resource.

Working with Authorization Rules

This discussion provides details about configuring and managing authorization rules:

- Displaying a List of Configured Authorization Rules
- Configuring Authorization Rules

- Setting Allow Access
- Setting Deny Access
- Setting Timing Conditions
- Viewing General Information About a Rule
- Modifying an Authorization Rule
- Deleting an Authorization Rule

Displaying a List of Configured Authorization Rules

You may find it useful to display a list of authorization rules before you define a new one.

To display a current list of authorization rules

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

   If you are working with the Access System Console, click the Policy Manager link at the top of the page.

2. Click My Policy Domains.

   The General page for the policy domain appears.

3. Select the policy domain that you want to view.

4. Select the Authorization Rules tab for the policy domain.

   The Authorization Rules page appears, as illustrated in the following screen shot, showing the list of authorization rules configured for the policy domain.

Configuring Authorization Rules

To configure an authorization rule, you define its general information, you set its Allow Access and Deny Access conditions, and you define actions for the rule, if any. This section describes how to configure general information for a rule.

You can specify general information about an authorization rule to identify the rule, to specify its authorization scheme, to enable or disable the rule, and so forth. Some of the information you can configure is optional.

You must specify an authorization scheme for every authorization rule you define. You can use the Authorization Scheme provided by the Access System or you can select a custom authorization scheme, if any are configured. See "About Authorization Schemes for Custom Plug-Ins" on page 6-45 for details.

You create all of the authorization rules to be used for a policy domain or any of its policies at the policy domain level.

To define an authorization rule

1. Launch the Access System, select the Policy Manager, and select My Policy Domains.
2. Select the policy domain that you want to see.
3. Select the Authorization Rules tab.

   A page appears listing existing authorization rules for the policy domain.

   Note: If you are creating a policy domain, you do not see any configured authorization rules.

4. Click Add.

   The General page for the authorization rule appears.
5. Specify a name for the authorization rule and, optionally, a brief description of it in the following text boxes:
   - **Name**: A name for this authorization rule.
   - **Description:** A brief description of this authorization rule.

     For example, for an authorization rule that includes a custom authorization scheme, you could explain the function the custom plug-in provides.
6. Select Yes from the Enabled list to enable the authorization rule or No to disable it.

   Select Yes if you want the authorization rule to be activated as soon as you click Save. Enabling an authorization rule makes it available for inclusion in an authorization expression. The rule is disabled by default.

   After an authorization rule is used in an authorization expression, you cannot disable it until it is removed from all of the expressions that use it.
7. For Allow takes precedence, select one of the following:
   - **Yes**: If you want the Allow Access condition to take precedence over the Deny Access condition.
   - **No:** If you want the Deny Access condition to take precedence over the Allow Access condition.

   If you configure Allow Access and Deny Access conditions for a rule, use this option to specify which condition of the rule should be honored if the user qualifies for both of a rule's conditions.
8. Determine when you want Access Server caches to be updated.
   - **Immediately:** Select Update Cache to update all Access Server caches immediately with information about this new prefix.
   - **Later**: If you do not select Update Cache, the Access Server caches are updated when they time out and read new information from the directory server.

9. Click Save.

   The General page appears displaying the information you specified.

10. Select the authorization scheme to include in the authorization rule.

    If the Master Access Administrator has not created custom authorization schemes, the only scheme available is the Oracle Authorization Scheme.

11. Click Add.

    The General page for an authorization rule appears.

Setting Allow Access

The Allow Access part of an authorization rule defines users and groups who are authorized to use the protected resource.

To set Allow access

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

   If you are working in the Access System Console, the link for the Policy Manager is at the top of the page.

2. Click My Policy Domains in the left navigation pane.
3. Select the policy domain that you want to see.
4. Select the Authorization Rules tab.

   A page appears listing the authorization rules for the policy domain.

   If you are creating a policy domain, you do not see any configured authorization rules.

5. Select the authorization rule whose Allow Access conditions you want to set.
6. Click the Allow Access tab.
7. Click Add if no conditions exist, click Modify if they exist.
8. Specify the users and groups who are allowed to access resources protected by this rule using the People, Role, Rule, and IP Address controls as indicated in the following list.

   Note: These options are alternatives. An end user or group specified in any of these fields is allowed access.

   a. **People:** Click Select User to select by user name
      - Use the Search facility to display configured users.
      - Click Add before the name of each user who is allowed to access resources protected by this rule.

   b. **Role:** Select No Role in the Role selection box to prevent users from being selected based on roles or select Anyone to allow anyone access to the protected resources.

   c. **Rule:** Enter an LDAP filter that specifies the users and groups who are allowed to access the protected resources using the plus and minus buttons to add new filters and delete existing ones.

d. **IP Address:** Enter the IP addresses of computers whose users are allowed access.

   Except where noted, the Access System supports the following conventions for IP addresses in Access System and Policy Manager:

   - An explicit address, such as 192.2.2.2.

   - An address with a wildcard, but the wildcard must be the last entry, such as 192.2.2.*, 192.2.*, or 192.*

   The Access System does not support the following items:

   - An address in which a wildcard is not the final entry. For example, 192.128.*.2 is not supported.

   - An entry of all wildcards, such as ***.*.*.*.

   If you entered an IP address using a format that is not supported, the error message "Invalid IP address entered" appears.

   For the IP Address fields, click the plus and minus buttons to add new IP addresses and delete existing ones.

9. Determine when you want Access Server caches to be updated.
   - **Immediately:** Select Update Cache to update all Access Server caches immediately with information about this new prefix.
   - **Later:** If you do not select Update Cache, the Access Server caches are updated when they time out and read new information from the directory server.
10. Click Save.

Setting Deny Access

The Deny Access part of an authorization rule specifies the users and groups who are denied the right to use the resources protected by this rule.

To set Deny Access

1. Launch the Access System and select the Policy Manager.
2. From the Policy Manager, select My Policy Domains, then click on the policy domain that you want to see.

   If you are in the process of defining the rule and have configured the rule's general information, you do not need to retrace this path.
3. Select the Authorization Rules tab.

   A page appears listing authorization rules for the policy domain.

   If you are creating a policy domain, you do not see any authorization rules.
4. Select the authorization rule for which you want to set Deny Access conditions.

   A set of panels for the authorization rule appears, with the General panel selected. Other panels for the rule are Timing Conditions, Allow Access, and Deny Access.
5. Click the Deny Access panel, then click Add if no authorization rules exist, or click Modify if they exist.
6. Specify the users and groups who are denied to access resources protected by this rule using the People, Role, Rule, and IP Address controls as indicated in the following list.

Note: These options are alternatives. An end user or group specified in any of these fields is denied access.

a. **People:** Click Select User to select by user name.
   - Use the Search facility to display configured users.
   - Click Add before the name of each user who is denied to access resources protected by this rule.

b. **Role:** Select No Role in the Role selection box to prevent users from being selected based on roles or select Anyone to deny anyone access to the protected resources.

c. **Rule:** Enter an LDAP filter that specifies the users and groups who are denied to access the protected resources using the plus and minus buttons to add new filters and delete existing ones.

d. **IP Address:** Enter the IP addresses of computers whose users are denied access.

   Except where noted, the Access System supports the following conventions for IP addresses in Access System and Policy Manager:

   - An explicit address, such as 192.2.2.2.

   - An address with a wildcard, but the wildcard must be the last entry, such as 192.2.2.*, 192.2.*, or 192.*.

   The Access System does not support the following items:

   - An address in which a wildcard is not the final entry. For example, 192.128.*.2 is not supported.

   - An entry of all wildcards, such as ***.*.*.*

   If you entered an IP address using a format that is not supported, the error message "Invalid IP address entered" appears.

   For the IP Address fields, click the plus and minus buttons to add new IP addresses and delete existing ones.

7. Determine when you want Access Server chaches to be updated.
   - **Immediately:** Select Update Cache to update all Access Server caches immediately with information about this new prefix.
   - **Later:** If you do not select Update Cache, the Access Server caches are updated when they time out and read new information from the directory server.

8. Click Save.

Setting Timing Conditions

Use the Timing Conditions option to set the time periods when the authorization rule is in effect. For example, you may want the rule to remain in effect only during business hours, Monday through Friday. If you do not set a timing condition, by default the authorization rule is always in effect. The rule's Allow Access and its Deny Access conditions remain in effect for the specified time period.

To set a timing condition

1. Launch the Access System, select the Policy Manager, and select My Policy Domains.
2. Select the policy domain that you want to see.
3. Select the Authorization Rules tab.

   A page appears listing the authorization rules for the policy domain.

   If you are creating a policy domain, you do not see any authorization rules.
4. Select the authorization rule for which you want to set timing conditions.

   The General panel for the authorization rule appears. Other panels for the authorization rule include Timing Conditions, Actions, Allow Access, and Deny Access.
5. Click the Timing Conditions panel.

   If any timing conditions exist, they are listed. Otherwise, this page reports that there are no timing conditions for the authorization rule.
6. Click Add to create a new timing condition if none have been defined, or click Modify if they exist.

   The Timing Conditions page appears.
7. Select either Greenwich Mean Time or Local time on Web server:

   - **Greenwich Mean Time:** A standard for universal time. If you use Greenwich Mean Time, this authorization rule is in force at the same time throughout the world.

     Use this option if you want this rule to be in force at the same time for your globally-dispersed workforce.
   - **Local time on Web server:** Indicates that users outside the server's time zone could be denied access.

     For example, if the server is located in New York, and the timing conditions do not allow access after 5 P.M., West Coast users would be denied access starting at 2:01 P.M.

     Note: If you want to restrict hours for users in various time zones, do not use this option. Instead, create a separate authorization rule that gives West Coast users access until 8 P.M. Eastern Time, and so forth.
8. Select a Start Date and End Date.

   Note: If you select the — option, for the Start Date, then this rule effectively does not have a Start Date. If you select the — option, for the End Date, then this Rule effectively does not have an End Date.
9. Select a Start Time and End Time:

   - You cannot choose only a Start Time or End Time. If you specify a Start Time, you must choose an End Time.

     By default, the Start Time and End Time fields are set to —, which means this rule does not have a Start Time and End Time. It is in effect 24 hours a day.

- When choosing a Start Time and End Time, you must make a selection for all three fields (hours, minutes, seconds). If you do not, the Start Time and End Time are invalid.

10. Select the Months of the Year, Days of the Month, and Days of the Week for which this rule is valid.

> **Note:** To select a single item (for example, a month) click to select it. To select more than one, hold down the Shift key as you select additional items in the same list. If you select the — option, this rule is in effect everyday.

11. Select Update Cache if you want all AccessGate and Access Servers caches to be updated *immediately* with information about these timing conditions.
12. Click Save.

Viewing General Information About a Rule

You may want to view general information about an authorization rule before you decide to modify the rule or use the rule in an authorization expression.

To view the general information for an authorization rule

1. Launch the Access System, select the Policy Manager, and select My Policy Domains.
2. Select the policy domain that you want to see.
3. Select the Authorization Rules tab.

   A page appears listing the authorization rules for the policy domain.
4. Select the authorization rule whose general information configuration you want to see.

   The General information panel for the authorization rule appears, showing the name, description, enabled status, and Allow Takes Precedence value for the rule. Other panels for the rule include Timing Conditions, Actions, Allow Access, and Deny Access.

Modifying an Authorization Rule

You can modify the authorization rules for a policy domain at any time. However, it is good practice to disable a rule before you modify it.

To modify an authorization rule

1. Launch the Access System, select the Policy Manager, and select My Policy Domains.
2. Select the policy domain that you want to see.
3. Select the Authorization Rules tab.

   A page appears listing the authorization rules for the policy domain.
4. Select the authorization rule that you want to modify.
5. Click Modify.

   The General page with editable text boxes appears.

6. Verify that the Enabled status box is blank to ensure the rule is disabled before modifying information.
7. Modify the general information as required, and any of the following:
   - **Timing Conditions:** Click the tab, and follow the instructions for defining them.
   - **Actions:** Click the Actions tab and follow the instructions for defining actions in "About Authorization Expressions" on page 6-14.
   - **Allow Access or Deny Access:** Click the appropriate link, and follow the instructions for defining the rules.
8. Click Save.

Deleting an Authorization Rule

You cannot delete an authorization rule that is used in an authorization expression for the policy domain or any of its policies.

To delete an authorization rule

1. Launch the Access System, select the Policy Manager, and select My Policy Domains.
2. Select the policy domain that you want to see.
3. Select the Authorization Rules tab.

   A page appears listing the authorization rules for the policy domain.
4. Select the check box for each rule that you want to delete.
5. Click Delete.

About Authorization Expressions

In some cases, a single authorization rule is all that is required to protect the resources of a policy domain or a policy. You can configure the rule to identify who is allowed access to the resources it protects, who is denied access to them, and under what conditions these controls apply, for example, when they apply and from which computer. An authorization rule does not need to cover all users in its Allow Access and Deny Access conditions. Users who request access to a resource that is protected by the rule but do not qualify for any of the conditions are, by default, denied access to the resource.

For other cases, it may be necessary to configure multiple authorization rules to protect resources. You can impose complex restrictions on different users. For example, you can define a policy that includes many authorization rules, a part of any one of which a user must meet to qualify for access to a protected resource (or to qualify for denial of access to it). You may also want the same policy domain to specify multiple conditions a user must meet. For example, you may require that the user meet two conditions—such as belonging to one group and using a computer assigned a specific IP address—to be granted access to the resource. To define the complete authorization conditions required for the resources you want to protect, you form an authorization expression. The Policy Manager provides an interface that makes it easy for you to form authorization expressions. You must create a default authorization expression for the policy domain, but you can also create an authorization expression for a policy within the domain.

This section describes authorization expressions, and how to create and manage them. It includes the following topics:

About the Contents of an Authorization Expression

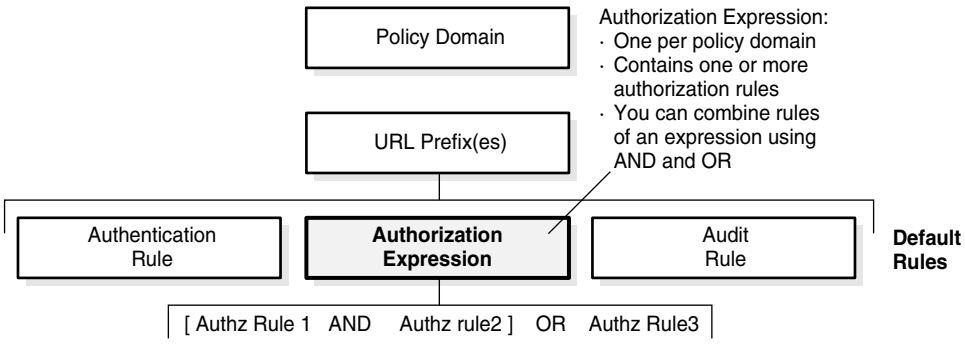
In an authorization expression, you define authorization requirements for a set of resources. An authorization expression can apply to resources for the entire policy domain or for one of its policies.

An authorization expression includes:

- One or more authorization rules
- The operators used to combine the rules

You can define only one authorization expression for a policy domain. This becomes the default authorization expression for the policy domain. You can define one authorization expression for each policy in the domain. An authorization expression can contain any of the authorization rules defined at the policy domain level. Figure 6–1 illustrates the default authorization expression for a policy domain. For the default protected URL, the authentication rule, authorization expression, and audit rule together form the defaults for the policy domain.

Figure 6–1 Authorization Expression



An authorization expression is always evaluated from left to right. The rules of an expression can be grouped using operators, and how they are grouped has a bearing on the outcome of the overall evaluation of the expression.

You can use two operators to combine the rules of an expression: AND and OR. You combine authorization rules to create authorization expressions that can include the following types of conditions:

- **A Compound Condition:** Specifies multiple conditions for which a user must qualify, either to be granted access to the requested resource or explicitly denied access to it, depending on the rest of the expression. You use the AND operator for this purpose. See "Authorization Rules Used in Example Scenarios" on page 6-18.
- **A Complex Condition:** Specifies two or more alternative conditions any of which a user must meet, either to be allowed access to the requested resource or denied access to it, depending on the condition and its relationship to the rest of the rules

of the expression. You use the OR operator for this purpose. See "Authorization Rules Used in Example Scenarios" on page 6-18.

See "About Evaluation of the Rules of an Expression" on page 6-16 for details explaining how grouping of the rules of an expression using AND and OR is interpreted.

About Authorization Expression Evaluation

Evaluation of an authorization expression can result in the following three conditions:

- **Authorization Success:** In this case, the user succeeds in gaining access to the requested resource. This result is associated with the Allow Access condition of the expression.
- **Authorization Failure:** In this case, the user fails to gain access to the requested resource. This result is associated with the Deny Access condition of the expression.
- **Authorization Inconclusive:** In this case, the rules of the expression produce conflicting results, and the user is denied access to the resource.

Status Codes for an Inconclusive Result

An expression can return a result of Inconclusive, in which case the Access System returns a major status code of Deny and a minor status code of Inconclusive.

The major status code of Deny is returned for Inconclusive results to maintain compatibility with previous releases of the system. The minor status code of Inconclusive is available to Oracle Access Manager systems to allow those systems to distinguish between true Deny results and Deny results returned because of an Inconclusive state.

An authorization expression result of Deny differs from an authorization expression result of Inconclusive even though the user is denied access to the resource in both cases. An application written to run with Oracle Access Manager can interpret the two status codes for an Inconclusive result and use the additional information for other purposes. For example, the application can then invoke other authorization engines instead of denying the user access to the resource.

About Evaluation of the Rules of an Expression

An authorization expression can contain a mix of compound conditions and complex conditions which determine whether a user can access a resource protected by the expression. When a user requests access to a protected resource, the user's information is checked against the rules of the expression.

The interplay between user information assessed against the rules of an expression, the position of the rules in the expression, and the way in which the rules are combined in the expression allows for a wide degree of variety. An authorization expression is exercised to different extents depending on these variables—that is, some of its rules might not be evaluated.

Precedence and Position: The Access Server processes the rules of an expression in the following way:

- **Precedence of Operators:** The AND operator takes precedence over the OR operator with respect to how rules of an expression are combined.

  That is, if an expression contains three or more rules combined in some way with the AND operator and the OR operator, the Access Server always associates the

rules on either side of the AND operator with it first, and then it combines the rules using the OR operator.

For example, given the following authorization expression,

```
R1 OR R2 AND R3
```

internally the Access Server creates the following grouping by default:

```
R1 OR (R2 AND R3)
```

The Access Server goes through the entire expression making these groupings based on AND taking precedence over OR before it evaluates the user's information against the rules.

For details about operators, see "Authorization Rules Used in Example Scenarios" on page 6-18.

Note: You can override the default way in which operators are interpreted by using parenthesis to enforce new groupings. See "About the Use of Parenthesis" on page 6-26 for details.

- **Position of Rules in an Expression:** The Access Server evaluates an expression from left to right.

  You do not assign to an authorization rule its priority among other rules. It would not be possible to reuse authorization rules if you assigned to each of them an evaluation priority. Rather, you position rules in an expression from left to right—which is the order in which they are evaluated—and you use operators to combine them. For details about operators, see "Authorization Rules Used in Example Scenarios" on page 6-18.

- **Use of Parenthesis to Override Default Precedence:** You can use parenthesis to override the default way in which the Access Server groups the rules of an expression. The Access Server continues to evaluate the rules of an expression from left to right, but it assesses the rules within the couplings and groups you create through use of parenthesis. See "About the Use of Parenthesis" on page 6-26.

About the Definitive Result of an Authorization Expression: The Access Server evaluates the rules of an expression until it can produce a definitive result. Evaluation of an authorization expression may produce a definitive Allow Access result, a Deny Access result, or an Inconclusive result.

For example, a user qualifies for the Allow Access condition of Rule 1, the Deny Access condition of Rule 2, and the Deny Access condition of Rule 3 of the following expression.

```
(Rule 1 AND Rule 2) OR Rule 3
```

In this case, evaluation of Rule 3 produces a definitive result of the expression, and the user is denied access to the resource. Neither Rule 1 nor Rule 2 has any bearing on the outcome of the expression because they produce conflicting results as part of an AND condition. Because Rule 3 is part of an OR condition, it stands on its own. If the user satisfies the rule's Allow Access or Deny Access condition, then Rule 3 defines the outcome of the expression.

For Rule 2 to be responsible for the definitive result, the user must qualify for either both the Allow Access conditions or both the Deny Access conditions of Rule 1 and Rule 2. In this case, Rule 3 would not be evaluated because evaluation of Rule 1 and

Rule 2 would produce a definitive result. Therefore, evaluation of Rule 3 would be unnecessary.

Authorization Rules Used in Example Scenarios

Table 6–1 contains examples of authorization rules that, if defined at the policy domain level, could be used in authorization expressions for the domain and any of its policies. The example authorization rules in Table 6–1 show only one condition of a rule—either its Allow Access condition or its Deny Access condition—not the full authorization rule.

An authorization rule need not specify both an Allow Access condition and a Deny Access condition, or either one alone. It can specify either condition, both conditions, or none. Table 6–1 identifies example authorization rules which are used in example scenarios throughout the rest of this chapter.

Table 6–1 Example Authorization Rules and Their Conditions

Authorization Rule	Condition
Rule 1	Allow anyone from the marketing department group access to the requested resource.
Rule 2	Allow anyone using a computer with the IP address 192.168.2.123 access to the requested resource.
Rule 3	Allow anyone from the human resources department group access to the requested resource.
Rule 4	Allow anyone from the Teleon project group access to the requested resource.
Rule 5	Deny anyone from the consultants group access to the requested resource.
Rule 6	Deny anyone from the Saber project group access to the requested resource.
Rule 7	Deny anyone using a computer with the IP address 192.168.5.123 access to the requested resource.
Rule 8	Allow anyone from the managers group access to the protected resource.
Rule 9	Allow anyone from the administrative assistants group access to the protected resource.

About the AND Operator

You use the AND operator to form a compound condition which combines authorization rules. Any number of rules can be combined using the AND operator to implement the full scope of conditions a user must meet to satisfy the authorization requirement. However, a user must satisfy the same kind of condition—either Allow Access or Deny Access—of all of the rules of the AND compound condition for the AND clause to produce a definitive result.

An authorization expression can contain more than one coupling or grouping of rules combined using AND. For example, it may contain several AND clauses, one connected to another by an OR operator.

Note: A user may qualify for both the Allow Access condition and the Deny Access condition of the same rule. In this case, whichever condition is configured to take precedence is the one that is honored. You configure this setting in the Allow takes precedence field.

Examples of Compound Conditions

The following scenarios use the example authorization rules in Table 6–1 to illustrate compound conditions.

For some of these examples, the Policy Manager Authorization Expressions page you use to create the expression is shown. Here is where to find information explaining how to use these pages to create authorization expressions:

- For the steps to follow to create an authorization expression, see "Creating Authorization Expressions" on page 6-29.
- For information explaining how to use the Authorization Expression interface portion of the Policy Manager to create expressions, see "Modifying an Authorization Expression as You Create It" on page 6-32. These instructions apply both to creating an expression and modifying an existing one.

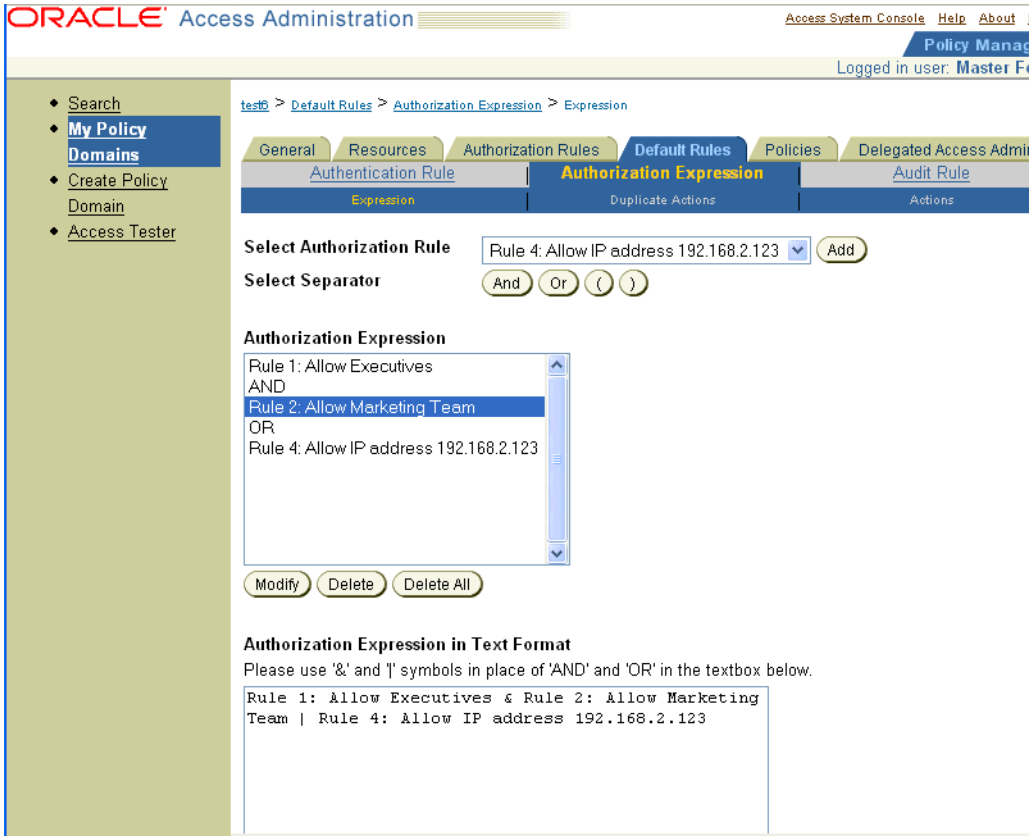
A Compound Condition Whose Two Authorization Rules Specify Allow Access Conditions: To be allowed access to a resource protected by the following authorization expression, a user must belong to the marketing department group and the IP address of the user's computer must be 192.168.2.123.

```
Rule 1 AND Rule 2
```

A Compound Condition Whose Three Authorization Rules Specify Allow Access Conditions: To be allowed access to a resource protected by the following authorization expression, a user must belong to the marketing department and the executive team, or must be accessing the resource from IP address 192.168.2.123.

```
Rule 1 AND Rule 2 AND Rule 4
```

Here is what the expression would look like if you configured it using the Expression panel of the Authorization Expression sub-tab.



A Compound Condition Whose Two Authorization Rules Specify Deny Access Conditions: To be explicitly denied access to a resource protected by the following authorization expression, a user must belong to the Consultants group and belong to the Saber project group.

```
Rule 5 AND Rule 6
```

About the OR Operator

An authorization expression can include a complex condition containing two or more alternative authorization rules. Authorization rules forming a complex condition are combined using the OR operator. Each of the authorization rules specified by a complex OR condition stands on its own. Unlike compound conditions using the AND operator, the user need qualify for the condition of only one of the authorization rules connected by OR operators.

An authorization expression can contain as many authorization rules connected using the OR operator as are required to express the authorization policy for the resources it protects. You can use the OR operator to connect authorization rules all of which have Deny Access conditions, all of which have Allow Access conditions, or which specify a mix of Deny Access and Allow Access conditions. You can connect single rules to single rules using OR, and you can connect a single rule to a clause containing rules combined using AND.

Examples of Complex Conditions

The following scenarios use the example authorization rules in Table 6–1.

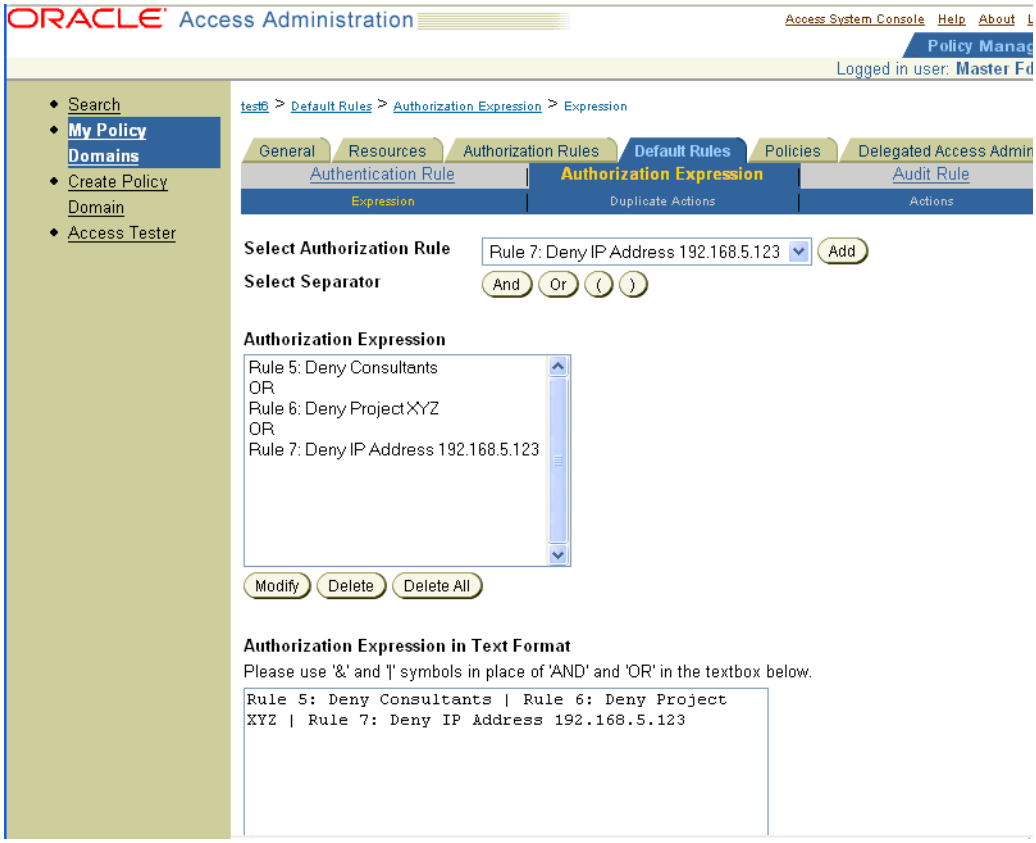
A Complex Condition Whose Two Rules Specify Allow Access Conditions: To be allowed access to a requested resource protected by the following rule, a user must either be a member of the marketing department group or the human resources department group.

```
Rule 1 OR Rule 3
```

Complex Condition Whose Three Authorization Rules Specify Deny Access Conditions: To be explicitly denied access to a requested resource, a user must belong to the Consultants group, or belong to the project XYZ group, or use a computer with the IP address 192.168.5.123.

```
Rule 5 OR Rule 6 OR Rule 7
```

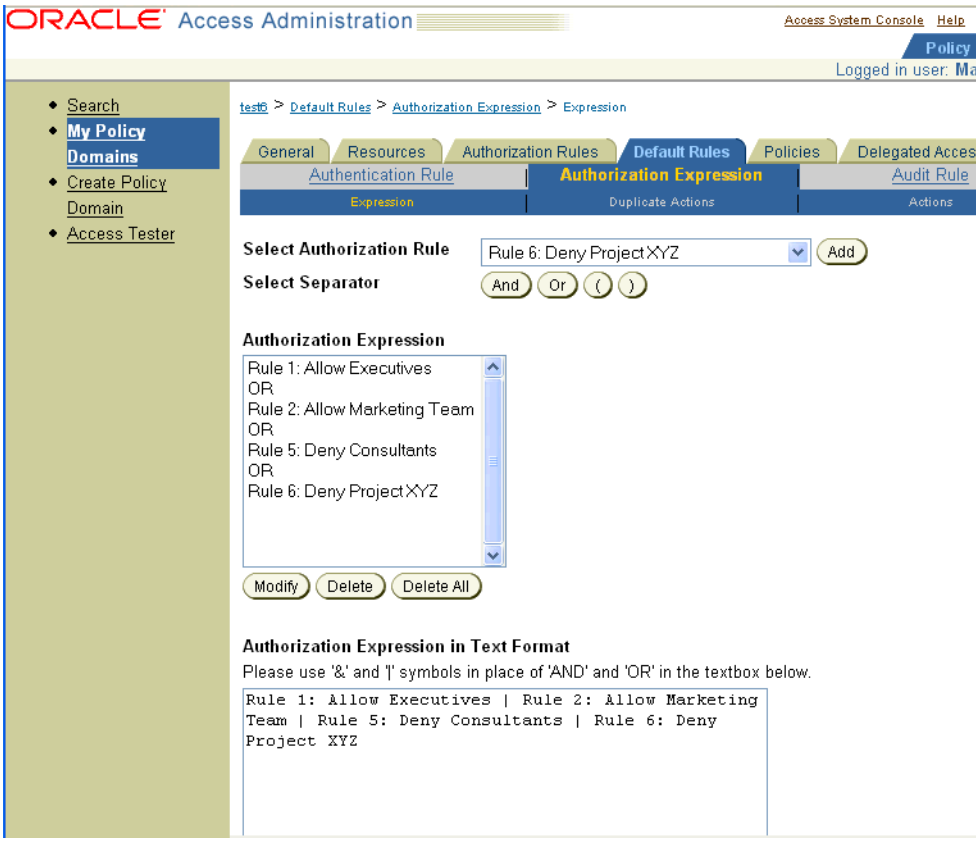
Here is what the expression would look like if you configured it using the Authorization Expression's Expression page.



A Complex Condition with Rules that Specify a Mix of Allow Access and Deny Access Conditions: To be allowed access to a requested resource protected by the following expression, a user must either be a member of the marketing department group or the executive team. To be explicitly denied access to a requested resource, a user must belong to the Consultants group or be a member of the XYZ project group.

```
Rule 1 OR Rule 2 OR Rule 5 OR Rule 6
```

Here is what the expression would look like if you configured it using the Expression panel on the Authorization Expression tab.



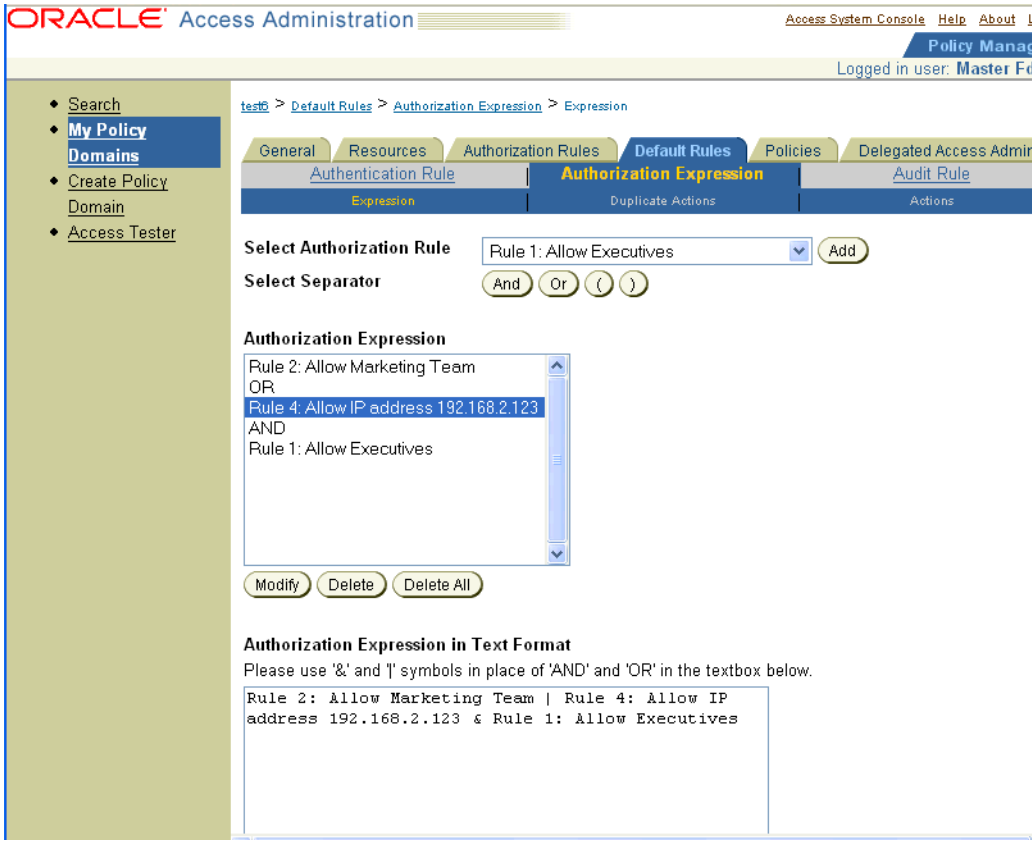
Compound Complex Expression Scenarios

The following scenarios use the example authorization rules in Table 6–1 to illustrate authorization expressions that contain both compound and complex expressions.

A Complex Condition Authorization Expression with Three Rules: A Delegated Access Administrator forms the following expression:

```
Rule 2 OR Rule 4 AND Rule 1
```

Here is what the expression would look like if you configured it using the Authorization Expression's Expression page.



Jane requests access to a resource protected by this authorization expression. The Access Server evaluates the expression to determine if Jane meets either of the following conditions that would allow her access to the resource:

- The IP address of Jane's computer is 192.168.2.123 and she belongs to the executive group (Rule 4 AND Rule 1)
- Jane is a member of the marking department group (Rule 2)

If parenthesis were used to make explicit the grouping of rules according to how the Access Server evaluates the authorization expression, the expression would look like this:

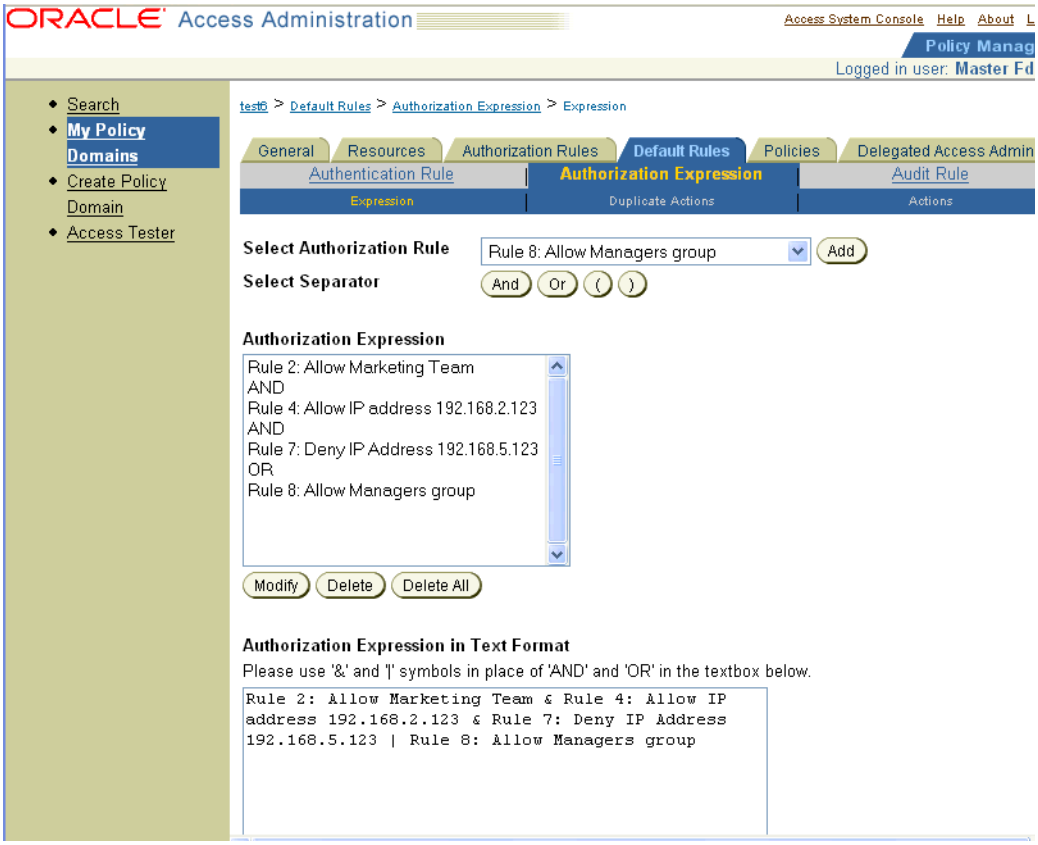
```
Rule 2 OR (Rule 4 AND Rule 1)
```

An expression is evaluated from left to right until a definitive result is produced. Jane meets the condition of Rule 1, which is followed by the OR operator, so she is granted access to the resource.

A Complex Condition Expression with Four Rules: A Delegated Access Administrator forms the following expression:

```
(Rule 2 AND Rule 4) AND (Rule 7 OR Rule 8)
```

Here is what the expression would look like if you created it using the Authorization Expression's Expression page.



Maurice is allowed access to a resource protected by this authorization expression because he satisfies the following conditions:

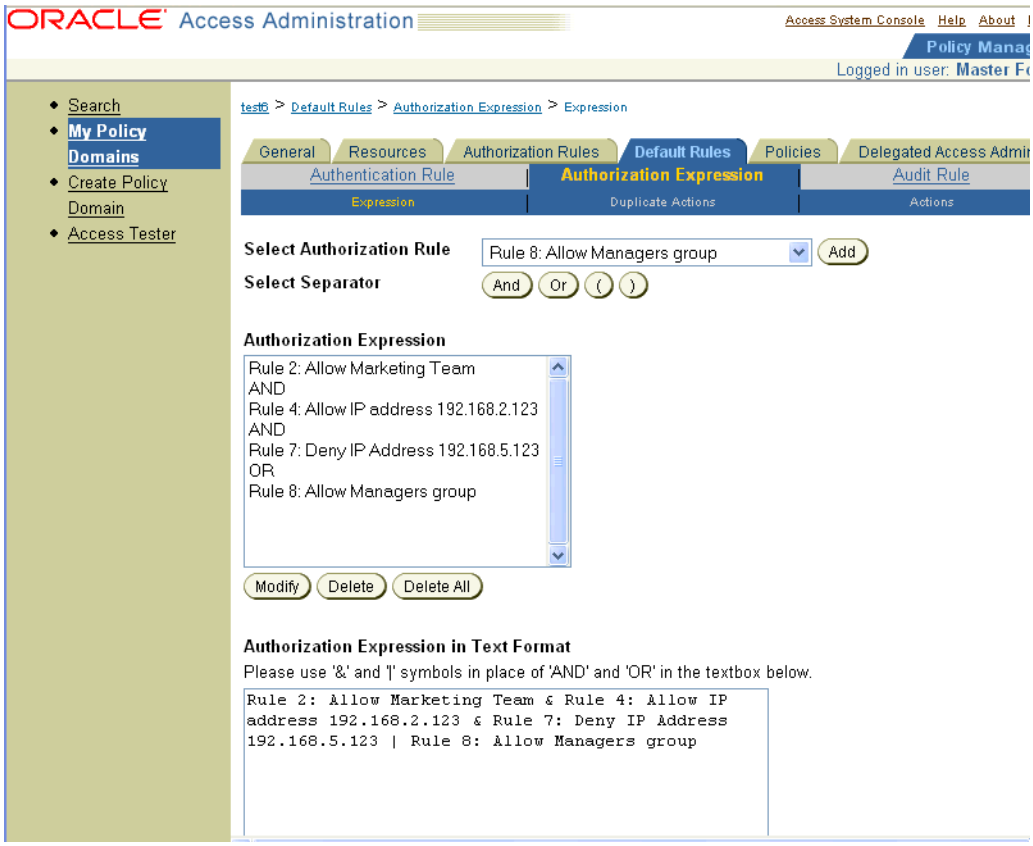
- He is a member of the marketing department and the IP address of his computer is 192.168.2.123. (Rule 2 AND Rule 4)
- He is also a manager and belongs to the Managers group. (Rule 8)

  The IP address of Maurice's computer is not 192.168.5.123 (Rule 7). However, he is not denied access for this reason because the authorization expression dictates that he meet either Rule 7 or Rule 8, but not both.

A Complex Condition Expression with Six Rules: A Delegated Access Administrator forms the following expression:

```
Rule 2 OR Rule 4 OR Rule 1 AND Rule 9 OR Rule 5 AND Rule 6
```

Here is what the expression would look like if you used the Authorization Expression's Expression page to configure it. Notice that the Authorization Expression List box does not show the last rule. To see the last rule, you would have to scroll down. However, the Text Format box wraps the text to show the complete expression.



If parenthesis were used to make explicit the grouping of rules, the expression would look like this:

```
Rule 2 OR Rule 4 OR (Rule 1 AND Rule 9) OR (Rule 5 AND Rule 6)
```

Following the order of precedence of AND over OR with respect to how rules are grouped and left-to-right processing of the rules, a user must qualify for one of the following conditions to gain access to the requested resource:

- The first single rule of the complex condition (Rule 2)

  A user who belongs to the marketing department group is allowed access to the resource.

- The second single rule of the complex condition (Rule 4)

  A user whose computer has the IP address 192.168.2.123 is allowed access to the resource.

- The first compound condition (Rule 1 AND Rule 9)

  A user who belongs to the executive team and who belongs to the ABC project group is allowed access.

- The second compound condition (Rule 5 AND Rule 6)

  A user who belongs to the Consultants group and the XYZ project group is denied access to the resource.

In its evaluation, the Access Server progresses through the expression until it evaluates a rule that produces the definitive result of the expression. If the Access Server completes evaluation of the expression and the user does not qualify for any of its

conditions, the result of the evaluation is Inconclusive. In such a case, because no rules apply to the user, no actions associated with rules are taken. However, the actions configured for the Inconclusive result of the expression are taken. For information about actions, see "About Authorization Actions" on page 6-37. For information about status codes returned for inconclusive results, see "Status Codes for an Inconclusive Result" on page 6-16.

About the Use of Parenthesis

By default, two rules on either side of an AND operator compose the compound AND condition. Rules on either side of an OR operator are alternatives. When no parenthesis are used to enforce grouping of rules, the AND operator takes precedence over the OR operator.

For example, if no parenthesis were used in the following expression to override the default way in which the rules of the following expression would be evaluated:

```
R1 OR R2 AND R3 OR R4 AND R5
```

the expression would be interpreted in the following way:

```
R1 OR (R2 AND R3) OR (R4 AND R5)
```

You can use parenthesis to override the normal grouping of the rules of an expression, for example, to give precedence to the OR condition over the AND condition.

The following example uses the same expression. In this instance of the expression, parenthesis are used to override the default grouping:

```
(R1 OR R2) AND (R3 OR R4) AND R5
```

About Large Authorization Expressions

Some directory servers impose an attribute value limit. For instance, Oracle Internet Directory imposes a limit for attribute values such that they cannot exceed 4000 characters. If your authorization expressions exceed 4000 characters an error message occurs when saving the expression:

```
No Authorization Defined
```

Before you add a large authorization expression, the Policy Manager globalparams.xml file requires the addition of the `policyDSMaxAttrValueLength` parameter. The value that you add should be the maximum limit imposed on an attribute length by the directory server. For Oracle Internet Directory, this value should not exceed 4000 characters. For other directory servers, consult the vendor-specific documentation for further details.

If you are adding the policy using an API in the Access Manager SDK, then you need to add the `policyDSMaxAttrValueLength` parameter to the associated Access Server globalparams.xml file. By default, this feature is disabled (there is no such parameter in the file by default). You must add the parameter manually.

Note: If the directory limit cannot be changed or cannot accommodate the size of the expression, the `policyDSMaxAttrValueLength` parameter will instruct Oracle Access Manager to compensate for the limit and allow creation of larger expressions.

For more information and details to perform this task, see "Problem with Large Authorization Expressions" on page E-4.

Working with Authorization Expressions

The following discussions provide procedures for working with authorization expressions:

- Viewing Authorization Expressions
- Creating Authorization Expressions
- Modifying an Authorization Expression as You Create It
- Deleting an Authorization Expression

Viewing Authorization Expressions

A policy domain can have only one authorization expression. Each of its policies can also have an authorization expression. If an expression has already been defined for either, you can look at its definition at any time.

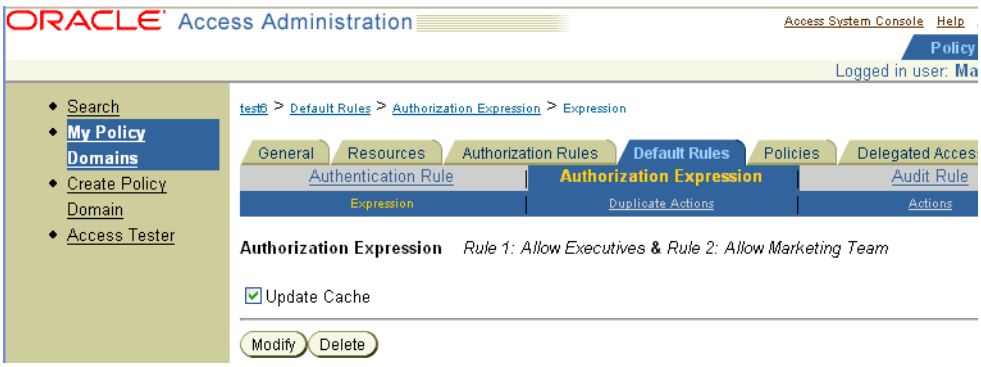
If an authorization expression exists for the policy domain or for a policy, the Expression page displays the entire authorization expression. If the authorization expression is long, the text is wrapped onto the next line, and so on, to display all of the expression.

An authorization expression includes the content of the expression—its rules and operators—and the configuration for the expression itself.

To view an authorization expression for a policy domain

1. Launch the Access System, select the Policy Manager, and select My Policy Domains.
2. Select the policy domain whose authorization expression you want to see.
3. Select Default Rules.
4. Select the Authorization Expression tab.

   The Authorization Expression page appears, as illustrated in the following screen shot. This page shows the name of the expression and its value configured for the policy domain. It also has three panels: Expression, Duplicate Actions, and Actions.



To look at values configured for the expression:

- Click Duplicate Actions.

If a duplicate actions policy has been configured for the authorization expression, this section defines how duplicate actions are handled for the policy domain protected by the authorization expression. A policy domain can include one or more policies.

See "About Duplicate Actions" on page 6-43 for details.

- Click Actions.

  This section defines the actions configured for this authorization expression.

5. Click Modify to look at the content of the expression.

   The configuration for an expression appears on the page used to create the expression or modify it.

   To see the actions configured for each rule of an expression, you must check the rule's configuration. See "About Authorization Rules" on page 6-4.

Viewing the Authorization Expression for a Policy

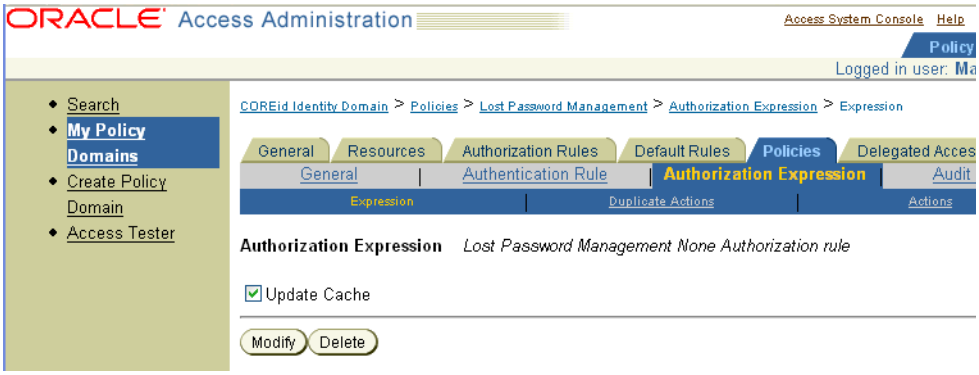
Each policy has its own authorization expression. You can view it from within the definition of the policy.

An authorization expression includes the content of the expression—its rules and operators—and the configuration for the expression itself.

To view an authorization expression for a policy

1. Launch the Access System, select the Policy Manager, and select My Policy Domains.
2. Select the policy domain containing the policy whose authorization expression you want to see.
3. Select Policies.
4. Select the policy whose authorization expression you want to see.
5. Click Authorization Expression.

   The Authorization Expression page appears, as illustrated in the following screen shot. This page shows the name of the expression, Lost Password Management None Authorization rule.



6. Click Modify to look at the content of the expression.

   The configuration for the expression appears on the page used to create it or modify it.

> **Note:** To see the actions configured for each rule of an expression, you must check the rule's configuration. See "About Authorization Rules" on page 6-4.

7. **Optional:** Look at values configured for the expression:
   - Click Duplicate Actions to display the section that defines how duplicate actions are handled for the resources protected by this policy. The setting for the policies authorization expression Duplicate Actions overrides that of the policy domain. See "About Duplicate Actions" on page 6-43 for details.

     The authorization expression for a policy may contain its own duplicate actions setting. In this case, the policy domain's duplicate actions setting overrides the one set for the policy domain.
   - Click Actions to display the section that defines the actions configured for this authorization expression.

Creating Authorization Expressions

The authorization expression for a policy domain applies to all resources of the domain unless those resources are protected by a policy containing an expression.

To create an authorization expression for a policy domain

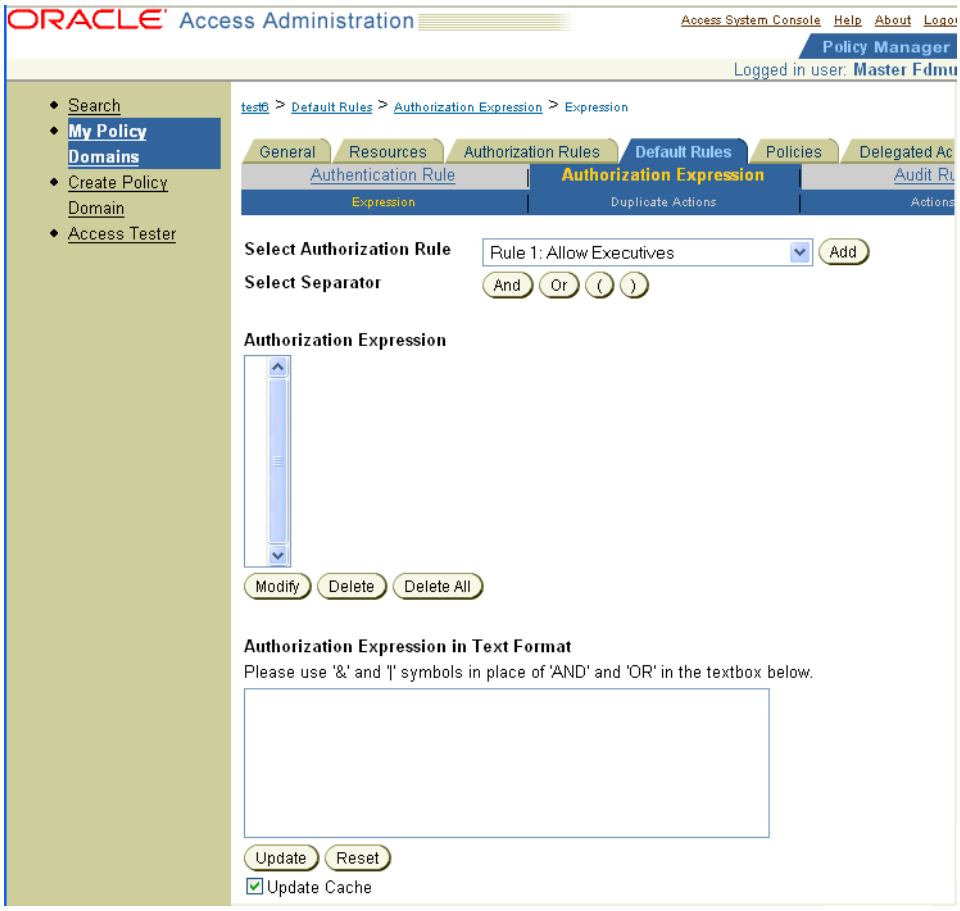
1. Launch the Access System, select the Policy Manager, and select My Policy Domains.
2. Select the policy domain for which you want to create an authorization expression.
3. Select Default Rules.
4. Select the Authorization Expression tab.

   The Authorization Expression page appears. If there is no defined authorization expression, a message appears, "There is no Authorization Expression defined."

   > **Note:** If an authorization expression exists, you can only modify its content. To replace it, you must modify all parts of it.

5. Click Add.

   The Authorization Expression page you use to create the expression appears, as illustrated in the following screen shot.



You use the Authorization Expression page to create an authorization expression.

6. Using the following steps, select the authorization rules for the authorization expression and the operators you want to use to combine those rules.

   Note: If you want to include the first rule in a parenthetical phrase, click the open parenthesis button before you add the first rule to the expression.

   a. From the Select Authorization Rule list, select the first rule to be added to the expression, and click Add.

   b. If the authorization expression includes multiple rules, select the operator to be used to combine the first two rules.

      – For the AND operator, click the And button beside Select Separator.

      – For the OR operator, click the Or button beside Select Separator.

      – To begin a parenthetical phrase, click the open parenthesis button.

      – To close a parenthetical phrase, click the close parenthesis button.

7. Continue to add rules to the authorization expression, and combine them with other rules until you have completed forming the expression to fit your authorization requirements.

8. Click Save to save the expression.

9. Select the Duplicate Actions tab in the Authorization Expression page.

   The Duplicate Action Headers page appears, as illustrated in the following screen shot.

10. Click Modify to select the duplicate actions policy. The Duplicate Actions page appears, as illustrated in the following screen shot.

11. Click Select the checkbox and the radio button for the type of Duplicate Actions handling you want.

    The duplicate actions policy you set at the authorization expression level overrides that set at the Access System Console level.

12. Determine when you want Access Server chaches to be updated.

    - **Immediately:** Select Update Cache to update all Access Server caches immediately with information about this new prefix.
    - **Later:** If you do not select Update Cache, the Access Server caches are updated when they time out and read new information from the directory server.

    You cannot save an authorization expression that contains syntax errors. When you click Save, the Access Server checks the authorization expression to ensure that it is well-formed. If an authorization expression contains a syntax error—for example, an error occurs if you include an AND or OR operator at the end of the expression—you must correct the error and then save the expression.

13. Click Save.

    After you save the authorization expression, the Authorization Expression view page appears showing the full expression. For details explaining how to use the features of the Authorization Expression page to create an expression, see "Modifying an Authorization Expression as You Create It" on page 6-32.

Creating an Authorization Expression for a Policy

The steps you use to create an authorization expression for a policy are similar to those for a policy. For details, see "Creating Authorization Expressions" on page 6-29. If you are already in a policy domain, you can skip steps 1 and 2 and start with Step 3.

To create an authorization expression for a policy

1. Launch the Access System, select the Policy Manager, and select My Policy Domains.
2. Select the policy domain containing the policy for which you want to create an authorization expression.
3. Select the Policies page.
4. Select the name of the policy for which you want to create an authorization expression.
5. Select the Authorization Expression tab.

   The Authorization Expression page appears displaying the message "No authorization expression is defined for this policy."
6. Click Add.
7. The Authorization Expression with an active list box, text entry box, and scrolling lists appears.

Modifying an Authorization Expression as You Create It

As you create an authorization expression, you may want to change the way you have combined the rules of the expression. You may change the form of an expression as you create it, for example, to express a different authorization policy or to correct errors.

If the authorization expression contains many components—rules and operators—a scroll bar is displayed at the right side of the authorization expression list box so that you can scroll to bring items into view.

You can modify an authorization expression in either of the following two ways:

- You can modify an authorization expression within the Authorization Expression list box.
- You can modify an authorization expression within the Authorization Expression in Text Format box.

Changes you make to an authorization expression in one box are reflected in the other box in the following ways:

- As you form the authorization expression by adding rules and operators to the Authorization Expression list box, the Authorization Expression in Text Format box is automatically updated to reflect the additions and modifications.

- After you make changes to an expression in the Authorization Expression in Text Format box, you must click the Update button for those changes to be reflected in the Authorization Expression list box.

The way some operators are expressed in the Authorization Expression list box differs from how they are expressed in the Authorization Expression in Text Format box. The following table shows the differences.

Table 6–2 Operators for List Box and Text Format Box

Operator in List Box	Operator in Text Format Box
AND	&
OR	|
(	(
)	)

You use buttons to enter operators in the Authorization Expression List box. You use keys to enter operators in the Authorization Expression in Text Format text box.

Using the Authorization Expression List Box

The Authorization Expression list box displays the authorization rules and the operators that you use to combine them as you select and add rules and operators to form the expression.

> **Note:** As you create an authorization expression using the Authorization Expression list box, the expression content is reflected in the Authorization Expression in Text Format editable text box.

To manipulate the content of an expression in the Authorization Expression list box, you use the following buttons:

- **Modify:** Replaces one rule of an authorization expression with another rule selected from the Select Authorization Rule list.

  To replace one operator with another, you swap the two operators directly by selecting one operator and clicking the button for the replacement operator.
- **Delete:** Deletes any selected item—a rule, an operator, or an open or close parenthesis—from the Select Authorization Rule list.
- **Delete All:** Clears the entire content of the authorization expression.

To replace one authorization rule with another

1. Launch the Access System, select the Policy Manager, and select My Policy Domains.
2. Select the policy domain whose authorization expression you want to modify.
3. Select Default Rules.
4. Select the Authorization Expression tab.

   The Authorization Expression view page appears showing the existing authorization expression.
5. Click Modify.

The Authorization Expression with an active selection list, text entry box, and scrolling list box appears.

6. Select the rule to be replaced in the Authorization Expression list.
7. Select the replacement rule in the Select Authorization Rule list.
8. Click the Modify button.

   The old rule in the Authorization Expression list is replaced by the new rule.

To replace one operator with another

1. Launch the Access System, select the Policy Manager, and select My Policy Domains.
2. Select the policy domain whose authorization expression you want to modify.
3. Select Default Rules.
4. Select the Authorization Expression tab.

   The Authorization Expression view page appears showing the existing authorization expression.
5. Click Modify.

   The Authorization Expression page with an active selection list, text entry box, and scrolling list box appears.
6. Select the operator to be replaced in the Authorization Expression list.
7. Click the button for the replacement operator.
   - To replace the OR operator with the AND operator, select OR in the expression, and click the And button.
   - To replace the AND operator in the expression, select it and click the Or button.

   The old operator is replaced by the new one in the Authorization Expression list.

To delete an item

1. Navigate to the Authorization Expression list.
2. Select the item to be deleted in the Authorization Expression list.
3. Click the Delete button.

To delete the entire content of an expression

1. Navigate to the Authorization Expression list.
2. Click the Delete All button.

Using the Authorization Expression in Text Format Box

As you form the authorization expression by adding rules and operators to the Authorization Expression list, the Authorization Expression in Text box is updated to reflect the additions and modifications.

You can modify the textual content of an authorization expression directly using the Authorization Expression in Text box.

Entering New Text: To modify the text, you use keyboard or keypad keys and symbols to enter new text or to overtype existing text. (In addition to typing the text, the main

difference is that you enter symbols to represent operators.) See Table 6–2 on page 6-33 for the symbols to use for operators.

Deleting Text: To delete text from the authorization expression, you use any of the standard approaches you take to handle text in a flat text file.

Updating the Authorization Expression List: To update the list with the changes you made in the Authorization Expression in Text Format text box, click the Update button directly beneath the text box.

Modifying an Existing Authorization Expression

If an authorization expression exists for the policy domain or for a policy, the Authorization Expression view page displays the entire expression. If the authorization expression is long, the text is wrapped onto the next line, and so on, to display all of the expression.

You can modify an authorization expression after it has been used to protect the policy domain or the policy for which it was created.

When modifying an authorization expression, you follow the same procedures you use to create an expression. This section describes how to navigate to the Authorization Expression page you use to modify an expression for a policy domain and for a policy.

To display the page for modifying the authorization expression for a policy domain

1. Launch the Access System, select the Policy Manager, and select My Policy Domains.
2. Select the policy domain for which you want to create an authorization expression.
3. Select Default Rules.
4. Select the Authorization Expression tab.

   The Authorization Expression view page appears showing the existing authorization expression.
5. Click Modify.

   The Authorization Expression page with an active selection list and two text entry boxes appears.

For the remainder of this process, see the steps of the following procedures for creating and modifying an authorization expression:

- "Creating an Authorization Expression for a Policy" on page 6-32
- "Modifying an Authorization Expression as You Create It" on page 6-32

To display the Authorization Expression page for a policy to modify the expression

1. Launch the Access System, select the Policy Manager, and select My Policy Domains.
2. Select the policy domain containing the policy for which you want to create an authorization expression.
3. Select the Policies tab.
4. Select the name of the policy whose authorization expression you want to modify.

5. Select the Authorization Expression tab.

   The Authorization Expression view page appears showing the existing authorization expression.

6. Click Modify.

   The Authorization Expression with an active list box, text entry box, and scrolling list box appears.

For the remainder of this process, see the steps of the following procedures for creating and modifying an authorization expression:

- "Creating an Authorization Expression for a Policy" on page 6-32
- "Modifying an Authorization Expression as You Create It" on page 6-32

Deleting an Authorization Expression

Before you can create a new authorization expression for a policy domain or for one of its policies, you must delete the existing one.

To delete the authorization expression for a policy domain

1. Launch the Access System, select the Policy Manager, and select My Policy Domains.
2. Select the policy domain whose authorization expression you want to delete.
3. Select Default Rules.
4. Select the Authorization Expression tab.

   The Authorization Expression view page appears showing the existing authorization expression.

5. Click Modify.

   The Authorization Expression edit page appears showing the content of the existing authorization expression.

6. Click the Delete All button beneath the Authorization Expression text box.

To delete the authorization expression for a policy

1. Launch the Access System, select the Policy Manager, and select My Policy Domains.
2. Select the policy domain containing the policy whose authorization expression you want to delete.
3. Select the Policies tab.
4. Select the name of the policy whose authorization expression you want to delete.
5. Select Default Rules.
6. Select the Authorization Expression tab.

   The Authorization Expression view page appears showing the existing authorization expression.

7. Click Modify.

   The Authorization Expression edit page appears showing the content of the existing authorization expression.

8. Click the Delete All button beneath the Authorization Expression list box.

About Authorization Actions

For every authorization rule, you can configure both a set of actions to be taken if a user is granted access to the requested resource after evaluation of the rule and a set of actions to be taken if a user is denied access to the resource. You can also configure sets of actions to be taken depending on the result of the authorization expression itself.

For both entities—rules and expressions—the definitive result of evaluation of the expression determines which actions are taken. Not all rules of an authorization expression contribute to the definitive result of the expression. The only actions taken are for the rules that led up to the definitive result of the expression. For explanation of the definitive result, see "About Evaluation of the Rules of an Expression" on page 6-16.

This section includes the following topics pertaining to actions:

- About Actions For Rules and Expressions
- About Kinds of Actions
- About the Use of HTTP Header Variables and Cookies
- About Passing Information Using Actions
- Which Actions Are Returned?
- About Complementary Actions
- About the Evaluation Order of Authorization Actions

About Actions For Rules and Expressions

In addition to being able to define to whom the Allow Access part and the Deny Access part of a rule applies when you configure the rule, you can also specify separate sets of actions for each result of the rule.

You can configure actions for the following results of evaluation of rules and expressions:

- **Authorization Success:** For both rules and expressions
- **Authorization Failure:** For both rules and expressions
- **Authorization Inconclusive:** For expressions only

  This result occurs when evaluation of the rules of the expression for which the user qualifies produce conflicting results, or the user does not qualify for any rules of the expression.

Additional information about these conditions is provided in the following sections:

- For a description of the results of rules of an expression, see "About Authorization Rule Evaluation" on page 6-6.
- For a description of the results of expressions, see "About Authorization Expression Evaluation" on page 6-16.

About Kinds of Actions

Actions allow you to:

- Redirect the user's browser to another URL.

You can redirect URLs from the Access Server to an AccessGate or a WebGate.

- Pass information about the user to downstream applications in the same policy domain or a different one.

  Using HTTP header variables or cookies, you can use actions to pass the following kinds of information:

  – User profile

  – User's DN

  – Static text strings

  See "About the Use of HTTP Header Variables and Cookies" on page 6-38 for details about using header variables to pass information to downstream applications.

About the Use of HTTP Header Variables and Cookies

For guidelines on the usage of header variables and cookies, see the section in Chapter 5 titled "About the Use of HTTP Header Variables and Cookies" on page 5-55.

How Caching Header Variables Affects their Availability

If a header variable's value is dynamic, the value is not available until the Access Server cache is refreshed.

The refresh frequency is set in the Policy Cache Timeout field in the Access Server Configuration/*Name of Access Server* screen. If you plan to use header variables with dynamic values, ask your Master Administrator about the refresh frequency.

How Web Servers Handle Header Variables

Web servers process header variables differently. This variability affects how you must implement header variables in your applications.

Here are some examples:

- Netscape/iPlanet Web servers precede Oracle Access Manager variables with the string, HTTP:

  – If you define a variable called HTTP_CN, Netscape/iPlanet produces a variable called HTTP_HTTP_CN.

  – When you write an application that must read a header variable, the application must look for a variable called HTTP_HTTP_CN and not HTTP_CN.

- Microsoft IIS expects header variables to be defined with a dash, not an underscore. You would enter HTTP–CN, not HTTP_CN.

  The receiving application must read the variable as if it had an underscore. It looks for HTTP_CN, not HTTP–CN.

- The Lotus Domino Web server cannot pass Oracle Access Manager header variables.

For information about how to use header variables for various servers, refer to your Web server's documentation.

About Passing Information Using Actions

Actions can pass information about users to other applications in the same or a different policy domain. Table 6–3 provides examples of how to use actions.

Table 6–3 Using Actions to Pass Information to Other Applications

Task	Example
Personalizing the end-user's interaction with the receiving application	You can use an action to send the user's name to a downstream application. The application could use the name to greet the user with a personalized message when the user logs in.
Passing information in a header variable	You can use a header variable: ■ To pass membership information ■ To pass information about a user for purposes of single sign-on For SSO to work, the target application must be able to use the variable.
Redirecting users to a specific URL upon failure or success of the attempt to authorize	You can use redirection to send the user to another location. For example, you can redirect the user to your portal page following authorization

When configuring authorization actions, you can use HTTP header variables to pass static values or identify attributes. See "Actions and Header Variables" on page 5-55 for details. In an authorization action, the header variable is returned on every HTTP request that invokes the authorization rule. Header variables are only returned on the initial HTTP request associated with authentication.

If you configure the `obmygroups` parameter in an authorization action, and a user has accessed a resource and is authorized based on the associated rule, the header variable is cached along with the fact that the user is authorized for the specific URL. If after accessing the URL, the user is added to a new group, the new group membership is not represented in the `obmygroups` results until the cache has expired, or the number of items in cache pushes the authorization from the cache, or the policy is modified, forcing a cache flush.

If the user is added to a group before accessing a resource, the group is represented when he or she accesses the resource and is authorized. These results are then cached. If the user is removed from the group, the group continues to appear in `obmygroups` until the user's authorization is removed from the cache.

> **See Also:** *Oracle Access Manager Deployment Guide* for performance implications of using obmygroups and tuning tips

Which Actions Are Returned?

Different actions are returned, depending on the result of the authorization expression and the rule or rules that were decisive in producing the definitive result. The Access Server returns the actions for the results of the definitive rules—the final definitive rule and those of the rules that led up to it. It determines the actions to return based on the following considerations:

- If the result of an authorization expression is Deny Access, the Authorization Failure actions for all of the definitive rules are returned.

For example, for the following compound complex authorization expression, the user qualifies for the Deny Access conditions of Rule 5, Rule 6, and Rule 7. The Authorization Failure actions are returned for all of these rules, but no actions for Rule 3 are returned.

```
(R5 AND R6) AND (R3 OR R7)
```

- If the result of the authorization expression is Allow Access, the Authorization Success actions for the definitive rules are returned.

  For example, for the following compound complex authorization expression, the user qualifies for the Allow Access conditions of Rule 1, Rule 2, and Rule 4. The Access Server returns the Authorization Success actions for Rule 1, Rule 2, and Rule 4, which are the definitive rules.

  ```
  (R1 AND R2) AND (R4 OR R3)
  ```

  Because Rule 4 is the final definitive rule, the Access Server stops evaluating the expression after it. It does not evaluate Rule 3 because it has no effect on the outcome.

About Complementary Actions

You can combine the actions resulting from evaluation of two or more rules to produce a desired result. For example, the Authorization Success actions for Rule 1 and Rule 2 in the following expression are combined to present a personalized greeting to the user for authorized users.

```
Rule 1 AND Rule 2 OR Rule 3
```

Here is how the actions for the rules are specified:

- For Rule 1, the Authorization Success action directs the Access Server to return the user's cn in the HTTP_CN header variable.
- For Rule 2, the Authorization Success action directs the Access Server to return the text 'Hello' in the header variable HTTP_GREETING.

For example, Sonal qualifies for both rules of the compound condition of the expression. She is a member of the marketing department group and the IP address of her computer is 192.168.2.123. Because she was successfully authorized after evaluation of the expression, Sonal is presented with a personalized greeting when she logs into the downstream application, the resource she requested.

About the Evaluation Order of Authorization Actions

When you set actions in the default authorization policy and in specific policies for a policy domain, the action that is applied to a user depends on what policy is enforced. For example, suppose that you define three policies in a policy domain:

- Policy 1
- Policy 2
- Policy 3

Each policy is checked in order, from top to bottom. If the third policy listed in the domain is the one that is enforced, the actions (for example, header variables) are taken from Policy 3. If you want to ensure that a particular action is always used, add the action to each policy in the policy domain.

Adding an action to multiple policies should not cause the action to be returned more than once. The topic of returning the same action multiple times is covered in "About Duplicate Actions" on page 6-43.

Working with Authorization Actions

Following discussions provide procedures to work with authorization actions:

- Setting Actions for Authorization Rules
- Setting Actions for Authorization Expressions
- About Duplicate Actions
- Setting the System Default Duplicate Actions Behavior
- Setting the Duplicate Actions Behavior for an Expression
- Creating Custom Authorization Actions

Setting Actions for Authorization Rules

Use the Actions feature to define an authorization rule's actions for responding to authorization success and authorization failure results. An action returns a specific value, such as the value of an attribute.

Actions you specify correspond with access conditions in the following way:

- Authorization success actions apply to Allow Access conditions.
- Authorization failure actions apply to Deny Access conditions.

To create an action for an authorization rule

1. Launch the Access System, select the Policy Manager, and select My Policy Domains.
2. Select the policy domain containing the authorization rule whose actions you want to set.
3. Select the Authorization Rules tab.

   A page appears listing the authorization rules for the policy domain.

   Note: If you are just now creating a policy domain, you do not see any authorization rules.

4. Select the authorization rule for which you want to set actions.
5. Click Actions.
6. Click Add.
7. For each of the following conditions, configure the actions to be taken—the RedirectURL and the user information to be returned:
   - Authorization Success
   - Authorization Failure
8. Click Save.

Configuring an Authorization Action When Using Disjoint Domains

If you have disjoint domains, you need to configure an authorization scheme that enables searches for users with identical user IDs who reside in disjoint domains.

To configure an authentication scheme for disjoint domains

1. In the action that you define upon success, you need to set the following values:

   Type: HEADERVAR

   Name: HTTP_OBLIX_UID

   Return Attribute: obuniqueid

   Note: This must be done for both the default identity and access policy domains.

2. In addition you need to make the following configuration file changes:

   In the following file:

   PolicyManager_install_dir/access/oblix/apps/common/bin/globalparams.xml

   change the value of whichAttrIsLogin to ObUniqueID

   Make the same change in the following file:

   IdentityServer_install_dir/identity/oblix/apps/common/bin/globalparams.xm

Setting Actions for Authorization Expressions

You can define actions for three kinds of results of evaluation of an authorization expression: authorization success, authorization failure, and inconclusive results of the expression evaluation.

To create an action for an authorization expression

1. Launch the Access System, select the Policy Manager, and select My Policy Domains.
2. Select the policy domain that the authorization expression whose actions you want to set belongs to.
3. Select Default Rules.
4. Select the Authorization Expression tab.
5. Click Actions.
6. Click Add.
7. For each of the following conditions, configure the actions to be taken depending on the result of evaluation of the expression (that is, the RedirectURL to use and the user information to return):
   - Authorization Success
   - Authorization Failure
   - Authorization Inconclusive
8. Click Save.

About Actions for Inconclusive Results

An inconclusive result can be returned for an authorization expression under the following two conditions:

- The user qualified for conflicting Allow Access and Deny Access rules.
- The user did not qualify for any rules of the expression.

For information about the status codes the Access Server returns when an expression is evaluated to a result of inconclusive, see "Status Codes for an Inconclusive Result" on page 6-16.

About Duplicate Actions

Because an authorization rule can be reused within an authorization expression, it is possible that evaluation of each instance of the authorization rule producing the same result can cause the Access Server to return the same action more than once.

It is also possible that different rules of an expression could return the same actions. Conflict can occur when, after evaluating the expression, two or more rules contributing to the definitive result produce the same actions. (See "About Evaluation of the Rules of an Expression" on page 6-16 for an explanation of the definitive result.)

For example, if the action of one rule is to set the HTTP_GREETING variable text string, and the action of another rule is to set the variable to a different value, a conflict occurs if the actions of both rules are returned. Because HTTP_GREETING can be set to only one text string, the Access Server must determine which one to use.

For all cases except RedirectURLs, you can set an option that determines how the Access Server should handle duplicate actions.

WARNING: For RedirectURL, the Access Server always returns the last URL it encounters. You cannot override this behavior.

How Duplicate Actions Are Handled

How the Access Server handles duplicate actions is defined by a system default setting, which you can configure. However, you can override the system default behavior for the individual authorization expressions of policy domains and policies. Here are the three behaviors from which you can choose:

- Duplicate—If you choose this option, the Access Server appends each new value it encounters to the information it returns to the application requesting authorization for the user. (The Access Server does not check for duplicate information.) Select this option if the application expects to receive information for all instances of the action. In this case, the application must process the values of all duplicate actions returned to it. Use of this option may incur performance issues.
- Ignore Duplicate—If you choose this option, the Access Server removes all duplicate actions, and only the first instance of the action is returned to the application requesting authorization for the user. Each time an action value is added, the Access Server checks existing values to determine if the new action duplicates an existing one. If the Access Server finds one, it does not add the new value to those it returns to the application. In this case, any information inherent to the value of the repeated action is lost.

  Because the Access Server must check for duplicate actions, use of this option may incur performance costs.

- Override—If you choose this option, only the value of the last instance of the action is returned. Each new value overwrites the previous one, and previous values are lost. Do not select this option if the application requesting the authorization expects the results of all duplicate actions. This option is the most efficient one.

Duplicate Actions and WebGate Restrictions

The ability to process duplicate actions applies to AccessGates only. The Access Server sends to the WebGate the actions as specified by the duplicate actions policy—whether Duplicate, Ignore Duplicate, or Override. However, the WebGate supports only a single value for a header variable. Although it receives the duplicate actions, the WebGate overrides duplicates so that the last value set for the header variable is used. Values set for the same header variable by previous actions are lost.

Setting the System Default Duplicate Actions Behavior

You can specify a system default setting for how the Access Server should handle duplicate actions, if any occur. By default, the system setting applies to handling of duplicate actions resulting from evaluation of all authorization expressions under control of the Access Server. However, you can override it for an individual authorization expression.

To set the system default duplicate actions behavior for the Access Server

1. Launch the Access System, select the Access System Console, and select Access System Configuration.
2. Select Common Information Configuration.
3. Click Duplicate Actions.
4. Select the radio button to set the duplicate action behavior: Duplicate, Ignore, or Override.
5. Click Save.
6. Restart the server for the duplicate actions policy change to take effect.

Setting the Duplicate Actions Behavior for an Expression

For each authorization expression, you can specify how you want the Access Server to handle duplicate actions if any occur as result of evaluation of the expression. By setting the authorization expression's Duplicate Actions value, you override the system default Duplicate Actions behavior.

To set the behavior for handling duplicate actions for an expression

1. Launch the Access System, select the Policy Manager, and select My Policy Domains.
2. Select the policy domain that the authorization expression belongs to.
3. Select Default Rules.

   A page appears listing the default rules and the authorization expression for the policy domain.
4. Select the Authorization Expression tab.
5. Click Duplicate Actions.

The Duplicate Action Headers page appears.

6. Select the radio button for the duplicate actions behavior for the expression: Duplicate, Ignore, or Override.

Creating Custom Authorization Actions

You can specify customized actions to be performed following successful authorization of a user or failure to authorize the user. Implementing a custom action requires an authorization plug-in. When defining a customized action:

- Authorization success actions apply to Allow Access conditions.
- Authorization failure actions apply to Deny Access conditions.

Refer to the *Oracle Access Manager Developer Guide* for details on creating a plug-in. For information about actions, see "About Authorization Actions" on page 6-37.

To implement a custom action

1. Launch the Access System, select the Policy Manager, and select My Policy Domains.
2. Select the policy domain that the authorization rule belongs to.
3. Select the Authorization Rules tab.

   A page appears listing the authorization rules for the policy domain.

   If you are creating a policy domain, you do not see any configured authorization rules.
4. Select the authorization rule for which you want to set custom actions.
5. Click Custom Actions.

   You are not able to select Custom Actions unless at least one authorization plug-in has been defined.
6. Click Add.
7. Enter the information for the custom action to be taken following successful authorization of a user or failure to authorize the user.
8. Click Save.

Note: You can define multiple custom actions for Authorization Success or Authorization Failure.

About Authorization Schemes for Custom Plug-Ins

You can create authorization schemes for custom plug-ins that perform authorization tasks. You must be a Master Access Administrator to create and manage authorization schemes. After you create an authorization scheme, a Delegated Access Administrator can include the scheme in an authorization rule.

Overview of Authorization Schemes and Custom Plug-Ins

The Access System provides a default authorization scheme called Oracle Authorization Scheme that you can use for any authorization rules you create. However, you can create custom authorization schemes that include custom plug-ins used to perform different or additional tasks from those of the default scheme. After

you create a custom authorization scheme, Delegated Access Administrators can include the plug-in within an authorization rule.

The Access System supports writing authorization plug-ins in C and any language supported by the Microsoft .NET framework, including C, C++, and Visual Basic. For details about managed code for authorization plug-ins, see the *Oracle Access Manager Developer Guide*.

About Authorization Plug-Ins

A custom authorization plug-in is a shared library (.dll or .so) that the Access Server uses to make outbound calls to external business logic for determining user authorization privileges and actions.

You can write a custom plug-in for any purpose. For example, you may want to look up a user's bank balance from a mainframe application to determine authorization privileges.

In some cases, the plug-in may pass authorization actions in addition to other parameters. The types of information a custom plug-in can pass are the same as those you can configure for an authorization rule. They are:

- User profile attributes
- Configuration parameters, required or optional
- Context-specific information, such as HTTP header information

Task overview: Providing customized authorization plug-ins

1. Write the custom authorization plug-in.

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

   An Oracle Access Manager developer at your organization writes the custom authorization plug-in using the authorization plug-in application programming interface (API). The authorization plug-in API enables the Access Server to call external business logic to determine if a user is authorized to access a resource.

2. A Master Access Administrator configures an authorization scheme for each custom plug-in you want to use. See "About Authorization Schemes for Custom Plug-Ins" on page 6-45.

   The scheme specifies information about the custom plug-in such as the location of the plug-in and the parameters it takes.

3. A Delegated Access Administrator with management rights for the policy domain can include the authorization scheme in an authorization rule. The authorization rule can then be included in one or more authorization expressions for a policy domain or any of its policies.

> **Note:** A custom plug-in for authorization must be installed on each application server you want to protect.

Working with Authorization Schemes

This section includes the following sections which describe how to create and configure an authorization scheme for custom plug-ins.

- Specifying Authorization Plug-In Paths and Parameters
- Viewing Authorization Schemes

- Adding an Authorization Scheme
- Modifying an Authorization Scheme
- Deleting an Authorization Scheme

Specifying Authorization Plug-In Paths and Parameters

To create an authorization scheme, you use the Authorization Management feature in the Access System Configuration component of the Access System Console. When you create a scheme, you enter information to be passed to the shared library in the User Parameter, Required Parameter, and Optional Parameter fields. You also specify one or more custom plug-ins in an authorization scheme.

When you specify a shared library for your plug-in, you can enter a complete path or a relative path to the plug-in. A relative path is evaluated with regard to the Access Server's installation directory.

For example:

```
lib/myplug_in
```

is evaluated as

```
AccesServer_install_dir/access/oblixaccess/oblix/lib/my_plug_in
```

For information on how to create shared libraries, see the *Oracle Access Manager Developer Guide*.

User Parameters

User parameters are user attributes that are passed to the shared library when the authorization scheme is invoked.

By default, the user's DN (distinguished name) and IP address are passed to the shared library. You cannot change this setting. However, you can select other attributes to help identify the user requesting the protected resource.

Required Parameters

All parameters are name-value pairs. Required parameters for a plug-in are configured by the Master Access Administrator. Parameters can be passed at the authorization scheme level or at the rule level.

If you pass the parameter name-value pair at the authorization scheme level, it cannot be overridden at the rule level.

When a Delegated Access Administrator configures an authorization rule using the plug-in, he or she must provide values in the rule for each required parameter not supplied at the scheme level. The parameters are then passed to the plug-in at run time.

If you do not pass a required parameter name-value pair at the scheme level, you must provide it at the rule level.

Optional Parameters for Authorization Plug-Ins

Optional parameters help to define more fully the behavior of a plug-in. Optional parameters for a plug-in are configured by the Master Access Administrator. When a Delegated Access Administrator configures an authorization rule that uses the plug-in, he or she can choose to provide a name-value pair for each of these parameters. If optional parameters are specified, they are passed to the plug-in at run time.

For example, suppose a user allowed to access a bank account wants to withdraw more money than exists in the account. The optional parameters may specify that this account does not include overdraft protection and it may deny the user's request.

Viewing Authorization Schemes

You may want to view the contents and definition of existing authorization schemes before you create new ones.

To view configured authorization schemes

1. Launch the Access System and select Access System Console, select Access System Configuration, and then select Authorization Management.

   The Authorization Management: List all authorization schemes screen appears.

2. Click the link for the scheme you want to view.

   The Details for Authorization Scheme page appears with the scheme's settings.

Adding an Authorization Scheme

If the existing authorization scheme does not meet your requirements, you may want to create a new one. In this case, as described in the previous sections, custom plug-ins must be available for the new scheme. Only a Master Access Administrator can create authorization schemes.

To create an authorization scheme

1. Launch the Access System and select Access System Console, select Access System Configuration, then select Authorization Management.

   The Authorization Management: List all authorization schemes page appears.

2. Click Add.

   The Define a new authorization scheme page appears.

3. In the Name field, type the name of the authorization scheme.
4. In the Description field, type a brief description of the scheme.
5. For the Plugin is managed code entry, if you are developing the plug-in using managed code, select Yes.
6. In the Managed Code Name Space field, enter the name space if you are using managed code. (If not, leave this field blank.)
7. In the Shared Library field, type the full path to the plug-in file or a path relative to the Access Server's installation directory without specifying the file extension.
8. In the User Parameter field, type the LDAP attributes to be passed to the plug-in.

   To pass in data from an external source to the plug-in, for example, to pass HTTP header variables, see "Retrieving External Data for an Authorization Request" on page 6-49.

9. In the Required Parameter field, type the name and value of parameters the policy domain authorization rule must send to the plug-in.

   If you specify the value for a parameter here, end users cannot change the value.

10. In the Optional Parameter field, type the name and value of parameters the policy domain authorization rules may send to the plug-in.

If you specify the value for a parameter here, end users cannot change the value.

For the User Parameter, Required Parameter, and Optional Parameter fields, click the plus (+) or minus (-) symbols to add or delete fields.

11. Click Save.

Modifying an Authorization Scheme

A Master Access Administrator is the only one who can modify an authorization scheme.

To modify an authorization scheme

1. Launch the Access System select Access System Console, select Access System Configuration, then select Authorization Management.

   The Authorization Management: List all authorization schemes page appears.

2. Click the link for the scheme you want to modify.

   The Details for Authorization Scheme screen appears.

3. Click Modify.

   The Modify Authorization Scheme screen appears.

4. Modify the parameters as necessary.
5. Click Save.

Deleting an Authorization Scheme

A Master Access Administrator is the only one who can delete an authorization scheme.

To delete an authorization scheme

1. Launch the Access System and select Access System Console, select Access System Configuration, then select Authorization Management.

   The Authorization Management: List all authorization schemes page appears.

2. Select the scheme you want to delete.
3. Click Delete.
4. Click OK to confirm your decision.

Retrieving External Data for an Authorization Request

An authorization scheme can obtain data from an external source, for example, a session, cookie, or header value that is set by a portal. The data is usually information about the user who is requesting access to a resource. The data is passed to a custom authorization plug-in. By obtaining external data, authorization decisions can be made dynamically. For example, if a user goes to a form to purchase an item for $1000, this $1000 amount can be dynamically evaluated against a limit—perhaps stored in a database—to determine if the purchase is authorized.

Oracle Access Manager obtains the external data using a "reverse action" in an authorization request. A reverse action refers to the process for obtaining data. Usually, when you use Oracle Access Manager, the data flows from the Access Server

to the AccessGate or WebGate. In contrast, a reverse action sends data from the AccessGate or WebGate to the Access Server.

The reverse action feature can be used with WebGates and with custom AccessGates. For both cases, you must write a custom authorization plug-in. The following is a source code example that can be the basis for this plug-in:

`Access_Server_install_dir\oblix\sdk\authorization\samples\req_context.c`

When you configure the authorization scheme, you must supply a path and a name for the shared library that is created when the code is compiled. If you have Access Servers on different platforms that access this library, the path and the name must be identical on both servers.

See the *Oracle Access Manager Developer Guide* for details. In particular, refer to the information on the isAuthorized call for the ObUserSession class.

When writing a custom AccessGate that correctly handles a reverse action, error processing for this plug-in is as follows. If an isAuthorized call fails to pass required data to the authorization plug-in, the Access Manager SDK returns ObUser_ERR_NEED_MORE_DATA. In this case, the AccessGate can use the getAuthorizationParameters call in the ObResourseRequest structure to discover what data is required, gather the data, and reissue the isAuthorized call. The access_test_cplus program in the Access Manager SDK installation directory contains examples of these calls.

To retrieve external data for an authorization request

1. Create an authorization scheme as described in "About Authorization Schemes for Custom Plug-Ins" on page 6-45.
2. In the User Parameter field, enter the following:

   `RA_source$name`

   or

   `RA_name`

   where *source* is one of the following:

   - server
   - header
   - post
   - query
   - cookie

   For information about the User parameter, see "User Parameters" on page 6-47.

   If you omit the value for *source*, sources are searched in the order shown in the list. Note that the Web server source (the server parameter) takes precedence over other sources. This prevents the request data, which is under control of the user, from overriding Web server data. For example, a remote_user cookie sent from a user does not override a remote_user variable sent by the Web server. The WebGate automatically extracts the requested data from the HTTP request.

   If the custom client or AccessGate is created using the Access Manager SDK, it is up to the application program calling the Access Manager API to collect this data.

3. Create a custom authorization plug-in to process the external data sent by the WebGate or custom AccessGate and to return an authorization decision and optionally, action data.

   Note that in the authorization scheme that you defined in this procedure, the RA prefix for the user parameter instructs the Access Server to go to the plug-in to make the external request.

Example: Configuring a WebGate to Use Authorization Data from and External Source

Most browsers accept several standard headers in the HTTP requests that they send to servers. In this example, an authorization scheme uses the accept-language header, tells the WebGate to obtain the value of the header that is sent from the user's browser, and authorizes users if the browser language is set to en-us.

In the following example, the distributed example authorization plug-in named req_context compares the value in an incoming header against another value.

The req_context authorization plug-in is a general purpose plug-in to check external data, for example, HTTP headers retrieved by an authorization action that looks for external data. The plug-in compares the external data to specified values, either fixed values or user attribute values. A type parameter of "fixed" means that the data specified by the Name parameter is to be checked against the actual string in the Value parameter. A Type parameter of "attribute" means that the named external data is to be checked against the value of the user attribute specified in the Value parameter. In this example, a fixed value is used, however, the target value could be an attribute.

For example, the req_context plugin parameters for one authorization rule that allows only American English browsers could be specified as follows:

Name: RA_accept-language
Type: fixed
Value: en-US

In this example, the value of the accept-language header is to be checked against the absolute (or fixed) value "en-us".

An authorization rule to allow only French browsers could specify the following:

Name: RA_accept-language
Type: fixed
Value: fr

You could use the req_context plug-in to check if the user's browser language matches a language configured for the user in an "expected-language" attribute in the user's directory entry, as follows:

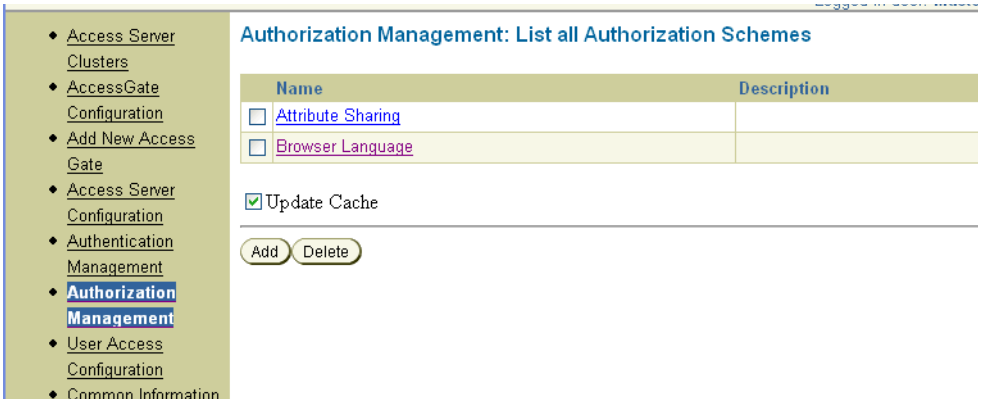
Name: RA_accept-language
Type: attribute
Value: expected-language

As an alternative method for finding a fixed value, you also could write a special purpose plug-in that only checks for "en-us", in which case the only required parameter would be the user attribute "RA_accept-language".

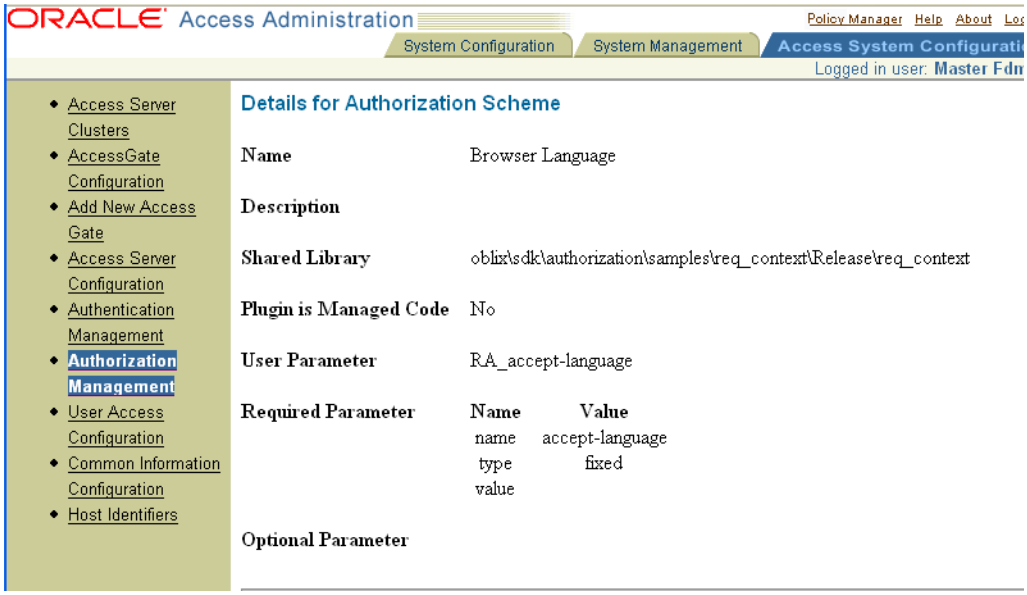
Note: Note that if you try to test the scheme shown in the following procedure, retrieval of the accept-language header may be case-sensitive on some browsers.

To configure a sample scheme to obtain external authorization data

1. In the Access System Console, create an authorization scheme named Browser Language:



Once you have defined the Browser Language authorization scheme, it appears in the list of schemes in the Authorization Management page. The details of the authorization scheme are as follows:



Note that the shared library is based on the sample code in the following plug-in:

`oblix\sdk\authorization\samples\req_context.c`

See "To retrieve external data for an authorization request" on page 6-50 for details. Note also that this scheme makes use of the RA_accept_language user parameter. In the required parameters for this scheme, the name accept-language is provided, with a type of fixed and no value.

2. In the Policy Manager, define a new policy domain.

   The authorization rule in this policy domain looks for the language setting of the user's browser and authorizes the user if the browser language value is en-us, as follows:

   Name: Browser language

Resource: http (/protected)

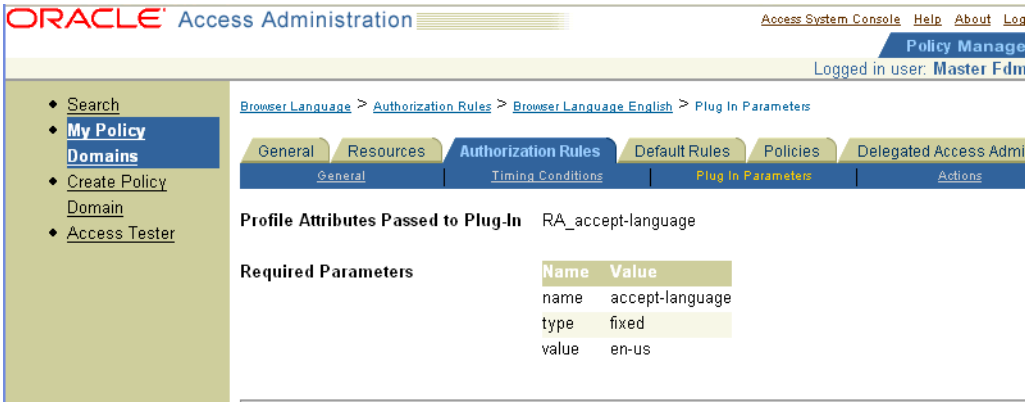
Authorization Rule Name: Browser Language English

Authorization Rule General: The authorization scheme (that was defined in the Access System Console) is Browser Language

Authorization Rule Plug-in Parameters: The profile attribute that is passed is RA_accept-language, and the value is en-us

The plug-in parameters for this policy domain appear as follows:

Name: accept-language
Type: fixed
Value: en-us



Auditing Authorization Events

An audit rule causes event-based data to be written to the audit log file. As a Master Access Administrator, you must create a Master Audit Rule in the Access System Console. As a Delegated Access Administrator, you can derive audit rules from the Master Audit Rule for your policy domains and policies, but you cannot create an alternative Master Audit Rule.

There is one audit log for each Access Server. You can configure the size of the audit log file and the rotation interval for a server. Depending on events, the audit log may contain some duplicate audit entries.

Information Logged on Success or Failure

Different information is written to the audit log depending on whether the user was authorized to use the requested resource.

For authorization failure, if information for a user does not exist in the directory, the Access Server denies the user access to a resource. In this case, the cn attribute is written in the log entry. No other attributes are written, because none are available. Because there is not an entry for the user, attributes such as givenname have no meaning. In this case, the user requesting access to a resource had not previously been authenticated.

About Creating a Master Audit Rule and Derived Rules

You can define audit rules for a policy domain and its policies. Any audit rules you define must be derived from a Master Audit Rule. A Master Audit Rule must be

created by a Master Access Administrator. Delegated Access Administrators can derive access rules from the Master Audit Rule, but they cannot create them.

For details explaining how to create and define these audit rules for policy domains and their policies, see the following sections in the policy domain chapter:

- "Auditing User Activity for a Policy Domain" on page 4-41
- "About Creating a Master Audit Rule and Derived Rules" on page 6-53
- "Creating an Audit Rule for a Policy Domain" on page 42
- "Defining an Audit Rule for a Policy" on page 4-43

7

Configuring Single Sign-On

The Access System's single sign-on capability enables users to access multiple protected URLs or applications with a single login. Before reading this chapter you should be acquainted with the terms and concepts covered in Chapter 4, "Protecting Resources with Policy Domains" on page 4-1.

This chapter covers the following topics:

Prerequisites

Before attempting to configure single sign-on, you need to have a working Identity and Access System. This includes installing and configuring your directory server, the Identity System, the Policy Manager and Access Server, and at least one WebGate or Access Gate. For complete details, see the *Oracle Access Manager Installation Guide*.

About Single Sign-On

Single sign-on gives users the ability to access multiple protected resources (Web pages and applications) with one authentication. The Access System enables you to protect Web sites and applications by defining what resources you want to protect and providing rules for accessing the resource. The rules are for:

- **Authentication:** Authentication is the process of proving that a user is who he or she claims to be. To authenticate a user, a WebGate presents the user's browser with a request for authentication credentials in the form of a challenge. The challenge is referred to as a challenge method or authentication method.
- **Authorization:** Authorization is the process of determining if a user has a right to access a requested resource. A user may want to see data or run an application program protected by a policy. The requested resource may belong to a policy

domain, or it may be covered within that domain by a specific policy that is different from the global one.

For more information on protecting access to a single resource, see Chapter 4, "Protecting Resources with Policy Domains" on page 4-1.

Different Types of Single Sign-On

Single sign-on can be implemented in a variety of ways:

- **Single domain:** You can set up single sign-on for a set of URLs within the domain (mycompany.com., for example).
- **Multi-domain:** You can set up single sign-on for a set of URLs that reside within multiple domains (mycompany.com and yourcompany.com., for example).
- **Applications and third-party products:** You can set up single sign-on between Oracle Access Manager and an IBM WebSphere Application Server, for example.

The first two implementations use encrypted cookies, as explained in "Single Sign-On Cookies" on page 7-2. For these implementations to work, end users must enable their browsers to receive cookies.

See Also:

- *Oracle Access Manager Integration Guide* chapter on integrating single sign-on with Oracle Application Servers
- *Oracle Fusion Middleware Security Guide* 11g for details about implementing authentication or identity assertion using the Oracle Access Manager Authentication provider with Oracle WebLogic Server

Single Sign-On Cookies

The Access System implements single-domain and multi-domain single sign-on through an encrypted cookie called the *ObSSOCookie*. The WebGate sends the ObSSOCookie to the user's browser upon successful authentication. This cookie can then act as an authentication mechanism for other protected resources that require the same or a lower level of authentication.

When the user requests access to a browser or another resource, the request flows to the Access Server. The user is logged in, and the ObSSOCookie is set. The Access Server generates a session token with a URL that contains the ObSSOCookie. Single sign-on works when the cookie is used for subsequent authorizations in lieu of prompting the user to supply authorization credentials.

When the cookie is generated, part of the cookie is used as an *encrypted session token*. The encrypted session token contains the following information:

- The distinguished name (DN) of the authenticated user.
- The level of the authentication scheme that authenticated the user.

  See "About Authentication and Authentication Schemes" on page 5-1 for details.
- The IP address of the client to which the cookie was issued.
- The time the cookie was originally issued.
- The time the cookie was last updated.

If the user has not been idle, the cookie is updated at a fixed interval to prevent the session from timing out. The update interval is one-fourth of the length of the idle session timeout parameter. See "Viewing AccessGate Profiles" on page 3-17 for details.

Unencrypted ObSSOCookie data includes:

- Cookie expiry time.
- The domain in which the cookie is valid.
- An optional flag that determines if the cookie can only be sent using SSL.

Security of the ObSSOCookie

The ObSSOCookie is a secure mechanism for user authentication. When the Access System generates the cookie, an MD-5 hash is taken of the session token. When the ObSSOCookie is used to authenticate a user, the MD-5 hash is compared with the original cookie contents to be sure no one has tampered with the cookie. MD-5 is a one-way hash, so it cannot be unencrypted. The Access Server does the comparison by hashing the session token again and comparing the output with the hash of the token already present in the cookie. If the two hashes do not match, the cookie is corrupt. The system relies on the fact that if someone tampers with the session token, the hashes will not match.

The single sign-on cookie does not contain user credentials such as user name and password.

Configuring the ObSSOCookie

Configuring the ObSSOCookie is a one-time activity conducted by a Master Administrator or Master Access Administrator. The cookie is encrypted using a configurable encryption key known as a *shared secret*.

- For shared secret keys used in installations of version 5.x, the RC4 encryption scheme was recommended.
- For shared secret keys used in installations of version 6.x, the RC6 encryption scheme was recommended.
- AES is a new encryption scheme introduced in version 7.0.

  In version 7.0 and higher, shared secrets use this encryption scheme as the default.

Oracle Access Manager 10.1.4 does grandfather the ObSSOCookie *only* if the shared secret is regenerated and not for changes in the configuration of the cipher to be used. Oracle Access Manager always tries to use the newer shared secret when decrypting the ObSSOCookie. If this is not successful, it uses the older shared secret. If this fails, the Access System queries the Access Server to see if a new shared secret was generated. If none of the keys is successful, the user is prompted to re-authenticate.

The shared secret encryption algorithm is an Oracle Access Manager-wide setting. It affects all encrypted cookies, not just the ObSSOCookie.

Oracle recommends that you upgrade all older WebGates even though these may coexist with 10.1.4 Access Servers. In environments that include a mix of WebGate releases, use the encryption scheme that corresponds to the earliest WebGate. For example:

- Use RC4 as the encryption scheme if you have release 5.x and 10.1.4 WebGates co-existing in the same system.

- Use RC6 as the encryption scheme if you have release 6.x and 10.1.4 WebGates co-existing in the same system.
- Use the AES encryption scheme if you have only release 7.0 or 10.1.4 WebGates co-existing in the same system.

For single sign-on to work with WebGates older than release 6, you must continue to use the RC4 encryption algorithm until all WebGates are upgraded. Similarly, release 6 WebGates require RC6 for the older WebGates to be compatible with new WebGates, and for single sign-on to work. See the *Oracle Access Manager Upgrade Guide* for more information about ensuring backward compatibility between new Access Servers and older WebGates.

Note: Oracle recommends that administrators use AES as the encryption algorithm. It is a much stronger algorithm than RC4 or RC6. RC6 encryption is deprecated in Oracle Access Manager 10.1.4, and its support will be dropped in future releases.

To configure the ObSSOCookie

1. Generate a key to encrypt the ObSSOCookie from the Access System Console.

   See "Creating a Shared Secret Key" on page 8-4 for details.

2. Decide if you want the ObSSOCookie to be sent only using SSL.

   See "Securing the ObSSOCookie in an Authentication Scheme" on page 5-16 for details.

Single Domain Single Sign-On

The simplest form of single sign-on occurs within a single domain. For example, suppose within the domain mycompany.com you are hosting several restricted Web sites on several hosts. You can set up single sign-on so that users with the right privileges can access all or a subset of these restricted areas after just one authentication.

In order for single domain single sign-on to work, you need a fully functional Oracle Access Manager system, including at least two WebGates, as described in the following sections.

The rest of this section discusses the following topics:

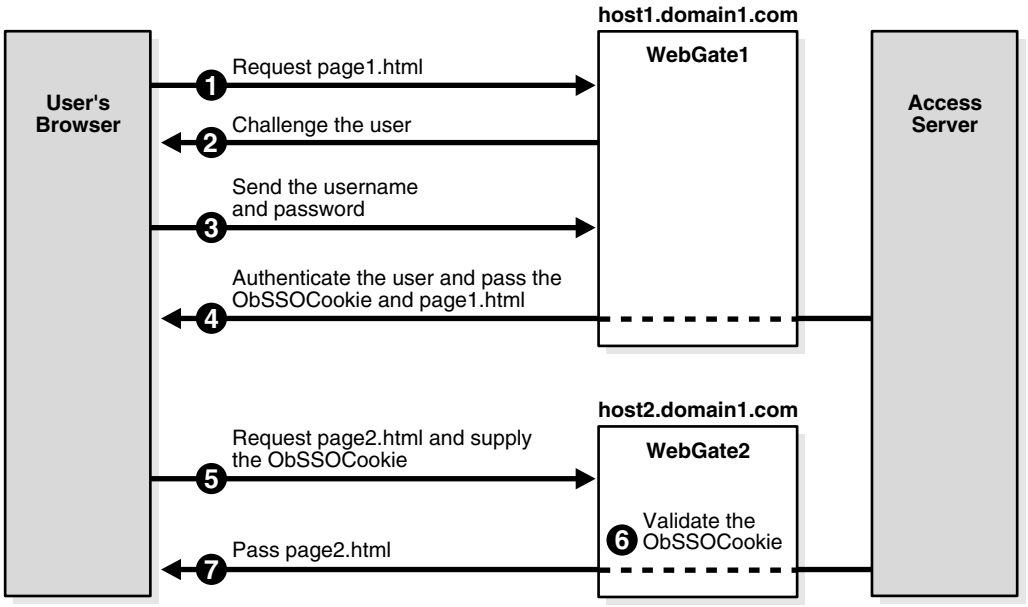
How Single Domain Single Sign-On Works

In single domain single sign-on, the ObSSOCookie is associated with a particular domain, for instance, domain1.com. The user authenticates to a WebGate that protects a server, for example, foo1.domain1.com and the ObSSOCookie is set. Then the user requests a resource on another server in that domain—for example, foo2.domain1.com. For the second request, the same ObSSOCookie can be used and the user does not have to re-authenticate even though he or she has requested information on separate servers in that domain.

Single domain single sign-on works by passing the ObSSOCookie among the WebGates configured for the domain. For example, suppose a user requests index.html on Host1 through a Web browser protected by WebGate1. The process overview in Figure 7–1 illustrates the events as WebGate1 on Host1 asks the user for a user name and password. If the Access Server accepts the user's authentication, the Access Server gives WebGate1 permission to give the user access to index.html. Then WebGate1 gives the user access to index.html along with the ObSSOCookie.

If this user now wants to access Host2, the user's Web browser sends a request to WebGate2 for a page from Host2 along with the ObSSOCookie. If the two WebGates have the same cookie domain, WebGate2 can look at the ObSSOCookie and determine if the user is authenticated. The user does not have to re-authenticate.

Figure 7–1 Single Domain Single Sign-On



In Figure 7–1, the process flow is as follows:

1. The user requests page1.html on host1.domain1.com.
2. The WebGate that protects this server presents an authentication challenge.
3. The user presents credentials that the WebGate passes to the Access Server.
4. The Access Server authenticates the user and passes the ObSSOCookie.
5. WebGate shows page1.html to the user.
6. The user requests page2.html on host2.domain1.com.

   This sever is protected by another WebGate and single-domain single sign-on is configured between this WebGate and the first one. The ObSSOCookie is included in the request.
7. The WebGate passes the ObSSOCookie to the Access Server, which validates the cookie and serves page2.html.

Setting up Single Domain Single Sign-On

The following is a summary of configuring a single domain for single sign-on.

Task overview: Enabling single domain single sign-on

1. Install a directory server and Web server according to the vendor's instructions.
2. Install and set up a working Oracle Access Manager system, as explained in the *Oracle Access Manager Installation Guide*.
   a. Install and set up the Identity System.
   b. Install and set up the Access System.
3. Set up a WebGate, as described in the procedure "To configure the WebGate" on page 7-6.
4. Configure access controls to a resource protected by this WebGate, as described in the "Task overview: Defining authentication and authorization schemes for single sign-on" on page 7-7.
5. Set up a second WebGate, as described in the procedure "To configure a second WebGate for single sign-on" on page 7-7.
6. Configure access controls to another resource protected by the second WebGate, again using the "Task overview: Defining authentication and authorization schemes for single sign-on" on page 7-7.
7. Specify the same primary cookie domain for the two WebGates.

Configuring the WebGates

This discussion assumes that you have completed WebGate installation as part of your Access System installation and setup. For more information, see "Prerequisites" on page 7-1.

To configure the WebGate

1. From the Access System Console, click Access System Configuration, then click AccessGate Configuration.
2. Configure the WebGate as explained in "Adding an AccessGate" on page 3-24, and be sure to:
   a. Add a domain name for the Primary HTTP Cookie Domain.

      For example:

      ```
      host1.mycompany.com
      ```

      Note: The more general the domain name, the more inclusive your single sign-on implementation will be. For example, if you specify b.com as your primary cookie domain, users will be able to perform single sign-on for resources on b.com and on a.b.com. However, if you specify a.b.com as your primary cookie domain, users will have to re-authenticate when they request resources on b.com.

   b. Set a value for user session timeout to define how long the ObSSOCookie lasts. Use the two Access Server parameters for setting this timeout:

      Maximum User Session Time: Specifies the number of seconds that a user's connection to a resource can last before the user must re-authenticate.

      Idle Session Time: Specifies the number of seconds that a cookie can remain valid without user activity. The shorter the session, the more frequently users

must re-authenticate. Shorter sessions are more secure because they leave less time for an unauthorized user to access an unattended browser or an intercepted cookie to be re-used in a replay attack.

For more information about these parameters, see Chapter 3, "Configuring WebGates and Access Servers" on page 3-1.

3. Configure multiple ways for a user to specify the fully qualified domain name.

   For single sign-on to work, users must enter a fully qualified domain name. You can create alternative ways to specify the domain name, as described in "Using Host Identifiers and Host Contexts" on page 4-31. If a preferred host is not specified, all known variations of IP addresses and URLs must be listed in the Host Identifier. This is the only way to prevent users from typing an IP address to bypass authentication and authorization.

4. Configure access controls to another resources protected by the second WebGate, as outlined in the following task overview.

Task overview: Defining authentication and authorization schemes for single sign-on

1. Create an authentication scheme for the domain and a level for the scheme.

   If you use different authentication schemes on the two WebGates, users can go from a higher authentication scheme to a lower one, but not from a lower one to a higher one.

   For example, if a user is granted access to a resource that has a Basic Over LDAP authentication scheme defined as having a level of 2, the user can access other resources that have schemes with the same or a lower level. However, if the user tries to access a resource with a more stringent authentication challenge, such as a scheme called Client Certificate with a level of 5, they must re-authenticate.

2. Create an authorization scheme, as described "Adding an Authorization Scheme" on page 6-48.

3. Take stock of your authorization schemes and consider the following:

   Users who use single sign-on may pass the authentication tests but may fail the authorization tests when attempting to access a second or third resource. Each resource in the domain may have a unique authorization scheme.

4. Configure a second WebGate for single sign-on, as described in the next procedure.

To configure a second WebGate for single sign-on

1. Configure a second WebGate for a set of resources in the same domain.

   Give the second WebGate a domain configuration identical to the first WebGate, and be sure that it communicates with an Access Server in the same installation as the first WebGate. See "Configuring AccessGates and WebGates" on page 3-17 and "Associating AccessGates and WebGates with Access Servers" on page 3-41 for details.

   For example, set up a WebGate for:

   ```
   host2.mycompany.com
   ```

2. From the Access System Console, click Access System Configuration, then click AccessGate Configuration.

3. Click the link for the first WebGate.
4. Click Modify.
5. In the Primary HTTP Cookie Domain field, enter the domain using a *.domain.domain* format.

   For example:

   ```
   oracle.com
   ```

6. Click Save.
7. Click Back.
8. Select the second WebGate, click Modify, and enter the same domain.

   Note: The primary HTTP cookie domains must be *identical* for the two WebGates.

9. Save your work.

   When two WebGates are set up, single sign-on should work between them. You must install a WebGate on each Web server that you want to protect.

Reverse Proxy Single Sign-On

If you are going to use a reverse proxy in a single sign-on configuration, be sure either to set the ipvalidation parameter to false or to add the proxy IP address to the IPValidationExceptions list in the AccessGate configuration. Otherwise, the reverse proxy hides the client's IP address. See "Configuring IP Address Validation for WebGates" on page 3-37 for details.

In some situations the Reverse Proxy does not pass the ObSSOCookie to Oracle WebLogic after a successful authentication. To avoid this issue, use Form Based authentication instead of Basic Over LDAP when using Reverse Proxy with Oracle WebLogic.

See Also: *Oracle Fusion Middleware Security Guide* chapter on configuring SSO for Oracle Fusion Middleware

Logout From a Single Domain Single Sign-On Session

By default, 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."

WebGate also treats any designated URL as a signal to log the user out of the single sign-on domain. The logout URL is configured in the AccessGate configuration page. See "AccessGate Configuration Parameters" on page 3-19 for details. If the configuration is not specified, then the default behavior is used.

For example, you can configure the following logout URLs:

```
/access/oblix/lang/%lang%/logout.html
/logout.htm
```

In the first example URL, `%lang%` represents the directory for a specific language pack.

The default behavior can be turned off by specifying custom URLs in the LogOutURLs setting for the WebGate. You can specify multiple logout URLs. See "AccessGate Configuration Parameters" on page 3-19 for details. For each browser request, the list of configured logout URLs is scanned to determine whether the user will be logged out of the single sign-on domain. On UNIX computers, the logout URLs are case sensitive.

The number of logout URLs affects the performance of the WebGate.

Multi-Domain Single Sign-On

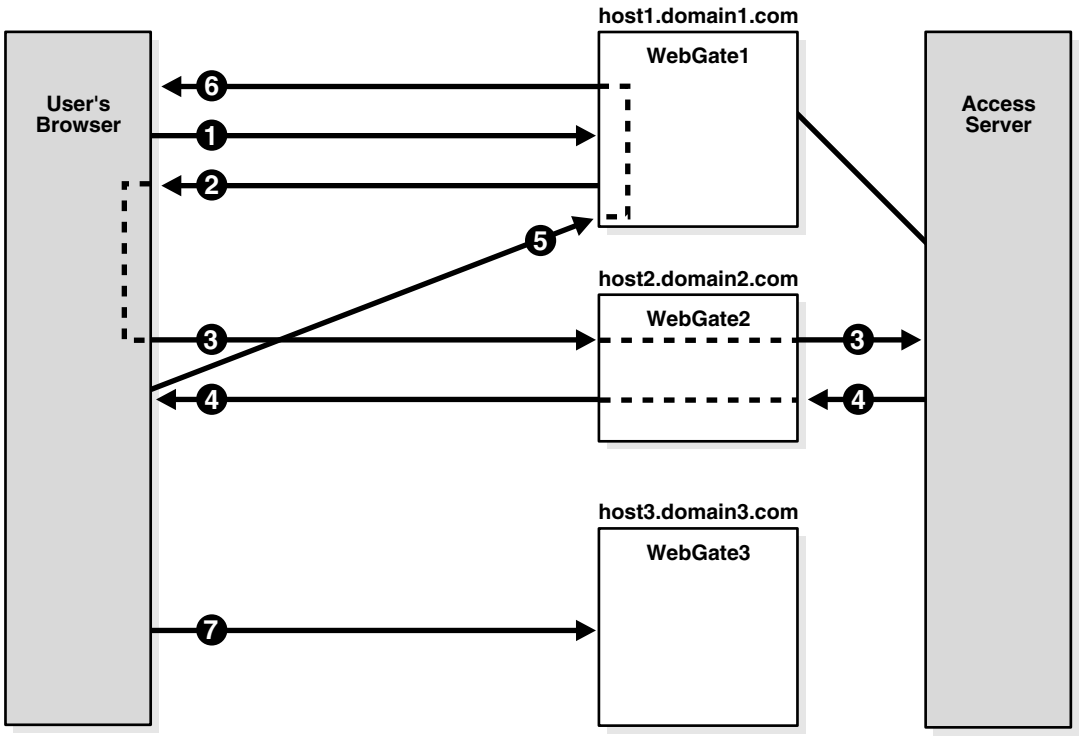
Multi-domain single sign-on allows a user authentication to be honored by all the hosts in two or more domains. The main objective in multi-domain single sign-on is to provide the user with an ObSSOCookie from each domain. Cookies cannot be sent across multiple domains. To achieve single sign-on across multiple domains, the Access System requires that you specify a primary domain for authentication. This primary domain acts as a central hub for all authentications. Regardless of what domain users try to authenticate to, each WebGate redirects them to the primary domain expressed as a single URL.

Multi-domain single sign-on is implemented, and works, in much the same way as single domain single sign-on. For more information, see "Single Domain Single Sign-On" on page 7-4 and note the following differences:

- For single domain single sign-on, you configure WebGates in one domain. However for multi-domain single sign-on, you configure WebGates on each authentication server in each domain and designate one of the authentication servers to be the primary authentication server.
- For single domain single sign-on, the WebGate provides the user with an ObSSOCookie from one domain, and that cookie is valid for each protected resource in the domain. However, for multi-domain single sign-on, a series of redirects provides the user with a different ObSSOCookie from a designated WebGate in each domain.
- For multi-domain single sign-on to work, WebGates in all domains must have access to the complete set of authentication schemes. This means that the Access Servers in your environment must use the same policy directory. If necessary, this directory can be replicated.
- Multi-domain single sign-on works only with WebGates, not AccessGates. For example, the applications discussed in "Application Single Sign-On" on page 7-12 have their own single sign-on methods. To integrate a scheme for AccessGate-based single sign-on with a scheme for WebGate-based multi-domain single sign on, you need to configure a proxy to act as a front end for these AccessGates.

  Figure 7–2 illustrates the process of providing the user with an ObSSOCookie from multiple domains. An explanation of the diagram follows.

Figure 7–2 Multi-Domain Single Sign-On



Process overview: Multi-domain single sign-on

1. The user initiates a request for a Web page from a browser.

   For instance, the request could be for host1.domain1.com/page1.html.

2. WebGate1 on host1.domain1.com sends the authentication request back through the user's browser in search of the primary authentication server.

   In this example, you have designated host2.domain2.com to be the primary authentication server.

3. The request for authentication is sent from the user's browser to the primary authentication server.

   This request flows to the Access Server. The user logs in and the ObSSOCookie is set for domain2.com. The Access Server also generates a session token with a URL that contains the ObSSOCookie.

4. The session token and ObSSOCookie are returned to the user's browser.

5. The session token and ObSSOCookie are sent to host1.domain1.com.

6. The WebGate on host1.domain1.com sets the ObSSOCookie for its own domain (domain1.com) and satisfies the user's original request for the resource host1.domain1.com/page1.html.

7. If the user later sends a request to host3.domain3.com, a similar set of redirections takes place to set the cookie for that domain.

   Since the ObSSOCookie for the primary domain has been set, the user would not have to log in to domain3.

As mentioned earlier, implementing multi-domain single sign-on is similar to implementing single domain single sign-on.

Task overview: Implementing multi-domain single sign-on

1. Use the "Task overview: Enabling single domain single sign-on" on page 7-6 as a guide and be sure to implement the differences described in "Multi-Domain Single Sign-On" on page 7-9.
2. Implement redirection as described in "Using Redirection to Enable Multi-Domain Single Sign-On" on page 7-11.
3. Test your implementation, as described in "Testing Multi-Domain Single Sign-On" on page 7-12.
4. Configure logout, as described in "Logout from a Multi-Domain Single Sign-On Session" on page 7-12.

The rest of this section discusses the following topics:

- Using Redirection to Enable Multi-Domain Single Sign-On
- Testing Multi-Domain Single Sign-On
- Logout from a Multi-Domain Single Sign-On Session

Using Redirection to Enable Multi-Domain Single Sign-On

For each WebGate in a multi-domain SSO configuration, you need to define an authentication scheme with redirection rules. For instance, suppose you have three authentication servers, each in a separate domain:

- host1.domain1.com
- host2.domain2.com—your primary authentication server
- host3.domain3.com

Each WebGate can only set the ObSSOCookie for its own domain. Therefore, you must create redirection rules so that when a user logs in, they are redirected to the primary authentication server. In this example, the primary authentication server is host2.domain2.com.

> **Note:** A redirection rule is needed even for the primary authentication server.

For more information, see the next procedure.

To configure redirection

1. From the Access Server Console, click Access Server Configuration, then click Authentication Management.
2. Click a link for an authentication scheme.
3. In the Challenge Redirect field, enter the primary authentication server for your multi-domain single sign-on scheme.
4. Repeat these steps for WebGates in each domain in your multi-domain SSO scheme, and for all authentication schemes that are protecting resources in these domains.

   This procedure redirects the servers across domains to the primary authentication server.

Next, you need to be sure that the ObSSOCookie can be passed among WebGates within a particular domain.

5. Within each individual domain, ensure that each WebGate is configured to use the same primary HTTP cookie domain. See "Configuring the WebGates" on page 7-6.

> **Note:** If you do not specify a primary cookie domain within a single domain, the multi-domain ObSSOCookie will not be usable by other WebGates within an individual domain.

6. Test your multi-domain single sign-on as described in "Multi-Domain Single Sign-On" on page 7-9.

Testing Multi-Domain Single Sign-On

To test a multi-domain single sign-on configuration, set your browser to notify you when you receive cookies. If single sign-on is working, you should receive notification of session cookies from each domain you have configured.

Logout from a Multi-Domain Single Sign-On Session

When you log out of an application, the Access System only removes the ObSSOCookie for the current domain. For example, if you are logged into domain1, domain2, and domain3, and you log out from domain1, only the ObSSOCookie for domain1 is removed.

The timeout of the cookie is always determined by the computer that performs the authentication. For example, suppose www.a.com sets a cookie expiration of one hour, and www.b.com sets a cookie expiration of 30 minutes. A user goes to www.b.com and is redirected to www.a.com for authentication. After 30 minutes the www.b.com cookie expires and the user is redirected to www.a.com. The cookie for www.a.com is still valid, so the user is not prompted to re-authenticate. The domain www.b.com sets a new cookie with a fresh timeout value. For details about idle session timeout and maximum user session timeout, see "AccessGate Configuration Parameters" on page 3-19.

You can set the www.a.com timeout value to be less than that of any other domain. This guarantees that authentication happens any time one of the other domain's cookies expires. The drawback is that if you set www.a.com expiration to be too short, single sign-on may not happen because www.a.com's cookie can expire before the user's next attempt at single sign-on. You need to determine the balance between single sign-on functionality and expiration policy.

> **WARNING: If you configure multi-domain single sign-on for your users, be sure to tell them to close all browser windows or to explicitly log out of each domain to which they are still logged in.**

Application Single Sign-On

The Access System enables you to create a web of trust in which a user's credentials are verified once and are provided to each application the user runs. Using these credentials, the application does not need to re-authenticate the user with its own mechanism. Application single sign-on allows users who have been authenticated by the Access System to access applications without being re-authenticated.

There are two ways to send a user's credentials:

- **Using Cookies:** A specific value is set on the browser's cookie that the application must extract to identify a user.
- **Using Header Variables:** An attribute name-value pair is appended to the URL that calls the application.

With both forms of single sign-on, additional programming is required.

Header variables can be redirected only to Web servers known or protected by the Access System. Header variables passed as authentication actions are not persistent during a user session. See "Managing Authentication Actions" on page 5-53 for information about authentication actions.

For example, when a user authenticates, they may be redirected to a portal index page:

```
http://mycompany.com/authnsuccess.htm
```

For authentication failure, an authentication action may redirect the user to an error page or a self-registration script:

```
http://mycompany.com/authnfail.htm
```

The rest of this section discusses the following topics:

Additional Information on Application Single Sign-On

For more information on application single sign-on, see the *Oracle Access Manager Integration Guide*. Here are a few of the integrations that Oracle is certifying:

Integration with Oracle Identity Federation: Enables federated authentication and authorization. The *Oracle Access Manager Integration Guide* discusses federated authorization. Federated authentication and authorization are discussed in *Oracle Secure Federation Services Administration Guide*.

Integration with OracleAS 10g, OC4J: Enables Oracle Access Manager single sign-on and identity management across applications that run on Oracle AS, such as Oracle eBusiness Suite.

Integration with RSA SecurID: SecurID is a two-factor authentication product from RSA Security. The Access System provides a plug-in and other components to provide native SecurID authentication.

Integration with mySAP: Enables Oracle Access Manager single sign-on for mySAP applications and other Oracle Access Manager-protected enterprise resources and applications. It also enables you to configure Oracle Access Manager authentication schemes for mySAP applications.

Integration with the Plumtree Corporate Portal: Provides companies with a Web enterprise solution for building customized, secure business portals with integrated, identity-based Web access management. In this solution, the Plumtree Corporate Portal acts as a gateway to an enterprise intranet or extranet, providing users centralized access to applications and content hosted by the enterprise.

Connector for WebSphere: Enables applications running on IBM WebSphere to be integrated with Oracle Access Manager access control and identity management features. The Connector for WebSphere enables J2EE resources and applications on WebSphere to use the Access System for authentication, authorization, auditing, and

single sign-on. It also provides the Identity System for identity management features such as delegated administration, dynamic groups, and workflows.

Security Provider for WebLogic SSPI—This implements single sign-on across J2EE applications that are deployed in the Oracle (formerly BEA) WebLogic platform. The Security Provider enables WebLogic administrators to use Oracle Access Manager to control access to business applications. The Security Provider provides authentication to WebLogic Portal resources and supports single sign-on between Oracle Access Manager and the WebLogic Portal Web applications. The Security Provider also offers user and group management functions.

Logging Out From an Application Single Sign-On Session

The Access System sets the ObSSOCookie for each user or application that accesses a resource protected by the Access System. The ObSSOCookie enables users to access other resources protected by the Access System that have the same or a lower authentication level. If you have configured a logout form and a logout URL as described in "Configuring a Single Sign-On Logout URL" on page 2-6, calling the SSO Logout URL removes the ObSSOCookie. This requires the user to re-authenticate the next time they access a resource protected by the Access System.

See "Configuring a Single Sign-On Logout URL" on page 2-6 for details.

Note: The logout.html form also contains javascript for removing the ObTemCookie set for the Identity System. It does not however, 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 form to remove these cookies.

Single Sign-On Between Identity and Access Systems

You can protect the Identity System with the Access System just as you would any other resource

When installing the Access System, you can indicate that you want to protect the Identity System applications with the Access System. This automatically creates two policy domains:

- A policy domain protecting the Access System applications starting with /access
- A policy domain protecting the Identity System applications starting with /identity

See the *Oracle Access Manager Installation Guide* for more information.

The rest of this section discusses the following topics:

Configuring Policy Domains for Single Sign-On

The Access System installation provides the option to configure policy domains automatically to protect Identity System applications, you can manually configure these policy domains at any time using the following guidelines.

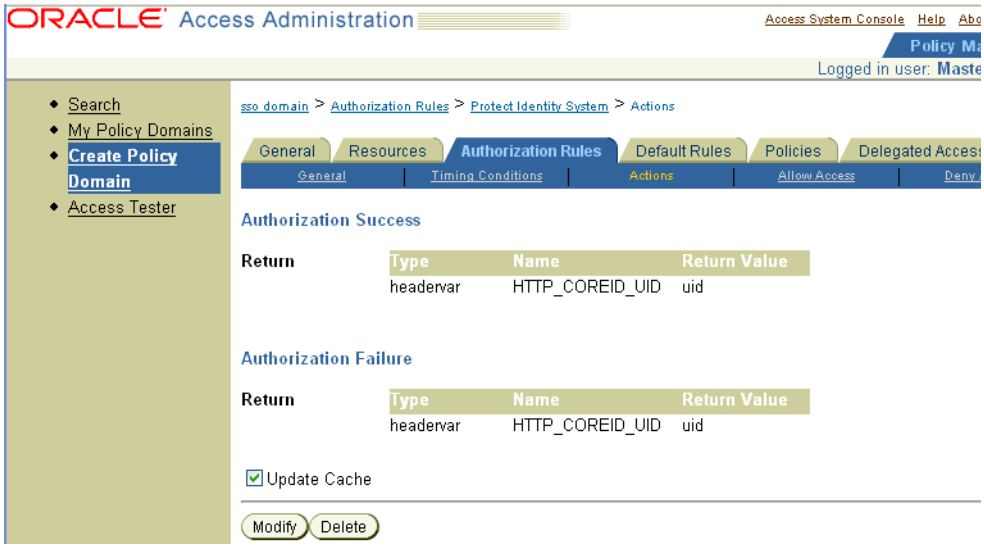
Note: These guidelines assume you are familiar with the process for creating policy domains. See Chapter 4, "Protecting Resources with Policy Domains" on page 4-1 for more information.

To create a policy domain that protects the Identity System applications

1. From the Policy Manager, create a new policy domain as described in Chapter 4, "Protecting Resources with Policy Domains" on page 4-1.
2. From the Resources tab, enter http as the resource type and enter /identity as the URL prefix.
3. From the Default Rules tab, create an authentication rule that protects the Identity applications using the challenge method of choice.

   The Oracle Access and Identity Basic Over LDAP authentication scheme includes the ability not to allow deactivated users access to the Identity System.
4. From the Default Rules tab, create an authorization rule that controls user access.

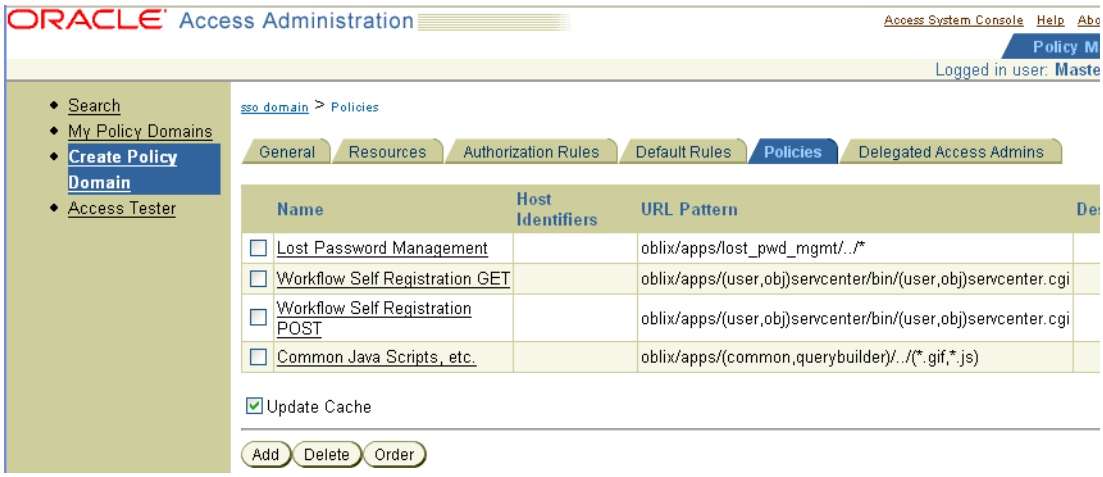
   Use the following as a guideline for configuring the authorization rule.

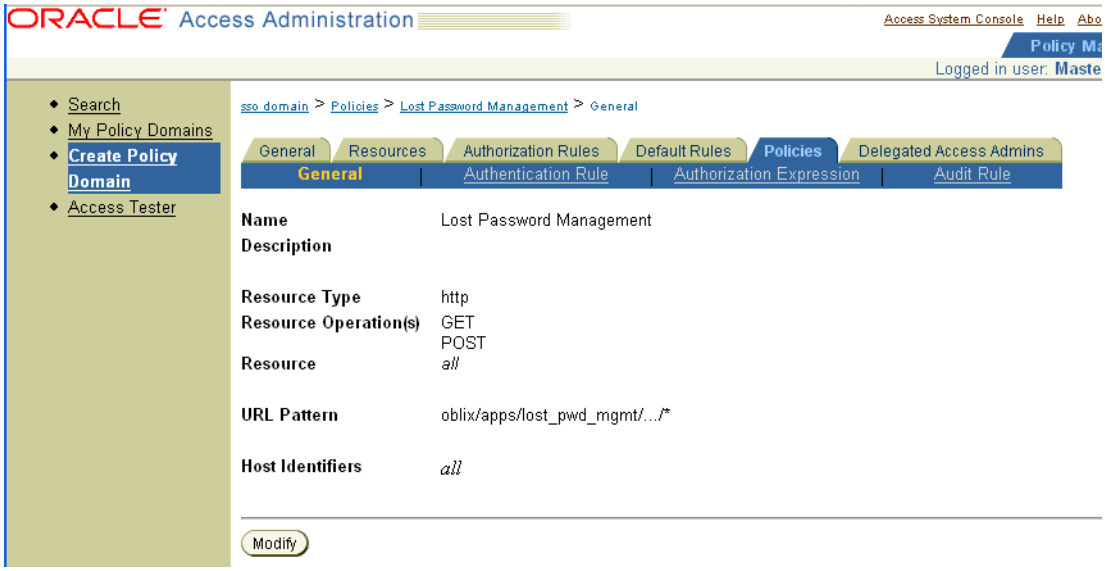


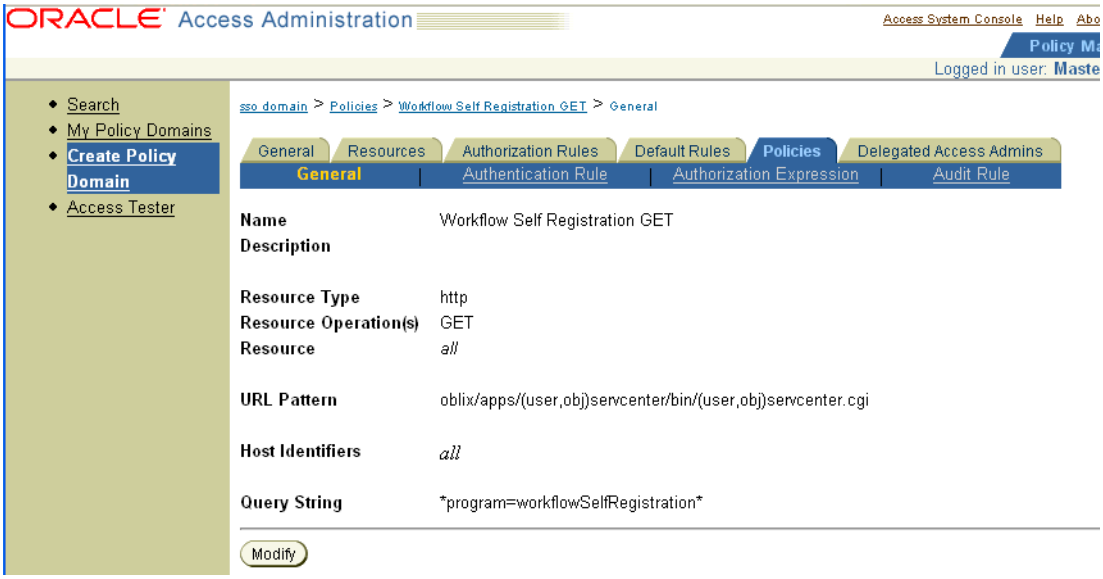
5. Next create the policies that allow access to key Identity System functionality such as Lost Password Management and Self Registration.

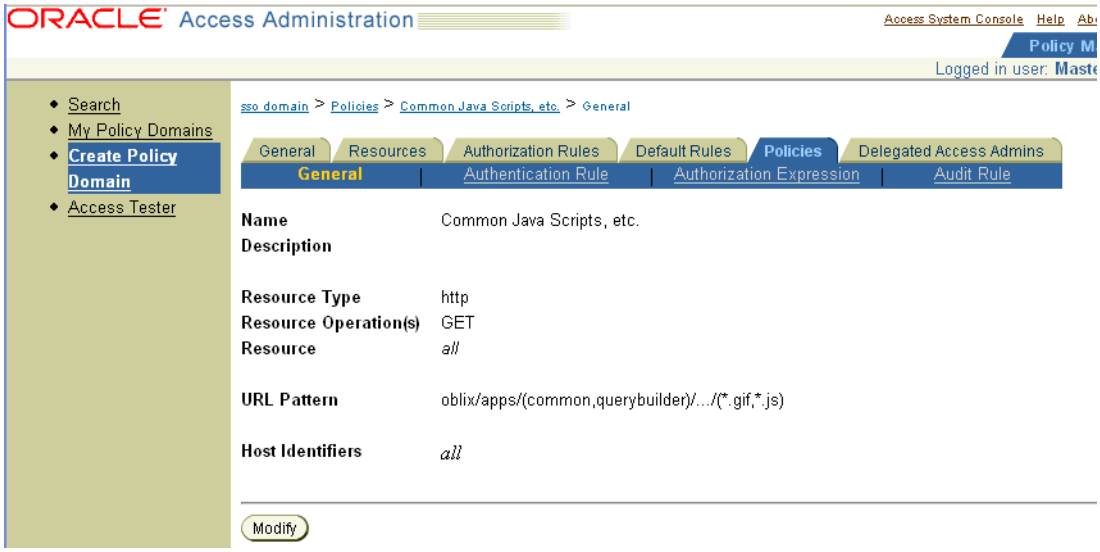
   The following four screens show a summary of the policies.

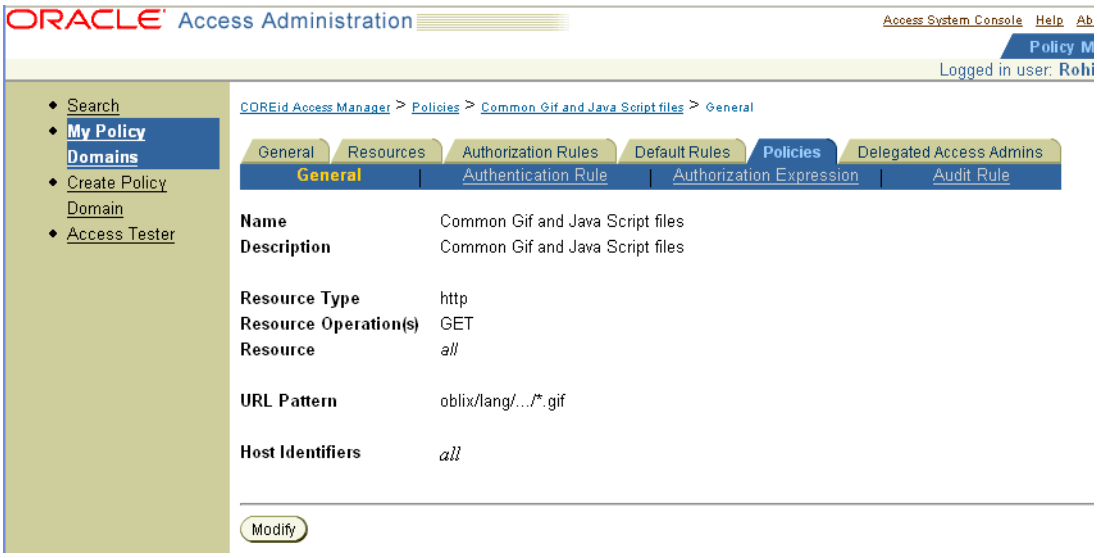
Note: For each policy, configure the Anonymous authentication scheme and configure users who are allowed or denied access.


ORACLE Access Administration
Access System Console Help Abo
Policy M
Logged in user: Maste
Search
My Policy Domains
Create Policy Domain
Access Tester
sso domain > Policies
General Resources Authorization Rules Default Rules Policies Delegated Access Admins
Name
Host Identifiers
URL Pattern
Lost Password Management
oblix/apps/lost_pwd_mgmt/.../*
Workflow Self Registration GET
oblix/apps/(user,obj)servcenter/bin/(user,obj)servcenter.cgi
Workflow Self Registration POST
oblix/apps/(user,obj)servcenter/bin/(user,obj)servcenter.cgi
Common Java Scripts, etc.
oblix/apps/(common,querybuilder)/../(*.gif,*.js)
Update Cache
Add Delete Order


ORACLE Access Administration
Access System Console Help Abo
Policy Ma
Logged in user: Maste
Search
My Policy Domains
Create Policy Domain
Access Tester
sso domain > Policies > Lost Password Management > General
General Resources Authorization Rules Default Rules Policies Delegated Access Admins
General | Authentication Rule | Authorization Expression | Audit Rule
Name Lost Password Management
Description
Resource Type http
Resource Operation(s) GET
POST
Resource all
URL Pattern oblix/apps/lost_pwd_mgmt/.../*
Host Identifiers all
Modify


ORACLE Access Administration
Access System Console Help Abo
Policy Ma
Logged in user: Maste
Search
My Policy Domains
Create Policy Domain
Access Tester
sso domain > Policies > Workflow Self Registration GET > General
General Resources Authorization Rules Default Rules Policies Delegated Access Admins
General | Authentication Rule | Authorization Expression | Audit Rule
Name Workflow Self Registration GET
Description
Resource Type http
Resource Operation(s) GET
Resource all
URL Pattern oblix/apps/(user,obj)servcenter/bin/(user,obj)servcenter.cgi
Host Identifiers all
Query String *program=workflowSelfRegistration*
Modify


ORACLE Access Administration
Access System Console Help Ab
Policy M
Logged in user: Maste
Search
My Policy Domains
Create Policy Domain
Access Tester
sso domain > Policies > Common Java Scripts, etc. > General
General Resources Authorization Rules Default Rules Policies Delegated Access Admins
General | Authentication Rule | Authorization Expression | Audit Rule
Name Common Java Scripts, etc.
Description
Resource Type http
Resource Operation(s) GET
Resource all
URL Pattern oblix/apps/(common,querybuilder)/.../(*.gif,*.js)
Host Identifiers all
Modify



To create a policy domain that protects the Access System applications

1. From the Policy Manager, create a new Policy Domain.
2. From the Resources tab, enter http as the resource type and enter /access as the URL prefix.
3. From the Default Rules tab, create an authentication rule that protects the Access System applications using the Challenge Method of choice.
4. From the Default Rules tab, create an authorization rule that allows/denies access to the appropriate users.
5. Add the same action as shown in step 4 from the previous section, "To create a policy domain that protects the Identity System applications" on page 7-15.
6. Create the policies that allow access to common Oracle Access Manager javascripts, gifs, and so on.
7. Configure the Anonymous authentication scheme and configure users who are allowed or denied access.

Displaying the Employee Type in the Top Navigation Bar

If single sign-on is enabled on the Identity System for connecting with another system (such as the Access System), you can use actions to define the user type in the header variables. The Access System picks up this user type and displays it if there is a correct corresponding value in the obnavigation.xml file. If no user type is set, the Access System uses the default defined in the obnavigation.xml file.

Troubleshooting SSO Between Identity and Access Systems

For information on troubleshooting, see "Troubleshooting Oracle Access Manager" on page E-1.

Enabling Impersonation in the Access System

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.

When running in a client's security context, a service becomes the client to an extent. One of the service's threads uses an access token (a protected object that represents the client's credentials) to obtain access to objects for the client.

The primary purpose of impersonation is to trigger access checks against a client's identity. The Access System overrides impersonation enabled with IIS. For details about enabling impersonation, see the *Oracle Access Manager Integration Guide*.

Troubleshooting Single Sign-On

For information on troubleshooting, see "Troubleshooting Oracle Access Manager" on page E-1.

Part III

Managing the Access System

Managing the Access System includes flushing password policy caches, as well as running diagnostics, and managing user access privilege reports and sync reports from within the Access System Console. Additionally, you can perform some tasks outside the Access System Console using Access System configuration files.

Part III contains the following chapters:

8

Access System Management

This chapter discusses several additional Access System configuration and management functions available within the Access System Console. Topics include:

- Prerequisites
- About Access System Configuration and Management
- Configuring User Access
- Creating a Shared Secret Key
- Flushing Password Policy Caches
- Running Diagnostics
- Managing User Access Privilege Reports
- Managing Sync Records
- Detecting and Restoring Corrupted Sync Records in Environments with Multiple Directory Servers

For more information about managing the Access System, see:

- Chapter 2, "Configuring Access Administrators and Server Settings" on page 2-1
- Chapter 9, "Managing Access System Configuration Files" on page 9-1

Prerequisites

Oracle Access Manager should be installed and set up, as described in the *Oracle Access Manager Installation Guide*. Read the *Oracle Access Manager Introduction*, which provides an overview of Oracle Access Manager not found in other manuals. Also, familiarize yourself with the *Oracle Access Manager Identity and Common Administration Guide*, which provides a brief review of Access System applications and installation; introduces Access System configuration and administration; and includes common functions, configuration, and administration.

About Access System Configuration and Management

Earlier chapters in this manual describe configuring administrators and viewing server settings through the Access System Console, System Configuration functions. That information is not repeated here.

The rest of this section discusses the following topics:

- Access System Configuration

- System Management

Access System Configuration

Numerous functions are available in the Access System Console, Access System Configuration tab, as shown in the following list. Unless indicated, other chapters in this manual describe Access System Configuration functions:

- Access Server Clusters: View existing Access Server Clusters, add new and modify existing Access Server Clusters, configure and delete Access Server Clusters.
- AccessGate Configuration: View existing AccessGates, add new and modify existing AccessGates, configure and delete AccessGates.
- Access Server Configuration: View existing Access Servers, add new and modify existing Access Servers, configure cache and audit settings.
- Authentication Management: Configure Authentication Rules.
- Authorization Management: Configure Authorization Rules.
- User Access Configuration: List revoked users, flush the user cache, as described in this chapter under "Configuring User Access" on page 8-3.
- Common Information Configuration: Generate a cryptographic key to encrypt cookies (covered here), configure a master auditing rule, manage resource type definitions, flush the Password Policy Cache (covered here), handle duplicate action headers. For more information on items covered here, see:
  - "Creating a Shared Secret Key" on page 8-4.
  - "Flushing Password Policy Caches" on page 8-6.
- Host Identifiers: Configure host identifiers.

System Management

There are several options available in the Access System Console to perform system management operations, which are described in this chapter:

- Diagnostics: Show Access Server details, including connection information, as described in "Running Diagnostics" on page 8-7.
- Manage Reports: Create, view, modify, and execute User Access Privilege Reports, as described in "Managing User Access Privilege Reports" on page 8-7.
- Manage Sync Records: Archive or purge Sync Records, as described in "Managing Sync Records" on page 8-9.

  See Also: "Detecting and Restoring Corrupted Sync Records in Environments with Multiple Directory Servers" on page 8-12 for details about using a command-line tool for detecting and recovering from sync record corruption when you have multiple Access Servers writing to multiple Directory Servers

For information about diagnostics, auditing, reports, and logging, see the *Oracle Access Manager Identity and Common Administration Guide*.

Configuring User Access

You use the User Access Configuration function available through the Access System Console, Access System Configuration tab, to manage revoked users and flush user data from the cache. This section covers the following topics:

- Revoking Users
- Flushing User Data from the Cache

> **Note:** You must be a Master Access Administrator or a Delegated Access Administrator with appropriate permissions to configure user access.

For more information on caches, see Flushing Password Policy Caches on page 8-6 and "Automatic Access System Cache Flush" on page 9-1. See also the *Oracle Access Manager Deployment Guide*.

Revoking Users

You can create and modify a list of users who are prohibited from accessing any of your resources. This list supersedes any other policies controlling user access to your resources. Once a user has been revoked, if the user tries to refresh the browser, or go to another protected resource, they are denied access. If a revoked user tries to log in, he or she is presented with the following error:

```
The user corresponding to the credentials (userid=xxxxxxxx
password=(omitted) Resource=/access/oblix RequesterIP=
nn.nn.nnn.nn HostTarget=http://hostname:port Operation=GET) used
in the login has been revoked by an administrator of the Access
System.
```

To create the revoked user list

1. In the Access System Console, click Access System Configuration, then click User Access Configuration.

   The User Access Configuration screen appears.

2. Click Revoked Users.

   The Modify User Revocation List screen appears, displaying the names of revoked users. If no revoked users exist, the Configure User Revocation List screen appears. If any exist, their names appear beneath the Revoked Users link.

3. Click Select User, then use the Selector feature (Select User button) to add or remove revoked users.

   See the *Oracle Access Manager Identity and Common Administration Guide* for instructions on using the Selector.

4. Click Save to save your changes (or click Cancel to exit without saving.

Flushing User Data from the Cache

You must be a Master Access Administrator to flush user data from the cache.

Flushing the user cache serves to automatically update changes in user access rights. In this case, information about selected users is removed from AccessGate and Access Server caches. For example, you might want to flush a user's information after that

user's rights to view or modify an attribute have changed. This is useful when a user needs immediate access to resources.

See the *Oracle Access Manager Identity and Common Administration Guide* for instructions on using the Selector.

To flush user information from the cache

1. From the Access System Console, click Access System Configuration, select User Access Configuration, then click Flush User Cache.

   The Flush all cached information for specified users page appears.

2. Perform the following steps to identify individuals and add them to a list of users whose information will be flushed from all caches.

   a. Select the criteria for your search from the first two lists. For example:

      Full Name

      That Contains

   b. Fill in the empty field to narrow the list of named that will be returned, and then click Go. For example:

      sch

   c. In the resulting list of names, click the Add button beside each name to be included in the cache flush operation.

      The names of people that you have selected and added appear on the right side of the page.

   d. Repeat steps a through c to finish building your list.

   e. Click Done when you have the list that you want.

      The Flush all cached information for specified users page returns with the list of selected names.

3. Click the Flush Cache button (or click Cancel to terminate the operation).

   You are prompted to confirm your decision. If you click OK the names are cleared from the page, and information about these users is flushed from AccessGate and Access Server caches.

4. Click OK to clear these names (or Cancel to terminate without clearing).

Creating a Shared Secret Key

You use the Shared Secret function available through the Access System Configuration, Common Information Configuration tab, to generate a key that encrypts single sign-on cookies sent from an AccessGate to a browser.

Note: You must be a Master Access Administrator to create a shared secret key. You should generate a cryptographic key as soon as possible after installing Oracle Access Manager, otherwise a less secure default is used.

AES is a new encryption scheme introduced in Oracle Access Manager 7.0. If you have a new installation of Oracle Access Manager 10.1.4, AES is the default encryption

scheme. RC6 encryption is deprecated in Oracle Access Manager 10.1.4, and its support will be removed in future releases.

If you have upgraded to Oracle Access Manager 10.1.4 from an older version, the older encryption scheme will be retained. Older WebGates may co-exist with newer WebGates as described in the *Oracle Access Manager Upgrade Guide*:

- Use RC4 as the encryption scheme if you have version 5.x and version 7.x WebGates co-existing together.
- Use RC6 as the encryption scheme if you have version 6.x and version 7.x WebGates co-existing together.
- Use AES encryption only when all the WebGates and Access Servers are upgraded to Oracle Access Manager version 7.0 and higher

Note: If the shared secret is generated more frequently than the session timeout, then the user may have a cookie that was encrypted using a shared secret that is more than two generations old. In this case, the cookie is rejected and the user is forced to re-authenticate.

To generate a cryptographic key

1. In the Access System Console, click Access System Configuration, click Common Information Configuration.

   The Common Information Configuration screen appears.

2. Click the Shared Secret tab at the top of the screen.

   The Generate shared secret screen appears.

3. Click Modify.

   The Generate shared secret page now includes various ciphers from which to choose.

4. Select the appropriate cipher option for the shared secret (Oracle recommends using the AES cipher).

5. Click Generate Secret only once.

   Oracle Access Manager generates a new cryptographic key and distributes it to each Access Server on your system. The new key replaces the existing key without disrupting service to end users. Re-authentication only happens when the session times out. This process is called grandfathering. Clicking Generate Secret multiple times can put the shared secret key in Identity out of synch with the key in the Policy Manager.

   A message informs you the operation was successful.

Changes to the Shared Secret Key

If you change the shared secret during a user session, the user does not need to re-authenticate. If a cookie is being decrypted with the old shared secret and the cookie is refreshed, it is encrypted with the new shared secret.

If the shared secret is changed more frequently than one-fourth the setting of the idle session timeout parameter, users may have to re-authenticate during a session. Otherwise, user are not required to re-authenticate during a session if the shared secret is changed.

Flushing Password Policy Caches

You use the Flush Password Policy Cache function (Access System Configuration tab, Common Information Configuration), to flush all password policies from the Access Server cache. Flushing the password policy cache removes existing password policies and adds newly configured policies.

Flushing the password policy cache serves to automatically update the password policy cached in the Access Server by removing existing password policies and adding newly configured policies. This is useful when you change the password policy from the Identity System and want the changes to be reflected when the Access Server evaluates password policies.

From the Flush Password Policy Cache tab you can also flush all redirect URLs used for password policy enforcement if you have configured redirect URLs. Flushing the redirect URLs serves to automatically update the password policy management redirect URLs whenever they are updated in the Identity System. For the Access System to recognize these changes, the administrator must flush the cache.

You can also flush all cached information for the specified lost password management policy. After clicking the Flush Password Policy Cache tab, a section labeled Flush all cached information for specified lost password management policy is available. A list enables you to choose the lost password policy to be flushed from the cache. If there are none, this is stated on the page. This is useful when you change the lost password policy from the Identity System and want the changes to be reflected when the Access Server evaluates password policies.

Note: You must be a Master Access Administrator to flush password policy caches. You can also automatically update this cache. For more information about updates to the Access Server cache, see the *Oracle Access Manager Identity and Common Administration Guide*.

To flush password policy caches and redirect URLs

1. Click Access System Console, click the Access System Configuration tab, select Common Information Configuration, then click the Flush Password Policy Cache tab.
2. **Flush all cached information for specified password policy**: Perform the following steps.
   a. From the list under the label (Flush all cached information for specified password policy), select the name of the policy that you want to flush from the cache.
   b. Click the Flush Cache button.
   c. Click OK to confirm your decision
3. **Flush all redirect URLs used for password policy enforcement**: Click the Flush Redirect URL button.
4. **Flush all cached information for specified lost password management policy**: Perform the following steps.
   a. From the list under the label (flush all cached information for specified password policy, select the name of the policy that you want to flush from the cache.
   b. Click the Flush Cache button.

c. Click OK to confirm your decision.

Running Diagnostics

You use the Diagnostics item on the Access System Console, System Management page to run diagnostics on all the Access Servers in your Oracle Access Manager system or selected servers.

To run diagnostics for Access Servers

1. From the Access System Console, select System Management, then click Diagnostics.

   You are asked to select the Access Servers on which you would like to run diagnostics

2. Select the option you want:
   - All Access Servers: Select All Access Servers, then click the Go button.
   - Specific Access: Servers Hold down the Control key, then click the names of the servers whose details you want, then click the Go button.

Managing User Access Privilege Reports

You use the Manage Reports function on the Access System Console, System Management page to manage user access privilege reports.

Each Access Server can collect audit information about the resource requests it handles. The list of existing reports is visible from the Manage Reports page. In addition, you can perform the following operations:

- Adding a Report
- Managing Reports

For more information on auditing and reports, see the *Oracle Access Manager Identity and Common Administration Guide*.

Adding a Report

You can create user access privilege reports that verify whether specific users have access to specific resources at specific times. Explanations to help you complete these fields appear in the following procedure.

To add a user access privilege report

1. From the Access System Console, select System Management, then click Manage Reports.
2. On the Manage User Access Privilege Reports page, click the Add button.
3. Complete the information as follows:

   Report Name: Choose a self-explanatory name for the audit report.

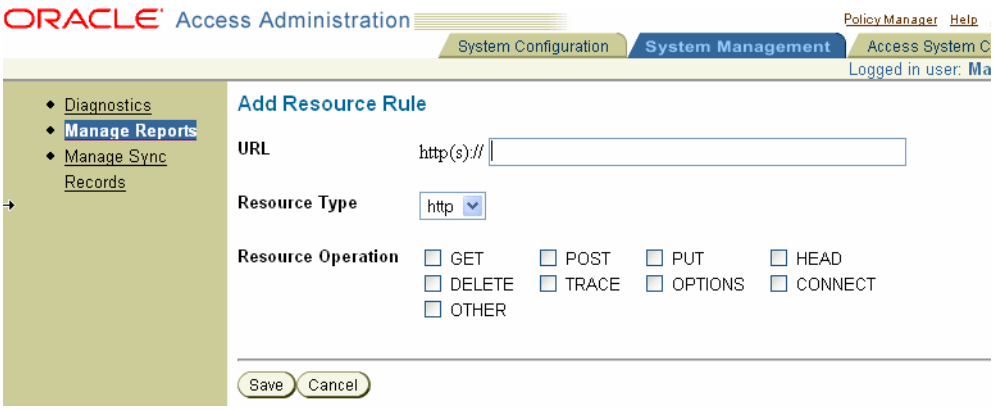
   Description: If you wish, you may describe the report.

   Access Server: Name of the Access Server that will collect the information for the report.

   Results Storage: Indicate whether the audit data should go to a disk file or a database.

- **Store in File**: Check the box beside this option, then specify the fully-qualified path and file name in the Name of File field.
- **Store in Database**: See the *Oracle Access Manager Identity and Common Administration Guide* for specific Oracle Access Manager configuration details and the *Oracle Access Manager Installation Guide* for information on database support.

List of Resources: Click the Add button beside this option to display the Add Resource Rule page, as shown in the following screen shot.



Note: You may add multiple resources to a report. Access information on each resource will be returned in the report.

---

4. On the Add Resource Rule page, complete the rule by specifying the following, then click Save to return to the Add New Report page:
   - **URL**: The URL of a target resource you want to add to the report.
   - **Resource Type**: Supported choices are HTTP and EJB.
   - **Resource Operation**: Check boxes appear beside operations you can include in the report. Oracle Access Manager will determine which are permitted against the specified resources, for the specified users, at the specified time.
5. On the Add New Report page, continue specifying the following information:

   From this IP Address: *Optional*. The IP address of the computer hosting the client browser whose access you want to test. This parameter is optional.

   Date/Time of Access: Select a button to determine when a specific resource will be available to the users specified by the current report:

   - Any: Oracle Access Manager will determine if there is at least some point in time when the resource is available.
   - Specific date and time: Indicates you want to identify a specific point in time so that Oracle Access Manager can determine if access is permitted at that particular moment.

   Check Access for the following users: Specify whether to check the access for all users in the directory or only those you designate.

   - selected users: Enables you to use the Selector page to locate and add specific users. Choose selected users, click the Select User button, specify your search criteria, then add specific users.

- all users: Indicates you want to check the access of all users in the directory.

6. Click Save on the Add Reports page to save the specifications for the report and display the name you specified as a link on the Manage User Access Privilege Reports page.

Managing Reports

From the Manage User Access Privilege Reports page (Access System Console, select System Management, then click Manage Reports), you can perform several operations:

- Add: Create a new report as described in "Adding a Report" on page 8-7.
- Delete: Check the box beside the report name on the Manage User Access Privilege Reports page, then click the Delete button to remove the report. Confirm that you want to delete the report when asked.

> **Note:** To delete or execute multiple reports simultaneously, check all the boxes on which to operate, then click the appropriate button.

- Execute: Check the box beside the report name on the Manage User Access Privilege Reports page, then click the Execute button. Confirm that you want to execute the report when asked.
- Refresh: Update the list of reports on the Manage User Access Privilege Reports page by clicking the Refresh button.
- Modify: Click a link on the Manage User Access Privilege Reports page to display the Manage Existing Report page, then change the parameters for the existing static audit report. See "Adding a Report" on page 8-7 for details about each option.

Managing Sync Records

This section includes the following topics:

About Sync Records

An Oracle Access Manager deployment can include multiple instances of the Identity Server, Access Server, and Policy Manager. Changes to a user profile or policy information can be made using the Identity System Console or Policy Manager. Configuration changes for Access Servers are made using the Access System Console. All changes are written to the directory server and must be propagated across all components to ensure an integrated and unified view throughout the system to ensure behavioral consistency.

Each change to a user profile or policy information or configuration details results in the generation of a corresponding global synchronization record (sync record). The sync record is stored in the directory server so that it can be shared between all components.

Following is an example of a global sync record for a policy:

```
obSyncRequestNo=<GSN>, cn=PSCMgmt,obapp=PSC,o=Oblix,<config tree>
```

Here, *GSN* refers to the global sync number that is used by the Access Server to track changes to policy information or user information, and to synchronize these changes with data in the directory server and update the directory server with the same information for all instances in the deployment.

A user profile sync record looks like the following example, which is similar to the sync record for a policy except that the cn is different:

```
obSyncRequestNo=<GSN>, cn=UserMgmt,obapp=PSC,o=Oblix,<config tree>
```

A change to a user profile in the Identity System Console, involves one Identity Server. The resulting user related sync record would look like the following example:

```
Dn: obSyncRequestNo=14,cn=UserMgmt,obapp=PSC,o=Oblix,o=company,c=us
     1> obSyncRequestNo: 2;
     1> obCompID: 20080526T17521689728;
     1> obSyncChangeType: 2;
     1> obSyncRequestType: 3;
     1> obSyncTime: 1211804638;
     2> objectClass: top; oblixSyncRecord;
     1> obver: 10.1.4.0;
```

Each line of the sync record provides different information about the change. Included are the sync request number (`obSyncRequestNo`); the component from which this sync request came (`obCompID`); the type of change (`ObSyncChangeType` refers to whether this is an addition, update, or delete); the request type value (`obSyncRequestType`); the time of the change (`obSyncTime`); the object class (`objectClass: top; oblixSyncRecord`); and the version of the Oracle Access Manager release (`obver: 10.1.4.0`). The policy identifier (`obCompSDID`); is not observed in a user-related sync record. For more information about these attributes and values, see the *Oracle Access Manager Schema Description*.

Over time, sync records accumulate. You can manage the space these records consume on the directory server by periodically archiving, or purging, all the records prior to a specified date.

Note: With Active Directory, modification of the user profile can be monitored with a value of `UserMgmtNodeEnabled` (`False`) in the Access Server Sever globalparams.xml file. For more information, see "Cache Flush Issues with Active Directory" on page E-5.

For additional information, see:

- Archiving Sync Records
- Purging Sync Records

See Also: "Detecting and Restoring Corrupted Sync Records in Environments with Multiple Directory Servers" on page 8-12

Archiving Sync Records

You must be a Master Access Administrator to archive sync records.

When you archive sync records, information is stored in an ldif file. The file is typically named *nnn*.ldif, where *nnn* is a string of numbers in the GSN sync records format: syncrecordsxxxxxxxxxx.YYYYMMDD.HHMMSS. This format assigns a unique sync

record number. xxxxxxxxxx is a unique number assigned by the system. The time is Universal Time Coordinated (UTC) time. The following file was created July 29, 2008. The cut off time is represented in hours:minutes:seconds (04:08:44), based on a 24-hour clock.

```
syncrecords1090998000.20080729.040844.ldif
```

The name represents a unique identifier, as well as the moment at which the file was created and the cut-off time for archiving or purging records. All records created prior to the cut-off time that you select will be archived or purged.

By default, the archived ldif file is stored in:

PolicyManager_install_dir\access\oblix\data\common

where *PolicyManager_install_dir* represents the directory where you installed the Policy Manager.

During this task, you will select a month, day, and year from lists provided. All records created prior to the date that you select will be archived.When you choose to archive sync records, the location where the ldif file containing the records is stored will be presented in a message for you to record. For example:

```
Successfully archived 210 sync records generated before the selected date to file
/export/home/COREid1014/webcomponent/access/oblix/data/common/syncrecords109099
8000.20080729.040844.ldif
```

Later on, you could review an archived (also known as an exported) .ldif file if needed. There is no need to import these sync records.

To archive sync records

1. From the Access System Console, click System Management.
2. In the left navigation pane, click Manage Sync Records.
3. On the Manage Sync Records page, choose a month, day, and year from the lists provided to specify the date of sync records generated.
4. Click the Archive Sync Records button.
5. When asked if you really want to archive the records, click OK to execute the action (or Cancel to revoke the operation).
6. Record the location when it is presented in a message. For example:

   ```
   /export/home/COREid1014/webcomponent/access/oblix/data/common/syncrecords109099
   8000.20080729.040844.ldif
   ```

Purging Sync Records

You must be a Master Access Administrator to purge sync records.

During this task, you will select a month, day, and year from lists provided. All records created prior to the date that you select will be purged. Purged records are deleted.

> **Note:** Oracle recommends that you archive sync records before you purge them. Later on, you could review an archived (also known as an exported) .ldif file if needed. There is no need to import these sync records.

To purge sync records

1. From the Access System Console, click System Management.
2. In the left navigation pane, click Manage Sync Records.
3. On the Manage Sync Records page, choose a month, day, and year from the lists provided to specify the date of sync records generated.
4. Click the Purge Sync Records button.
5. When asked if you really want to purge the records, click OK to execute the action (or Cancel to revoke the operation).

Detecting and Restoring Corrupted Sync Records in Environments with Multiple Directory Servers

If your environment does not include multiple Access Servers writing to multiple directory servers, you can skip this section. This section provides the following topics and tasks:

About Cache Flush and Sync Records with Multiple Directory Servers

Oracle Access Manager stores three types of information in the directory server:

- User data, which can be distributed across multiple directory server instances
- Configuration data, which cannot be distributed across multiple directory server instances

> **Note:** You might have replication configured wherein two master directory servers include the same configuration information, which is described later in this topic.

- Policy data, which cannot be distributed across multiple directory server instances

Any modification to a user profile or policy information is updated in the directory server. These modifications require an Access Server cache flush to ensure that authentication and authorization information is up to date. Enabling the Update Cache feature in Oracle Access Manager forces a cache flush every time an entry is written to the directory server. If you do not select Update Cache, the Access Server caches are updated when they time out and read new information from the directory server.

Whenever a cache flush request is received by the Access Server (every time an entry is written to the directory server when the Update Cache feature is turned on or on time out and read operations if the Update Cache feature is off), the Access Server:

- Fetches the latest oblixGSN entry in which GSN is maintained from the directory server (purging sync records does not impact this).
- Checks the oblixGSN objectclass, which is used in the cache flush mechanism
- Checks the global sequence number (a value in the obSeqNo attribute within the oblixGSN objectclass) that represents the last flush request number
- Increments the global sequence number (obSeqNo value) and stores it in the sync record (oblixSynchRecord)
- Updates the sync record with cache flush information, including `obSyncRequestNo`, `ObSyncChangedType`, and `obSyncTime`

See Also: "About Sync Records" on page 8-9

For a successful cache flush operation, the sync record in the directory server must be unique, obSeqNo must have a single unique value, and the oblixGSN objectclass should have a single obSeqNo. For example:

```
obSeqNo=15,cn=obapp=PSC,o=Oblix,<config tree>
```

GSN information is stored in the configuration data in a single directory server instance. GSN information refers to the entry for the oblixGSN objectclass, which contains the global sequence number (`obSeqNo`) value.

Note: If you have replication configured for a high availability environment, you might have two master directory server instances containing the same configuration information. In such cases, the Access Server updates information in the directory server and the directory servers periodically synchronize their contents with each other.

When you have multiple Access Servers writing to multiple directory servers, you will have a sync record for a particular entry or change within each directory server. Both the sync record (`oblixSynchRecord`) and global sequence number (`obSeqNo`) should be the same. However, these records can get out of sync due to the time lag between directory server synchronizations.

A corrupted entry might have multiple obSeqNo values or multiple entries under a single obapp=PSC,o=oblix,<config tree> node. For example:

```
obSeqNo=101,obSeqNo=102,obapp=PSC,o=Oblix,<config tree>
```

or

```
obSeqNo=101,obapp=PSC,o=Oblix,<config tree>
obSeqNo=102,obapp=PSC,o=Oblix,<config tree>
```

When corruption occurs, you will see inconsistent performance between Access Servers. Therefore, you need a way to detect the problem and recover. Recovery requires removal of corrupted entries from the directory server.

Oracle Access Manager enables you to detect sync record corruption in the directory server and recover from it using a new command-line tool. For more information, see "About Detection and Recovery".

About Detection and Recovery

A command-line tool named recovergsncorruption that you can use for detection and recovery from sync record corruption is provided and stored in the following path:

PolicyManager_install_dir\access\oblix\tools\recovergsncorruption\

All parameters that are needed during the recovery process can be included as arguments from the command line or can be provided through a script. If you do not provide parameters as arguments on the command line, you will be prompted for the values one by one.

Table 8–1 recovergsncorruption Parameters and Values

Parameter	Value	Description
-i	*PolicyManager_install_dir*	Provides the installation path for the recovergsncorruption utility
-isCacheFlushDisabled	yes	Provides your confirmation that you have performed prerequisite tasks in the following topics to prepare for recovery. Use this when you want to perform recovery. Prerequisites include: ■ "Disabling Access Server Cache Flush and GSN Updates" on page 8-18 ■ "Blocking Updates from the Policy Manager and Applications using AMAPI" on page 8-18
-recoverCorruption	no	Launches corruption detection without recovery, as follows: ■ Detects corruption ■ Reports findings (corruption versus no corruption) ■ Logs progress in a file named recovergsncorruption.log in the following path *PolicyManager_install_dir*\access\oblix\tools\recovergsncorruption\recovergsncorruption.log ■ No archive is created. ■ No modifications are made to any sync records or GSN.

Table 8–1 (Cont.) recovergsncorruption Parameters and Values

Parameter	Value	Description
-recoverCorruption	yes Note: Before recovery, you must perform preparation tasks to ensure that cache flush operations are halted. **See Also**: isCacheFlushDisabled Yes	Launches corruption detection and recovery, as follows: ■ Archives existing sync records in a .ldif file in the directory server ■ Displays the name and path for the archived .ldif for possible use later ■ Starts detection and recovery, including modifying sync records and GSN values in the directory server ■ Logs progress in a file named recovergsncorruption.log in the following path *PolicyManager_install_dir*\access\oblix\tools\recovergsncorruption\recovergsncorruption.log ■ Confirms recovery success or failure

You can also write a script that uses this tool. The syntax in the script will be similar to that when you are running the utility from the command-line. For example:

To detect corruption:

```
recovergsncorruption -i PolicyManager_install_dir -isCacheFlushDisabled yes
-recoverCorruption no
```

To detect and recovery from corruption:

```
recovergsncorruption -i PolicyManager_install_dir -isCacheFlushDisabled yes
-recoverCorruption yes
```

In either case, before recovery you must ensure that cache flush operations are halted to prevent any possible problems during recovery when sync records are updated.

See Also:

- "Disabling Access Server Cache Flush and GSN Updates" on page 8-18
- "Blocking Updates from the Policy Manager and Applications using AMAPI" on page 8-18

During recovery, the following processes are performed using the original sync records in the directory server.

Process overview: Recovery processing

- Duplicate obSeqNo values in a single oblixGSN entry are removed from the directory server (oblixGSN entry will have a single unique ObSeqNo value)
- Duplicate entries for the oblixGSN objectclass are removed from the directory server
- Sync records that contains the deleted obSeqNo are updated with a new obSeqNo value

- obSeqNo is updated: After updating the sync records, obSeqNo is updated with the largest ObSyncRequestNo in the oblixGSN entry
- Progress is logged in a file named recovergsncorruption.log in *PolicyManager_install_dir*\access\oblix\tools\recovergsncorruption\recovergsncorruption.log

If recovery is successful, you are notified. In this case, there will be no duplicate entries for the oblixGSN objectclass in the directory server. The oblixGSN objectclass will have a single unique ObSeqNo value; duplicate values are removed and ObSeqNo is updated (incremented). There are no duplicate sync records. Sync records containing deleted values are removed. For more information, see "Recovering from Sync Record Corruption" on page 8-20.

If recovery is not successful, you need to perform steps to revert to the earlier status of the directory server (before recovery was attempted. For more information, see "Correcting Errors that Occurred During the Recovery Process" on page 8-21.

Task overview: Detecting and restoring corrupted sync records with multiple directory servers

1. Review the following topics:
   - About Sync Records on page 8-9
   - About Cache Flush and Sync Records with Multiple Directory Servers on page 8-12
   - The rest of this task overview
2. **Detection**: Perform tasks in "Detecting Sync Record Corruption" on page 8-16.
3. **Recovery**: Perform tasks as described in the following topics:
   a. Disabling Access Server Cache Flush and GSN Updates
   b. Blocking Updates from the Policy Manager and Applications using AMAPI
   c. Recovering from Sync Record Corruption
   d. Correcting Errors that Occurred During the Recovery Process
   e. Restoring Cache Flush Operations

Detecting Sync Record Corruption

You perform this task when Access Server behavior is inconsistent. You must be a Master Access Administrator to perform this task.

The command is entered on a single line without breaks. The example in this procedure appears to be entered on multiple lines due to formatting constraints for the book.

To detect sync record corruption

1. From the computer hosting the Policy Manager, locate the recovergsncorruption command-line tool in the following path:

   PolicyManager_install_dir\access\oblix\tools\recovergsncorruption\

2. Start the detection process by providing the following command-line arguments:

```
recovergsncorruption -i PolicyManager_install_dir -isCacheFlushDisabled yes
-recoverCorruption no
```

Alternatively, you can run the tool from the command line with no options and let the tool prompt you for the parameter values. For example, recovergsncorruption prompts you as follows:

```
Please enter the Policy Manager installation directory:
Identity and Policy cache flush disabled? [yes|no]:
Do you want recovery if GSN corruption is detected? [yes|no]:
```

3. Review the resulting message to determine how to proceed:
   - **No Sync Record Corruption was Detected**: You are finished.
   - **Sync Record Corruption was Detected**: Proceed to "Disabling Access Server Cache Flush and GSN Updates"

Checking the Log After Detection or Recovery

You perform this task after detecting or recovering from GSN corruption. The log file is created for your reference. It contains information related to the corruption detection or the corruption recovery process; information related to the archived .ldif file; details related to detection or recovery operation success or failure. Also, if the recovery operation fails, the reason for the failure is identified.

The log file is named as described in "About Detection and Recovery" on page 8-14 contains the following information:

Each time the recovergsncorruption command-line tool is run, log messages are appended to existing information in the log file. You can delete the log file at any time.

The log file contains the name of generated .ldif file that can be imported later, if desired. A typical log file might look like the following example:

```
############################## Starting ##############################.
Writing Sync records in:
D:/install_dir/access/oblix/data/common/GSNsyncrecords20080630.121814.ldif
Successfully ported the sync records
No corruption was detected under: obapp=PSC,o=Oblix,o=company,c=us
Recovery of policy Sync records completed successfully
Corruption was detected under: cn=UserMgmt,obapp=PSC,o=Oblix,o=company,c=us

Deleting following entry
obSeqNo=3,cn=UserMgmt,obapp=PSC,o=Oblix,o=company,c=us
Deleted successfully
Deleting following entry
obSeqNo=2,cn=UserMgmt,obapp=PSC,o=Oblix,o=company,c=us
Deleted successfully
Adding GSN entry
obSeqNo=3,cn=UserMgmt,obapp=PSC,o=Oblix,o=company,c=us
Recovery of user sync records completed successfully
############################## Finished ##############################.
```

To check the log file after GSN corruption detection or recovery

1. On the computer where you performed detection or recovery, locate recovergsncorruption.log in the following path:

   PolicyManager_install_dir\access\oblix\tools\recovergsncorruption\

2. Review the messages to see what was detected, removed, added, or if there was any failure and why.
3. If the logged details, including references to sync records created by the tool, are not required, you can delete the log.

Disabling Access Server Cache Flush and GSN Updates

You perform this task after sync record corruption is detected. You must be a Master Access Administrator to perform this task.

To disable Access Server cache flush, you must set the '`doAccessServerFlush`' parameter to `false` in the basedbparams.xml file of each Identity Server. This prevents the Access Manager SDK from sending requests to flush the Access Server cache. For more information, see the topic on "Configuring the Access Manager SDK for the Identity System" in the *Oracle Access Manager Identity and Common Administration Guide*.

After a successful recovery, you can undo this operation, as described in "Restoring Cache Flush Operations" on page 8-21.

To disable cache flush between the Identity Server and Access Server

1. Locate the basedbparams.xml file in the following Identity Server path:

   IdentityServer_install_dir/identity/oblix/data/common/basedbparams.xml

2. In the basedbparams.xml file, locate the `doAccessServerFlush` parameter and set it to `false`.

   ```
   <NameValPair ParamName="doAccessServerFlush" Value="false" />
   ```

3. Save the file.
4. Restart the Identity Server.
5. Restart the Access Server.
6. Repeat these steps for all Identity Servers and restart servers one by one.
7. Proceed to "Blocking Updates from the Policy Manager and Applications using AMAPI".

Blocking Updates from the Policy Manager and Applications using AMAPI

After disabling cache flush operations between the Identity Server and Access Server, you need to block updates that occur automatically based on settings defined in the System Console. You do this by disabling the Update Cache option in the System Console.

Update Cache: This option must be turned off on the following pages:

- Identity System Console: There are no Update Cache options on pages in the Identity System Console
- Policy Manager: Nearly all pages related to a policy domain provide an Update Cache option. See the:
  - My Policy Domains page
  - Pages under the Resources tab
  - Pages under the Authorization Rules tab
  - Pages under the Default Rules tab
  - Pages under the Policies tab
- Access System Console: Several pages under the Access System Configuration tab include an Update Cache option that must be disabled. See:

- Authentication Management page and individual authentication scheme tabs: General, Plugins, Steps, and Authentication Flow
- Common Information Configuration, Master Audit Rule tab
- Host Identifiers, List all host identifiers page

Policy Manager API: In addition to turning off the Update Cache options, you must ensure that the Policy Manager API is off.

> **Note:** Policy Manager API and Access Management Service are used interchangeably in the System Console.

You turn Policy Manager API functionality off in the following Access System Configuration pages:

- Access Server Configuration
- Access Server Clusters
- AccessGate Configuration

After a successful recovery, you can undo this operation, as described in "Restoring Cache Flush Operations" on page 8-21.

To block updates from the Policy Manager and applications

1. From the Access System Console, click the Access System Configuration tab, click Access Server Configuration, and perform the following steps:
   a. Click the name of an Access Server.
   b. Modify the Access Server configuration if needed to turn the Access Management Service Off.
   c. Repeat these steps for all Access Servers.
2. From the Access System Configuration tab, click Access Server Clusters and perform the following steps:
   a. Click the name of a cluster.
   b. Confirm that the Policy Manager API Support Mode is off.
   c. Repeat these steps for all Access Server clusters.
3. From the Access System Configuration tab, click AccessGate Configuration and perform the following steps:
   a. Locate and click the name of an AccessGate.
   b. Under the label Access SDK Client, confirm that the Access Management Service is Off.
   c. Repeat these steps for all AccessGates that use the Access Server.
4. From the Access System Configuration tab, click Authentication Management and perform the following steps:
   a. On the Authentication Management: List all Authentication Schemes page, confirm that the Update Cache option is not checked.
   b. Click the name of an authentication scheme.
   c. Click the General tab, click Modify and confirm that the Update Cache option is not checked.

d. Click the Plugins tab, click the name of a plug-in, and confirm that the Update Cache option is not checked.

e. Click the Steps tab and confirm that the Update Cache option is not checked.

f. On the Authentication Flow page, click a step name, and then confirm that the Update Cache option is not checked.

g. Repeat these steps for each authentication scheme you have defined.

5. From the Access System Configuration tab, click Common Information Configuration, and perform the following steps:

   a. Click the Master Audit Rule tab.

   b. Confirm that the Update Cache option is not checked.

6. From the Access System Configuration tab, click Host Identifiers, from the List all host identifiers page, confirm that the Update Cache option is not checked

Recovering from Sync Record Corruption

You perform this task after blocking updates from the Policy Manager. You must be a Master Access Administrator to perform this task.

During recovery, the following processes are performed using the original sync records in the directory server:

- Duplicate obSeqNo values in a single oblixGSN entry are removed from the directory server (oblixGSN objectclass will have a single unique ObSeqNo value)
- Duplicate entries for the oblixGSN objectclass are removed from the directory server
- Sync records for the deleted obSeqNo are updated. For example, deleted sync records might also be in a corrupted state after synchronization (the same as GSN entries), so recovergsncorruption attempts repair on these.
- obSeqNo is updated with the maximum or largest ObSyncRequestNo

If recovery is not successful, you need to perform steps to revert to the earlier status of the directory server (the state before recovery was attempted). For more information, see "Correcting Errors that Occurred During the Recovery Process" on page 8-21.

To recover from sync record corruption

1. Locate the recovergsncorruption command-line tool in the following path:

   PolicyManager_install_dir\access\oblix\tools\recovergsncorruption\

2. Start detection and recovery by providing the following command-line arguments:

   ```
   recovergsncorruption -i PolicyManager_install_dir -isCacheFlushDisabled yes
   -recoverCorruption yes
   ```

3. Proceed as follows:

   - **Error During Recovery Process**: Proceed to "Correcting Errors that Occurred During the Recovery Process" on page 8-21.
   - **Recovery Completed Successfully**: All corruptions have been resolved and cache flush will proceed without problem. In this case, you can restore cache flush operations as described in "Restoring Cache Flush Operations" on page 8-21.

Correcting Errors that Occurred During the Recovery Process

If the sync record recovery process occurred without error, you can skip this topic. You must be a Master Access Administrator to perform this task.

If errors occurred during sync record recovery, you can import the ldif file that was archived during the recovery process to restore sync records in the directory server to the state they were in before recovery began. The ldif contains entries for the oblixgsn objectclass (used in cache flushing) and obsyncrecord (describes what component has been flushed and what policy domain or policy it belongs to) under the PSC node. These were present before recovery was attempted.

The sync records ldif file that was created during the recovery process will be stored as follows:

PolicyManager_install_dir\access\oblix\data\common\

A naming convention of GSNsyncrecordsyear month day.hour minute seconds.ldif is used. For example:

GSNsyncrecords.20080729.040844.ldif

To recover from errors that occurred during the sync record recovery process

1. Confirm that the global sync number update is disabled as described in "Disabling Access Server Cache Flush and GSN Updates" on page 8-18.
2. Confirm that updates from the Policy Manager and applications remain blocked as described in "Blocking Updates from the Policy Manager and Applications using AMAPI" on page 8-18
3. Locate the sync records .ldif file that was created during the recovery process. For example:

   ```
   PolicyManager_install_dir\access\oblix\data\common\
   GSNsyncrecords.20080729.040844.ldif
   ```

4. Run the commands for your directory server to import the ldif and restore the directory server to its state before recovery.
5. Proceed as follows:
   - **Successful**: You are finished.
   - **Not Successful**: You need to manually remove the corruption from the directory server. See your directory server documentation for details.

Restoring Cache Flush Operations

You must be a Master Access Administrator to perform this task.

After a successful recovery, you can enable Access Server cache flush by setting the '`doAccessServerFlush`' parameter to `true` in the basedbparams.xml file of each Identity Server. You must also unblock updates from the Policy Manager.

To enable Access Server cache flush

1. Locate the basedbparams.xml file in the following Identity Server path:

   IdentityServer_install_dir/identity/oblix/data/common/basedbparams.xml

2. In the basedbparams.xml file, locate the `doAccessServerFlush` parameter and set it to `true`.
3. Save the file.

4. Restart the Identity Server.
5. Restart the Access Server.
6. Repeat these steps for all Identity Servers and restart each server one by one.
7. Restore blocked updates from the Policy Manager and applications by reversing the actions you took to block these updates.
   - **Update Cache**: Turn on the Update Cache option on pages in Identity System Console, Policy Manager, and Access System Console.
   - **Policy Manager API**: Ensure that the Policy Manager API is on for the Access Server Configuration, Access Server Clusters, and AccessGate Configuration pages.

   Note: The terms "Policy Manager API" and "Access Management Service" are used interchangeably in the System Console.

   For details, see "Blocking Updates from the Policy Manager and Applications using AMAPI" on page 8-18.

9

Managing Access System Configuration Files

Some Access System administration tasks are performed outside the Access System Console. This chapter contains the following topics:

- Prerequisites
- Automatic Access System Cache Flush
- Synchronization of Access System Components
- Reducing Overhead for Viewing Policy Domains
- Customizing the Policy Manager User Interface

For more information about managing the Access System, see:

- "About Access System Configuration and Management" on page 8-1
- "Configuring Access Administrators" on page 2-1

Prerequisites

Oracle Access Manager should be installed and set up as described in the *Oracle Access Manager Installation Guide*. Read the *Oracle Access Manager Introduction* manual, which provides an overview of Oracle Access Manager not found in other manuals. Also, familiarize yourself with the chapters in this manual that explain Access System configuration and administration. Finally, the *Oracle Access Manager Identity and Common Administration Guide* describes functions that are common to the Identity and Access Systems.

Automatic Access System Cache Flush

The Identity System and the Access System use different user and group caches. You can implement automatic cache flushing for the Access System to ensure that the Access Server's cache is replaced with the latest information. For this, you must set the Access Management Service option to "On" for the Access Server and associated AccessGate configuration profiles. For more information about flushing the Access Server caches, see:

- Table 3–1, " Access Server Configuration Parameters" on page 3-4
- Table 3–3, " AccessGate Configuration Parameters" on page 3-19
- "Flushing User Data from the Cache" on page 8-3
- "Flushing Password Policy Caches" on page 8-6

- The *Oracle Access Manager Deployment Guide* provides more information about Oracle Access Manager caches.

Synchronization of Access System Components

You can synchronize two aspects of the Access System:

- **System Clocks**: This is required.
- **Component Configurations**: You have the option of copying some or all configuration information from one Access System component to another.

For information on synchronizing the configuration of two Access System components, see the *Oracle Access Manager Installation Guide*. See also:

- Synchronizing System Clocks
- Changing Default Configuration Cache Timeout

Synchronizing System Clocks

The clocks of all computers hosting Oracle Access Manager components must be synchronized. Without synchronization, users may not be able to log in to the components or log in to the System Console.

The two possible scenarios are:

- WebPass and Policy Manager are installed on one computer, and Identity Server is installed on another computer.
- WebPass is installed on a computer without Policy Manager, and is configured to route requests to two or more Identity Servers.

To implement synchronization

1. Specify a value for the `loginslack` parameter, located in each of these files:

   PolicyManager_install_dir`/access/oblix/apps/common/bin/oblixbaseparams.lst`

   Identity_install_dir`/identity/oblix/apps/common/bin/oblixbaseparams.xml`

   where *PolicyManager_install_dir* is the directory in which the Policy Manager is installed and *Identity_install_dir* is the directory in which Identity Server is installed.

2. The value that you set specifies the acceptable maximum time difference, in seconds, between the two clocks.

For the first scenario, you must set the value for the loginslack parameter in both files to the same number. For the second scenario, you must set the value for the parameter in each identity server installation directory to the same number.

Changing Default Configuration Cache Timeout

A second way to reduce off-time network traffic between both the WebGate and Access Server and between the Access Server and the LDAP directory server is to change the default configuration cache timeout for WebGate and Access client configurations that are cached in the Access Server.

> **See Also:** "About the Cache Timeout", in the *Oracle Access Manager Deployment Guide*.

To change the default configuration cache timeout

1. Navigate to the globalparams.xml file located in:

   WebGate_install_dir/access/oblix/apps/common/bin/globalparams.xml

   where *WebGate_install_dir* is the directory where WebGate is installed.

2. Add the following parameters and specify their values:

   - clientConfigCacheMaxElems

     The default value is 9999.

   - clientConfigCacheTimeout

     The default value is 59 seconds.

The default values listed should cause no change in the system behavior on non-Apache Access clients. An Apache Web server with WebGate will now avoid excessive hits to the directory server.

Reducing Overhead for Viewing Policy Domains

You can reduce overhead on the My Policy Domains page by turning off the display of the Resource Type and URL Prefix columns on that page. Note that these columns may contain useful information, so the gain in performance is a tradeoff.

To turn off the display of Resource Type and URL Prefix columns

1. Locate the Policy Manager's globalparams.xml file:

   PolicyManager_install_dir/access/oblix/apps/common/bin/globalparams.xml file

   where *PolicyManager_install_dir* is the directory where Policy Manager is installed.

2. Set the value of the parameter limitAMPolicyDomainResourceDisplay to true.

   By default, the value of this parameter is false. The Resource Type and URL Prefix columns are displayed by default. For more information on Policy Domains, see "About Policy Domains and Their Policies" on page 4-6.

Customizing the Policy Manager User Interface

When you invoke the Policy Manager, the My Policy Domains page is displayed. This page lists all of your policy domains. If you are interested in a certain policy domain, you can scroll through the list to find it. If you are responsible for a large number of policy domains, the list will be long. An easier and faster way to find the desired policy domain would be to search for it by name.

Rather than displaying the My Policy Domains page as the first page you see in the Policy Manager, you may set the Search page as the default. In addition, you may customize the Search page. Topics in this section explain:

- Setting the Search page as the Default Page
- Customizing the Policy Manager Search Interface

For additional information on customizing these items, see the *Oracle Access Manager Customization Guide*.

Setting the Search page as the Default Page

With the Access System, you can change the first page displayed by the Policy Manager from the My Policy Domains page to the Search page. The Master Administrator responsible for the Web server can change the default by modifying the configuration base parameter list file, oblixbaseparams.lst. Changes made to this file occur at the Access Server level. If you change the default, it affects all users of the Policy Manager.

To set Search as the default page

1. Open the following file in an editor:

   PolicyManager_install_dir/access/oblix/apps/common/bin oblixbaseparams.lst

   where *PolicyManager_install_dir* is the directory where Policy Manager is installed.

2. Locate the following section in the file:

   ```
   policyservcenter_application_info:
   ```

3. Change the entry as follows:

   From:

   ```
   PROGRAM:../../policyservcenter/bin/policyservcenter.cgi
   ```

   To:

   ```
   PROGRAM:../../policyservcenter/bin/policyservcenter.cgi?progr
   am=navbar&selected_prog= searchframepage
   ```

4. Save the file and close it.

5. Restart the Web server.

Customizing the Policy Manager Search Interface

When you perform a search in the Policy Manager, the default number of results shown is 8. This means that 8 results are displayed just beneath the search bar. You may want to change the default value. You may also want to limit the type of searches by altering what appears in the Policy Manager Search page list, which by default includes the following values:

- That Contains
- Contains in Order
- That Begins with
- That Ends with

For more information, see the following procedures:

- To change the default number of search results
- To change search parameters

To change the default number of search results

1. Locate and open the following file in a text editor:

   PolicyManager_install_dir\access\oblix\apps\common\bin\oblixbaseparams.lst

2. Change the default value of defaultDisplayResultVal to a number other than 8.

3. Save the file, and restart the Web server.

To change search parameters

1. Locate and open in a text editor the policyservcenparams.lst file:

   PolicyManager_install_dir\access\oblix\config\policyservcenparams.lst

2. Locate the following ObEnhanceSearchList parameter and values:

```
\ObEnhanceSearchList:
BEGIN:vNameList
OOS:MOOS
OSM:MOSM
OBW:MOBW
OEW:MOEW
END:vNameList
```

3. Comment out or delete the values from this list of values.
4. Save the file and restart the Web server.

Part IV

Appendices

Information presented in this part explores how to trigger form-based authentication and 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. In addition, you can enable impersonation in the Access System that overrides impersonation enabled with IIS.

Part IV contains the following appendices:

A

Form-Based Authentication

Authentication involves determining what credentials a user must supply when requesting access to a resource, gathering credentials over HTTP, and returning an HTTP response that is based on the results of credential validation.

Form-based authentication enables you to create customized Web forms that process user logins using the Access System's authentication and authorization mechanisms. These forms are HTML pages that allow you to present login information in different languages, to display user interface elements that comply with your company's presentation standards, and to add functions to the login page: for example, for lost password management.

This chapter covers the following topics:

About Form-Based Authentication

As described in "Configuring User Authentication" on page 5-1, the Access System challenges the user with a form that was configured in an authentication scheme under the following conditions:

- If a Web resource is protected using a policy with an authentication scheme that requires a form and there is no valid session cookie (ObSSOCookie).
- If the valid session cookie exists, but it is from a lower authentication level (regardless of the challenge method).

The authentication challenge is an HTML form with one or more text input fields for user credentials. In a typical form-based authentication, users enter a user name and password in two text boxes on the form. The most common credential choices are user name and password, but any user attributes can be used, for example, user name, password, and domain. A Submit button posts the content of the form. When the user clicks the Submit button, the form data is posted to the Web server. WebGate intercepts and processes the form data. Upon validation of the user credentials collected in the form, the user is authenticated.

You may want to use form-based authentication for reasons such as the following:

- Using form-based login and a standardized logout means that the user experience for login and logout features will be consistent across browsers.

  See "Configuring Logout" on page B-1 for details.

- You can apply your organization's look and feel in the authentication process.

  For example, a custom form can include a company logo and a welcome message instead of the standard user name and password window used in Basic authentication.

- You can gather additional information at the time of login.

- You can provide additional functionality with the login procedure, such as a link to a page for lost password management.

  See the information on lost password management in *Oracle Access Manager Identity and Common Administration Guide* for details.

The forms that you create for form-based authentication only collect user credentials. The processing of authentication and authorization are handled by other functions. See Chapter 4, "Protecting Resources with Policy Domains" on page 4-1 for an overview.

The following is a summary of configuring form-based authentication. See "Configuring Form-Based Authentication" on page A-8 for details.

Task overview: Configuring form-based authentication

1. Create an HTML form where the user's credentials, such as user name and password, can be submitted.

   See "Considerations when Creating a Form" on page A-7 for details.

2. Place the form in an unprotected directory, or in a directory protected by an Anonymous authentication scheme, on your Web server with a WebGate.

   The same login form and its associated authentication scheme can be used by multiple policy domains.

3. Set up an authentication scheme to use form-based authentication and define the path to the login form.

   See "Defining and Managing Authentication Schemes" on page 5-7 for details.

4. Call the form action using HTTP GET or POST.

   See "About the Form Action" on page A-10 for details.

5. Protect the target URL in the action of the login form with a policy.

   See Chapter 5, "Configuring User Authentication" on page 5-1 for details.

6. Configure the challenge parameters and passthrough mode in the authentication scheme.

   See Challenge Parameters on page A-3 for details.

7. Specify the plug-ins.

   See "Plug-Ins Used with Form-Based Authentication" on page A-5 for details.

The rest of this section discusses the following topics:

- Challenge Parameters
- Redirection
- Plug-Ins Used with Form-Based Authentication

- Session Cookie and Authentication Actions
- Header Variables
- Using an External Call for Data in an Authentication Request

Note: On IIS, you must install the Access Manager and WebGate under the same folder or form-based authentication will fail.

Challenge Parameters

As discussed in "Defining a New Authentication Scheme" on page 5-9, When you select the Form challenge method, you are required to provide the following three parameters in the Challenge Parameter fields.

See Also: "About the sensitivecreds Challenge Parameter" on page 5-6

Challenge Parameter	Description
`form:`	Indicates where the HTML form is located relative to the host's document directory. For example: `form:/login.html`
`creds:`	Lists all fields used for login in the HTML form. Creds: is a space-separated list. For example: `creds:login password` When the user submits a login form that is protected by an authentication scheme with a Form-based challenge method, WebGate processes the credentials that were specified on this parameter. For a from that uses METHOD=POST processing, the browser sends a POST request to the Web server with the credential data from the form in the body of the request. If the form uses METHOD=GET, the browser sends a GET request with query string parameters with the same names as those specified on the `creds` parameter. Oracle recommends that you use POST processing, if possible. Note: You can specify the `creds` parameter for the other types of challenge methods.
`action`	The URL that the HTML form is posting to.
`ssoCookie`	Used to secure the single sign-on cookie and to set a cookie that can persist over multiple sessions. See "Securing the ObSSOCookie in an Authentication Scheme" on page 5-16 and "Retaining the ObSSOCookie Over Multiple Sessions" on page 5-17 for details.

Note: During form-based authentication with a custom plug-in, the original resource name is not available to the plug-in within the pre-defined names in the Challenge Parameter `creds` list. For example, in the Authentication Plug-in API the `ObAnPluginInfo` construct contains the `Creds` data type where the Access Server provides four pre-defined names within this list: Resource, Operation, RequesterDN, and RequesterIP.

When using form-based authentication, the Resource returned by the API is the resource that the login form POSTs to (not the actual resource of the original URL).

Additional parameters, `passthrough` and `maxpostdatabytes`, are optional.

Challenge Parameter	Description
`passthrough:`	This parameter value determines whether the WebGate redirects the browser back to the original requested resource or passes the login credentials on to another program. The Access System assumes that the URL given for the form in the authentication scheme is on the same computer as WebGate. Possible values are yes or no. Accept the default value of no if you want WebGate to redirect the browser back to the originally requested resource. This omits a form challenge parameter. Specify yes if you want to pass the login credentials through to a post-processing program.
`maxpostdata bytes:`	The value of this optional parameter specifies the maximum number of data bytes that an end user can post for authentication to a Web server that uses a WebGate. If the POST data exceeds the threshold set by maxpostdatabytes in the form-based authentication scheme, user receives an error and a log entry is added at the DEBUG3 log level in the oblog.log file. Example: `maxpostdatabytes:100` If you omit this parameter, the end user can post unlimited length strings for authentication to a Web server that is protected by a WebGate. Very long strings can cause the WebGate or Web server to crash, denial of service, or another fatal error.

Enter `passthrough:yes` if you want to pass the login credentials to a post-processing system, for example, if you want to pass the login credentials to a post-processing program for single sign-on to an application that does not accept header variables.

If you set `passthrough:yes`, note that this settings causes Oracle Access Manager to perform post-processing of the form. The post-processing causes Oracle Access Manager to forget the redirect URL (the original URL of the user's request). Therefore, when you set `passthrough:yes`, you must also enter a landing page in the action of the form and protect the landing page. Alternatively, you can create an authentication action that redirects the user to the landing page.

The default is `passthrough:no`, whether it is explicitly set or if the field is left blank. If you accept the default passthrough mode but want to redirect users to a page other than the originally requested resource, in the policy domain rule specify a redirection to another page upon authentication success. If redirection to the login form occurs as

described in "Redirection" on page A-5, and passthrough mode is not set for the form authentication scheme, WebGate redirects the browser back to the originally requested resource. You can use the ObRequestedUrl header variable to redirect.

> **Note:** Setting `passthrough:yes` sends the password in a URL string. This string can be obscured, however, you should check your organization's security policies before configuring passwords to be sent in the URL.

Redirection

If the login form is the page that user requests, redirection is not needed. However, users can attempt to go around a login form, for example, by bookmarking pages. In these cases, WebGate redirects the request to the login form. After authentication success, WebGate redirects the user back to the requested resource.

A cookie named obFormLoginCookie maintains the original request information. By default, this cookie is set when the browser is first redirected to the form. Information in this cookie includes:

- The requested URL
- The requested operation
- An authentication scheme
- The URL of the host to return to

Without this cookie, WebGate would be unable to send the originally requested resource upon authentication.

When the user authenticates, the ObSSOCookie is also set. For more information on the ObSSOCookie, see "Single Sign-On Cookies" on page 7-2.

Plug-Ins Used with Form-Based Authentication

You need several plug-ins to work with your form authentication scheme. The order of the plug-ins is also important.

Credential Mapping Authentication Plug-In

Credential mapping is defined for each login form. The credential_mapping plug-in performs the task of mapping the user-supplied credentials to a unique DN in the directory server. WebGate searches the directory for profiles with attributes matching the form credentials. It handles the password credential consistently with basic authentication.

Logically, password validation can only happen after the user is identified. Therefore, the credential_mapping plug-in must be used before validate_password and must be the first plug-in specified in your form-based authentication scheme.

Validate Password Authentication Plug-Ins

Form authentication uses the same validate_password plug-in that is used in basic authentication. You can configure the name of the password field.

More Possible Custom Authentication Plug-Ins

As with basic authentication, custom authentication plug-ins can be used to check the user name and password using other login services and user repositories. In fact, the same processing functions could be used for both basic and form user name/password

authentication. Custom authentication plug-ins can also process other user credential data.

> **Note:** During form based authentication with a custom plug-in, the original resource name is not available to the plug-in within the pre-defined names within the Challenge Parameter creds list. For example, in the Authentication Plug-in API the ObAnPluginInfo construct contains the Creds data type where the Access Server provides four pre-defined names within this list: Resource, Operation, RequesterDN, and RequesterIP.
>
> When using form-based authentication, the Resource returned by the API is the resource that the login form POSTs to (not the actual resource of the original URL).

For more information about plug-ins, see "Configuring a Form-Based Authentication Scheme" on page A-8.

Session Cookie and Authentication Actions

If WebGate intercepts the form login, it can build the session cookie and carry out the authentication actions.

> **Note:** If a form authentication scheme on IIS is configured with the passthrough option, and the target of the login form requires the data posted by the form, the WebGate extension method (where the WebGate DLL is the action of the form) cannot be used. The WebGate filter method (where the action of the form is a protected URL that is not the WebGate DLL) must be used instead, and the postgate DLL must be installed and enabled. See the *Oracle Access Manager Installation Guide* for details.

Header Variables

Form-based authentication schemes can pass authorization actions in header variables. However, they cannot pass *authentication* actions in header variables.

Using an External Call for Data in an Authentication Request

An authentication scheme can collect context-specific information before submitting the request to the Access Server. Context-specific information can be in the form of an external call for information. This information can be of the following types:

- server: variables set by other Web server plug-ins
- header: HTTP header variables
- post: posted data
- query: query string data
- cookie: HTTP cookie

To retrieve external data for an authentication request

1. Create an authentication scheme as described in "Defining a New Authentication Scheme" on page 5-9.

2. In the Challenge Parameter field, specify the following:

   creds:*source*$name

   or

   creds:name

   where *source* is one of the following:

   - server
   - header
   - post
   - query
   - cookie

If you omit the source, sources are searched in the order shown in the list.

> **Note:** The Web server source (the server parameter) takes precedence over other sources. This prevents the request data, which is under control of the user, from overriding Web server data. For example, a remote_user cookie sent from a user will not override a remote_user variable set by the Web server.

If the client is a WebGate, as opposed to the Access Manager SDK, the WebGate will extract the requested data. If the client is the Access Manager SDK, it is up to the calling program to collect this data.

For a plug-in to make use of the creds parameter, you specify what is passed in the obMap credentials parameter of the ObUserSession object. See the *Oracle Access Manager Developer Guide* for details.

Considerations when Creating a Form

You need to create a custom form that you want users to see when they access a protected resource. The form can be as complex as you want it to be. Within the form, you must at least provide fields for a user to submit a login and password.

> **Note:** Do *not* protect the form or any of its components (such as GIFs and links) with an authentication method, or use an Anonymous authentication scheme.

Key areas to consider when you are designing a form are:

- The ObFormLoginCookie as described in "ObFormLoginCookie" on page A-8.
- Form actions, as described in "About the Form Action" on page A-10.
- Form actions and webgate.dll, as described in "Notes for Microsoft IIS" on page A-11.
- Character set encoding for the login form must be set to UTF-8, as shown and described in "Sample Login Form" on page A-15.

ObFormLoginCookie

As previously mentioned, WebGate sets the ObFormLoginCookie when the browser is first redirected to the form. This can become a problem in the following situations:

- If your login form has a link for Password Management that is protected by an Anonymous authentication scheme, the user is redirected back to the login form instead of going to the lost password link.
- After the login has been completed, WebGate marks the ObFormLogin Cookie "done" and will not allow the user to use the form login again within the same browser instance. This causes a problem for the oblogout functionality. When a user tries to log out, and then log back in, WebGate bypasses the form login processing.

You can avoid these situations by entering an action challenge parameter when you configure your form authentication scheme. See "Protecting Resources with Policy Domains" on page 4-1 for details.

Configuring Form-Based Authentication

The following procedures describe how to configure a form and an authentication scheme for the form.

Task overview: Creating a form for authentication

1. Create a custom form that you want users to see when accessing a protected resource, using considerations described in "Considerations when Creating a Form" on page A-7.

   Note: Do not protect the login form or any of its components (such as GIFs and links) with an authentication method. Alternatively, you can protect these items with an Anonymous authentication scheme.

2. Place the form in an unprotected directory, or in a directory protected by an Anonymous authentication scheme, on your Web server with WebGate.

   The same login form and its associated authentication scheme can be used by multiple policy domains.

3. Configure a form-based authentication scheme, as described in "Configuring Form-Based Authentication" on page A-8.

The rest of this section discusses the following topics:

- Configuring a Form-Based Authentication Scheme
- Notes for Microsoft IIS
- Including Users in the obMappingFilter

Configuring a Form-Based Authentication Scheme

When you create an authentication scheme you include the name, an optional description, and the level of the authentication scheme. Parameters and options are described within the following procedure. For more information about authentication schemes, see Chapter 5, "Configuring User Authentication" on page 5-1.

Note: When a form resides on the same server as a WebGate, the relative form URL given for the form in the authentication scheme is on the same computer as WebGate. In this case, you do not include the https:// (or http://) *host*:*port* portion of the URL in the authentication scheme. However, when the form resides on a remote server, the host and port are required in the authentication scheme.

To configure a form-based authentication scheme

1. In the Access System Console, click Access System Configuration, select Authentication Management, then click Add.

   The Define a New Authentication scheme screen appears.

2. Enter the following for the authentication scheme:
   - A name.
   - A description.
   - The level of the authentication scheme: The level of the scheme is a number that corresponds to the relative security level for this scheme. Higher levels are considered more secure.

3. Select Form as the Challenge Method, as described in "About Challenge Methods" on page 5-5.

4. In the Challenge Parameter field, enter the following:

   ```
   form:relative_form_URL
   creds:credential_names
   action:Action_URL
   passthrough:[yes] (Optional)
   ```

   - The Access System assumes the relative form URL given for the form in the authentication scheme is on the same computer as WebGate.

     Do not include the `http://host:port` portion of the URL if the authentication scheme is on the same computer as the WebGate.

     For example:

     ```
     form:/login.html
     ```

   - Credential names are a space-separated list of expected credential names from the form.

     For example:

     ```
     creds:login password
     ```

   - The Action URL sets the ObFormLoginCookie to be returned only when the form posts the login credentials.

     For example:

     ```
     action:/access/dummy.cgi
     ```

     For more information, see "About the Form Action" on page A-10.

   - The default passthrough mode is no. Accept the default if you want the Access System to automatically redirect users to their original requested resource.

5. Specify whether or not you want the user to authenticate using SSL.

You can also use Challenge Redirect to redirect the users to a central location storing all forms.

6. If you answered yes to SSL, specify the Challenge Redirect URL for your secure server.
7. Enter the following two required plug-ins:

Order	Plug-in Name	Plug-in Parameters
1	credential_mapping	*obMappingBase="o=company,c=us"* (the base DN in the LDAP search). *obMappingFilter="[(Identity Login Attribute=%form input field for login%)]"*
2	validate_password	*ObCredentialPassword="[form input field for password]"*

WARNING: The directory login attribute is an attribute defined in the Identity System using a semantic login type, as discussed in the Oracle Access Manager Identity and Common Administration Guide. Also, you cannot have spaces in the filter. The Policy Manager does not validate the string that you provide as the credential_mapping filter, so it is possible to enter an erroneous filter. No error occurs while saving; however, the filter will fail and the plug-in will return "Authentication Failed" each time it is run.

For information about users and the obMappingFilter, see "Including Users in the obMappingFilter" on page A-12.

8. Click Save.

About the Form Action

The form action does not process the credentials for authentication. This is the job of the Access System plug-ins that you configure for the form-based authentication scheme.

In the `input` elements of the login form, the `name` attribute values must match the values that you specified in the `creds` parameter for the challenge method. When a user logs in using this form, he or she must supply all of the credentials on the form correctly, or the WebGate redirects the user back to the login form.

In the `form` element of the login form, the `action` attribute is a URL to which form data is posted when the user submits the form.

For example, in the following form the action URL is /access/dummy and the method is POST:

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html;charset=utf-8" >
</head>
<form name="myloginform" action="/access/dummy" method="post">
UserID <input type="text" name="userid" size="20" value="user1k1">
Password <input type="password" name="password" size="20" value="oblix">
<input type="submit" name="submit" value="Login">
</form>
</html>
```

The action URL is configured so WebGate sets the ObFormLoginCookie for the action URL path, and this cookie is only returned on the form post. When a user submits credentials, the form action is called using the HTTP GET or POST method. The form action does not process the user's credentials for authentication. That is the job of the plug-ins configured for the form-based authentication scheme.

The form action can be a call to a URL that does not do anything. When the form posts to an action URL, WebGate intercepts the post because of the ObFormLoginCookie. WebGate processes the credentials in the post data, authenticates the user, and redirects the user to the originally requested URL as indicated by the ObFormLoginCookie. Since the action URL is never reached, it does not actually have to exist. All that is required is that a policy protect the action URL. In the previous example of a form, the action URL /access/dummy is protected by a policy domain that protects all URLs subordinate to /access. However, /access/dummy, as the name implies, does not exist.

The form action can also be a call to a script that does post-authentication processing. For example, you may have a script that does post-processing on credentials to achieve single sign-on for an application that does not accept header variables. When the form action is a script, the authentication scheme must be configured with the passthrough:yes challenge parameter. This tells WebGate that the action URL is a script that must be executed after the form login. In this case, WebGate does not redirect the user to the originally requested URL. WebGate allows the Web server to continue processing the action URL. WebGate passes the originally requested URL in the ObRequestedURL header variable to the action URL script, and the script can redirect to the original URL if desired.

Note: The form action URL must reside in a policy domain protected by the Access System.

Forms that Reside on Servers Other Than a WebGate

When the form resides on the same server as a WebGate, the submit action assumes that the local host is being used. However, if the form is on a different server from the WebGate, the submit action in the form must return the data to the Web server where the WebGate resides.

Notes for Microsoft IIS

Because of the IIS architecture, the WebGate ISAPI plug-in checks all incoming requests for post-processing data. You must do one of the following:

- Either set your form action to call the webgate.dll, for instance:

  ```
  action="/access/oblix/apps/webgate/bin/webgate.dll"
  ```

  Note: With version 6.5, a new directory structure was instituted to accommodate localization. Before version 6.5, the form action contained a different path to webgate.dll.

- Or ensure the WebGate filter post-processing is turned on by setting the following Registry entry:

  ```
  HKEY_LOCAL_MACHINE\SOFTWARE\Oblix\Oblix
  COREid\version\WebGate\postdata="yes"
  ```

where version is the version number of the installed product.

Including Users in the obMappingFilter

This topic describes:

- Including Only Active Users
- Including Non-Active Users

Including Only Active Users

You may want to include only activated users in your obMappingFilter so that only activated users can login. For this task, you must filter users whose obuseraccountcontrol=ACTIVATED.

To include only active users in the obMappingFilter

1. Follow the procedure "To configure a form-based authentication scheme" on page A-9.
2. In the mapping filter, specify only active users. An example:

```
obMappingFilter="(&(objectclass=wwmOrgPerson) (uid=%loginid%) ( | ( !
(obuseraccountcontrol=*)) (obuseraccountcontrol=ACTIVATED)))"
```

> **Note:** This example uses the Oracle sample data (wwmOrgPerson). Change this object class to your site-specific object class. The uid=%loginid% assumes the form has a field called loginid and that this value is also included in the creds field.

Including Non-Active Users

You may want to include non-active users in your obMappingFilter so that deactivated users cannot login. For this task, you filter users with a status of obuseraccountcontrol=PENDING-ACTIVATION or PENDING DEACTIVATED.

To include only non-active users in the obMappingFilter

1. Follow the procedure "To configure a form-based authentication scheme" on page A-9.
2. In the mapping filter, specify the inactive users. For example:

```
obMappingFilter="(&(objectclass=wwmOrgPerson)
(uid=%userid%)(!(|(obuseraccountcontrol=
PENDING-ACTIVATION)(obuseraccountcontrol=DEACTIVATED)
(obuseraccountcontrol=PENDING-DEACTIVATION))))"
```

> **Note:** This example uses the Oracle sample object class wwmOrgPerson. You must change this object class to your site-specific object class. The uid=%loginid% assumes the form has a field called loginid and that this value is also included in the creds field.

Form Examples

The following sections contain examples of forms that can be used for form-based authentication:

- Form Scheme Examples
- Sample Pop-Up Forms
- Sample Multi-Language Form

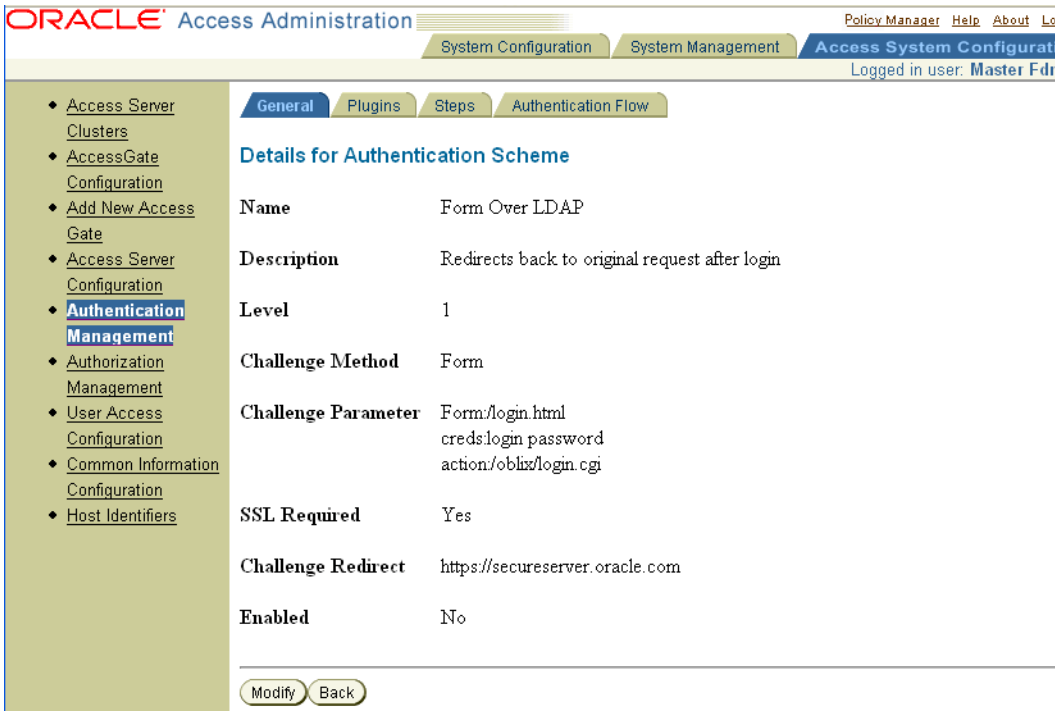
Form Scheme Examples

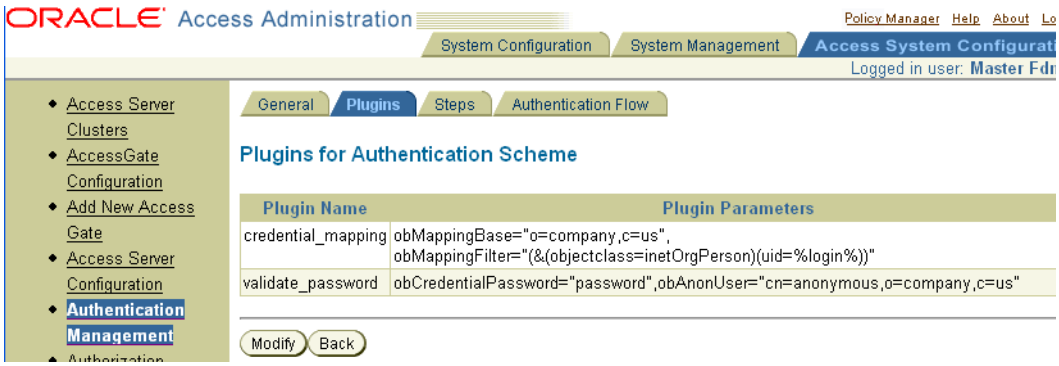
The following are examples of HTML forms and corresponding authentication schemes.

Basic Example

The following is a very simple login form:

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html;charset=utf-8" >
</head>
<h1>My Login Form</h1>
   <form name="loginform" action="/oblix/login.cgi" method="post">
   Login ID: <input type="text" name="login" size="20" value="">
   <p>
   Submit:<input type="submit" name="submit" value="OK">
   <p>
   Password:<input type="password" name="password" value="">
   </form>
   </html>
```





Annotated Example

The following is another sample login form. It shows the minimum requirements for a form login authentication scheme. A production login form can be enhanced for aesthetics and branding. An example of an authentication scheme using this form is as follows:

Name: Sample Form Login

Description: Uses SampleLoginForm.html

Level: 1

Challenge Method: Form

Challenge Parameters:

form: /loginforms/SampleLoginForm.html

creds: -userid password

action: /access/oblix/apps/webgate/bin/webgate.dll

SSL Required: no

Challenge Redirect: (none)

Enabled: yes

Plug-ins:

```
credential_mapping
obMappingBase="o=Company,c=US",
obMappingFilter="(&(&(objectclass=gensiteorgperson)
(uid=%userid%))(|(!(obuseraccountcontrol=*)) (obuseraccountcontrol=ACTIVATED)))"
validate_password obCredentialPassword="password
```

For Active Directory, use "user" for the object class and "samaccountname" for the login. For example:

```
credential_mapping for Active Directory
obMappingBase="ou=Hokaido,dc=perry,dc=oblix,dc=com",
obMappingFilter="(&(&(objectclass=user)(samaccountname=%login%)
)(|(!(obuseraccountcontrol=*))(obuseraccountcontrol=ACTIVATED)) )"
```

The login form must be either unprotected or protected by an authentication scheme with a challenge method of None. This ensures that the user is not re-challenged when redirected to the login form. For the sample scheme, you can configure a policy domain that protects the form using the Anonymous authentication scheme. This sets

a temporary ObSSOCookie when the login form is displayed. The ObSSOCookie is rewritten after a successful login.

In the sample scheme, the userID is the uid attribute from the user's directory profile. The credential_mapping plug-in searches the user directory from the base o=Company,c=US. The credential_mapping plug-in searches for the gensiteorgperson object that contains a uid matching the submitted userID. The additional information in the ObMappingFilter determines whether the user is activated. The validate_password plug-in performs a BIND to the directory, using the submitted password and the user profile DN retrieved when the credential_mapping plug-in searches the directory.

The action is the WebGate local URL. This URL must be protected using any authentication scheme. For example, you might use the Policy Manager policy domain that was optionally created during setup of the Policy Manager.

In the case of IIS, the WebGate action is executed as an ISAPI extension, which allows it to safely obtain the post data containing the credentials. In the case of other Web servers, WebGate intercepts the post request (because the action URL is protected) and extracts the post data for authentication. WebGate sets the ObFormLoginCookie using the action challenge parameter as its path. This ensures that the ObFormLoginCookie is returned only on the post request from the form submission. The ObFormLoginCookie contains information about the originally requested resource. After a successful authentication, WebGate uses this information to redirect the user's browser to the originally requested resource. In the redirection, WebGate sets the ObSSOCookie with the user identity, authentication scheme level, session start and refresh time, and browser IP address.

Sample Login Form

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<HEAD>
<TITLE>Oracle Access Manager Sample Login Form</TITLE>
<META http-equiv=Content-Type content="text/html; charset=utf-8">
<META content="MSHTML 6.00.2800.1226" name=GENERATOR>
</HEAD>
<BODY>
<H2>Oracle Access Manager Sample Login Form</H2>
<FORM name=SampleLoginform action=/access/oblix/apps/webgate/bin/webgate.dll
method=post>
UserID
<INPUT name=userid>
Password
<INPUT type=password name=password>
<INPUT type=submit value=Login name=submit>
</FORM>
</BODY>
</HTML>
```

Starting with release 10.1.4, WebGates receive only UTF-8 encoded input data. For this reason, form-based authentication supports non-ASCII login credentials (username/password). When you use form-based authentication with 10g Release 3 (10.1.4) WebGates, you must ensure that character set encoding for the login form is set to UTF-8.

To set the login form encoding to UTF-8

1. Add the following META tag to the HEAD tag of the login form HTML page.

```
<META http-equiv="Content-Type" content="text/html;charset=utf-8">
```

2. If you upgrade an earlier WebGate to 10.1.4, you must also update the login form HTML page after upgrading.

Sample Pop-Up Forms

The following JSP and ASP code samples create a pop-up login form. This prevents any issues that can arise when a login form is included as a frame within a frameset. The JSP code must be used with a Web server that has a JSP servlet engine. The ASP code must be used with IIS or another Web server with an ASP engine.

When you use one of these login pop-up examples, you need to configure an authentication scheme using one of the following challenge parameters:

`form:login/login.asp` (assuming the ASP form is stored under the /login folder)

or

`form:login/login.jsp` (for the JSP form)

JSP Code Sample

```
<%@ page import="java.util.*" %>
<%
int launchStatus = -1;
String URLVal = "";
String HTTPStart = "http://";
String QueryStr = request.getQueryString();
String ServerName = request.getServerName();
String PathName = request.getServletPath();
if (QueryStr != null)
{
   if (QueryStr.indexOf("launchForm") == -1)
   {
      launchStatus = -1;
   }
   else
   {
      launchStatus = 0;
   }
   URLVal = HTTPStart.concat(ServerName);
   URLVal = URLVal.concat(PathName);
   URLVal = URLVal.concat("?");
   URLVal = URLVal.concat(QueryStr);
   URLVal = URLVal.concat("&launchForm=TRUE");
}
else
{
   URLVal = HTTPStart.concat(ServerName);
   URLVal = URLVal.concat(PathName);
   URLVal = URLVal.concat("?launchForm=TRUE");
}

if ((launchStatus != 0))
{
%>
   <HTML>
   <HEAD>
   <SCRIPT Language="Javascript">
   function openLoginForm()
```

```
   {
      newUrl = "<%= URLVal %>";
      child = window.open(newUrl, "loginFormWindow",

"toolbar=no,directories=no,menubar=no,status=no,scrollbar=no,resizable=yes,widt
h=670,height=400");
      if (child.opener == null)
      {
         child.opener = window;
      }

      window.name = "loginOpener";

      if (navigator.appName == "NetScape") {
         child.focus();
      }
   }
   </SCRIPT>
   </HEAD>
   <BODY bgcolor="#ffffff" onload="openLoginForm(); return true;">
   <center>
   <p>
   <hr>
   <p>
   <Font face="verdana" size="2">
   Please enter your login credentials
   </Font>
   <p>
   <hr>
   <p>
   </center>
   </BODY>
   </HTML>

<%} else %>

   <html>
   <script language="JavaScript">
   function setAction()
   {
      document.forms[0].target=self.opener.name;
      document.forms[0].submit();
      window.close();
   }
   </script>
   <body>
   <center>
   <h1>User Login</h1>
   <br>
   <br>
   <form name="frmlogin" action="/FormProtect/login.cgi" method="post"
      target="loginOpener">
   <hr>
   <b>User ID  </b><input type="text" name="txtUserID">
   <br>
   <b>Password  </b><input type="password" name="pwdPassword">
   <br>
   <input type="button" title="Login" onclick="javascript:setAction();
      " value=Submit>
   <hr>
```

```
   </center>
   </form>
   </body>
   <html>
```

ASP Code Sample

```
<%
dim launchForm
launchForm = Request("launchForm")
if launchForm <> "True" then
   'This is the plain/blank HTML page
%>
   <HTML>
      <HEAD>
      <SCRIPT Language="Javascript">
         function openLoginForm()
         {
            // now open the new window with newUrl
            <% if len(request.servervariables("QUERY_STRING")) > 0 then %>
               newUrl= "<%=request.servervariables("URL") & "?" &
                  request.servervariables("QUERY_STRING") & "&launchForm=True"%>";
            <% else %>
               newUrl= "<%=request.servervariables("URL") & "?launchForm=True"%>";
            <% end if %>
            child = window.open(newUrl, "loginFormWindow",

"toolbar=no,directories=no,menubar=no,status=yes,scrollbar=yes,resizable=yes,wi
dth=670,height=400");
            if (child.opener == null)
            {
               child.opener = window;
            }

            window.name = "loginOpener";

            if (navigator.appName == "NetScape")
            {
               child.focus();
            }
         }
      </SCRIPT>
      </HEAD>
      <BODY bgcolor="#ffffff" onload="openLoginForm(); return true;">
      <center>
      <p>
      <hr>
      <p>
      <Font face="verdana" size="2">
      Please enter your login credentials
      </Font>
      <p>
      <hr>
      <p>
      </center>
      </BODY>
   </HTML>

<% else %>
```

```
   <HTML>
   <SCRIPT language="JavaScript">
   function setAction()
   {
      document.forms[0].target=self.opener.name;
      document.forms[0].submit();
      window.close();
   }
   </SCRIPT>
   <BODY>
   <CENTER>
   <H1>User Login</H1>
   <BR>
   <BR>
   <form name="frmlogin" action="/FormProtect/login.cgi"
         method="post" target="loginOpener">
   <HR>
   <B>User ID  </B><input type="text" name="txtUserID">
   <BR>
   <B>Password  </B><input type="password" name="pwdPassword">
   <BR>
   <input type="button" title="Login" onclick="javascript:setAction();
         " value=Submit>
   <HR>
   </CENTER>
   </FORM>
   </BODY>
   <HTML>
<% end if %>
```

Sample Multi-Language Form

Non-ASCII user credentials are supported in only form-based authentication.

The following ASP code sample is a multi-language form that supports both Spanish and English.

Multi-Language Form

```
<%
Option explicit
dim strLanguage, strNewLanguage, intPointer
dim bolLoginToCOREid
bolLoginToCOREid = Request("LoginToCOREid")
if bolLoginToCOREid = true or bolLoginToCOREid = "true" then
   bolLoginToCOREid = true
else
   bolLoginToCOREid = false
end if

strLanguage = Request.Cookies("PrefLang")
'Response.Write "lenguaje:" & strLanguage
if strLanguage = "" or strLanguage = "EN" then
   strLanguage = "EN"
   strNewLanguage = "SP"
   intPointer = 0
else
   strLanguage = "SP"
   strNewLanguage = "EN"
   intPointer = 1
```

```
end if

dim strUser(1),strPassword(1),strEnter(1),strPreferences(1),strCancel(1)
dim
strLanguageDescription(1),strForgot(1),strDescription(1),strMsgUandP(1),strMsgU
(1)
dim strUserType(1),strCOREidUser(1),strCOREidAdmin(1)

strUser(0) = "User:"
strUser(1) = "Usuario:"
strPassword(0)="Password:"
strPassword(1)="Contraseña:"
strEnter(0) = "Enter"
strEnter(1) = "Proceder"
strPreferences(0)="Preferences"
strPreferences(1)="Preferencias"
strCancel(0)="Cancel-Portada"
strCancel(1)="Cancelar-Portada"
strLanguageDescription(0)="Espanol"
strLanguageDescription(1)="English"
strForgot(0)="Forgot your password?"
strForgot(1)="¿Olvidó su contraseña?"
strMsgUandP(0)= "Please enter your user name and password."
strMsgUandP(1)= "Por favor teclee su usuario y contraseña."
strMsgU(0)= "Please enter your user name."
strMsgU(1)= "Por favor teclee su usuario."
strUserType(0) = "User Type:"
strUserType(1) = "Tipo de Usuario:"
strCOREidUser(0) = "Oracle Access Manager User"
strCOREidUser(1) = "Usuario Oracle Access Manager"
strCOREidAdmin(0)= "Oracle Access Manager Admin"
strCOREidAdmin(1) = "Administrador Oracle Access Manager"

strDescription(0)="Click ""Preferences"" to see and modify some of your
attributes." & _
            "<p>Da un clic en ""Español"" para cambiar esta pagina de idioma." & _
            "<p>Click ""Forgot your password?"" if you don't remember your
                        password and you need to change it, " & _
            "you will be prompt to answer your challenge phrase."

strDescription(1)="Da un clic en ""Preferencias"
                    " para ver y modificar algunos de tus atributos." & _
            "<p>Click on ""English"" to change the language of this page." & _
            "<p>Da un clic en ""¿Olvidó su contraseña?"
                    " si no recuerdas tu clave y deseas cambiarla, " & _
            "será necesario contestar tu frase personal."

dim identityProgram
dim userDN
dim finalURL

identityProgram="/identity/oblix/apps/userservcenter/bin/
userservcenter.cgi?program=modify&tab_id=Employees"
userDN = Request.ServerVariables("HTTP_USERDN")
finalURL = identityProgram & "&uid=" & userDN

dim obTemp
dim ObSSO
dim ObLogin
ObSSO = "ObSSOCookie=loggedout; path=/; domain=.oblix.com"
```

```
Response.Cookies("ObFormLoginCookie") = "done 1"
Response.Cookies("ObFormLoginCookie").Expires = Date() - 1

obtemp = "ObTEMP=%23comp_cookie=false%23; path=/"

%>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<!-- saved from url=(0018)http://10.26.3.90/ -->
<HTML><HEAD>
<TITLE>Login</TITLE>
<meta http-equiv="pragma" content="no-cache">
<%if bolLoginToCOREid then%>
<meta http-equiv="Set-Cookie" content="<%=ObLogin%>"
<%end if%>

<META http-equiv=Content-Type content="text/html; charset=utf-8">

<SCRIPT LANGUAGE=javascript>
<!--

   //Functions for keydown

   nextfield = "login";
   netscape = "";
   ver = navigator.appVersion; len = ver.length;
   for(iln = 0; iln < len; iln++) if (ver.charAt(iln) == "(") break;
   netscape = (ver.charAt(iln+1).toUpperCase() != "C");
   function keyDown(DnEvents) {
     k = (netscape) ? DnEvents.which : window.event.keyCode;
     if (k == 13) {
       if (nextfield == 'done')
       {
          setAction();
      }else{
          eval('document.loginform.' + nextfield + '.focus()');
         return;
      }
     }
    }
   document.onkeydown = keyDown;
   if (netscape) document.captureEvents(Event.KEYDOWN|Event.KEYUP);
   //\Functions for keydown

   expireDate = new Date
   expireDate.setFullYear(expireDate.getFullYear()+7)
   var URLs = new Array(2);
   URLs[0] = "/identity/oblix/apps/userservcenter/bin/
         userservcenter.cgi?usertype=delegatedIdentityAdminBIZ";
   URLs[1] = "/identity/oblix/apps/admin/bin/
         front_page_admin.cgi?usertype=systemAdmin";

   function setCookie (name, value, expires) {
       document.cookie = name + "=" + escape (value)
            + "; expires=" + expireDate.toGMTString() +  "; path=/";
   }
   function delCookie (name) {
       var expireNow = new Date();
       document.cookie = name + "=" + "; expires=Thu,
            01-Jan-70 00:00:01 GMT" +  "; path=/";
```

```
}

function changeLanguage(){
   setCookie("cemexPrefLang","<%=strNewLanguage%>");
   document.location.reload(true);
}
// Delete the cookie
function deletecookie(){
   if (document.cookie != "") {
       thisCookie = document.cookie.split("; ")
           expireDate = new Date
            expireDate.setDate(expireDate.getDate()-1)
          for (i=0; i<thisCookie.length; i++) {
            cookieName = thisCookie[i].split("=")[1]
                document.cookie = "cookieName="+cookieName + ";expires=" +
                        expireDate.toGMTString();
          }
   }
}
function killObCokies(){
   // Kill Any Cookies...

   document.cookie = "<%=obSSO%>"
        document.cookie = "<%=obLogin%>"
   //document.cookie = "ObTEMP=; path=/";
   //delCookie("ObSSOCookie");
   //delCookie("ObFormLoginCookie");
   //delCookie("ObTEMP");
}
function mySubmit() {
   if (!((loginform.login.value.length > 0) &&
      (loginform.password.value.length > 0))) {
      alert("<%=strMsgUandP(intPointer)%>");
      return;
   }
   // Kill Any Cookies...
   killObCokies();
   //document.cookie = "ObSSOCookie=loggedout; path=/;
      domain=.cemexnetlab.com"
   document.location.reload(true);

   document.cookie = "<%=obTemp%>";

   myWindowHandle = window.open
      ('about:blank','myWindowName','scrollbars=yes,width=600,height=500');
   loginform.action="/identityredirect/redirector.asp";
//   loginform.action="/identity/oblix/apps/userservcenter/bin/
      userservcenter.cgi?program=modify&usertype=endUser";
   loginform.target="myWindowName";
   loginform.submit();
}
function setAction() {
   if (!((loginform.login.value.length > 0) &&
      (loginform.password.value.length > 0)))
   {
      alert("<%=strMsgUandP(intPointer)%>");
      document.loginform.login.focus();
      return;
   }
   killObCokies();   // Kill Any Cookies...
```

```
      document.cookie = "<%=obTemp%>";

   <%if bolLoginToCOREid then%>
      loginform.action = eval
("URLs["+loginform.selectName.options[loginform.selectName.selectedIndex].value
+"]");
   <%else%>
      loginform.action="/access/oblix/apps/webgate/bin/webgate.dll";
   <%end if%>
      loginform.target="";
      loginform.submit();
   }
   function lost() {
      if (!(loginform.login.value.length > 0)) {
         alert("<%=strMsgU(intPointer)%>");
         return;
      }
      // Kill Any Cookies...
      killObCokies();
      myWindowHandle = window.open
         ('about:blank','myWindowName','scrollbars=yes,width=600,height=500');
      loginform.action="/identity/oblix/apps/lost_pwd_mgmt/bin/
         lost_pwd_mgmt.cgi";
      loginform.target="myWindowName";
      loginform.submit();
   }
-->
</SCRIPT>
</HEAD>


<BODY leftMargin=0 topMargin=0 scrolling="no">
<%if bolLoginToCOREid then%>
   <form name="loginform" action="/identity/oblix/apps/userservcenter/bin/
         userservcenter.cgi?usertype=delegatedIdentityAdminBIZ" method="post">
<%else%>
   <form name="loginform" action="/access/oblix/apps/webgate/bin/webgate.dll"
         method="post">
<%end if%>

<input type="hidden" name="ObLoginDomain" value="dc=oblix,dc=com">

<TABLE cellSpacing=0 cellPadding=0 width=763 align=center border=0>
  <TBODY>
  <TR vAlign=top>
    <TD width="39%" colSpan=2>
      <TABLE cellSpacing=0 cellPadding=0 width="100%" border=0>
        <TBODY>
        <TR>
          <TD vAlign=top width="99%" bgColor=#cc0033>
            <TABLE cellSpacing=0 cellPadding=0 width="100%" border=0>
              <TBODY>
              <TR>
                <TD align=left><B><FONT face="Verdana, Arial, Helvetica,
                                  sans-serif" color=#ffffff size=2>Login</FONT></B>
</TD>
      </TR></TBODY>
             </TABLE>
            </TD>
        </TR>
```

```
          <TR>
            <TD>
              <TABLE cellSpacing=0 cellPadding=0 width=255 border=0>
                <TBODY>
                <TR>
                  <TD align=middle colSpan=3>
                    <TABLE width="90%" border=0>
                      <TBODY>
                      <TR>
                        <TD align=right width="50%"><FONTclass=classBold>

<%=strUser(intPointer)%> </FONT> </TD>
                <TD align=left width="50%">

                    <input type="text" name="login" size="16"
                                          onFocus="nextfield='password';" value="">

                 </TD></TR>
           <TR>
                 <TD align=right width="50%"><%=strPassword(intPointer)%>
                                              </TD>
                       <TD align=left width="50%">

                          <input type="password" name="password"
                                    onFocus="nextfield='done';" value="" size="16">
                      </TD></TR>
                          <%if bolLoginToCOREid then%>
                <TR>
                       <TD align=right width="50%">
<%=strUserType(intPointer)%> 
                       </TD>
                       <TD align=left width="50%">
                 <select name="selectName">
                     <option selected value="0"><%=strCOREidUser(intPointer)%>
                            </option>
                     <option value="1"><%=strCOREidAdmin(intPointer)%>
                            </option>
                 </select>
                       </TD></TR>
                       <%end if%>
                                  <TR>
                                     <TD colSpan=2>
                       <A href="javascript:setAction();"><%=strEnter(intPointer)%>
                                               </A>  
                       <A href="javascript:mySubmit();">
<%=strPreferences(intPointer)%></A>  
                       <A href="javascript:loginform.reset();">

<%=strCancel(intPointer)%></A>  
                       </TD></TR>
                       <TR>
                        <TD colSpan=2> </TD></TR>
                       <TR>
                        <TD align=right colSpan=2>
                        <A href="javascript:changeLanguage();">

<%=strLanguageDescription(intPointer)%></A>
                        </TD></TR>
                      <TR>
                        <TD class=classNormal align=right colSpan=2>
```

```
                        <A href="javascript:lost()"><%=strForgot(intPointer)%></A>

                      </TD></TR></TBODY>
                    </TABLE></TD></TR></TBODY>
                   </TABLE></TD></TR></TBODY>
                  </TABLE></TD>
    <TD width="1%"></TD>
    <TD width="60%">
      <TABLE border=0>
        <TBODY>
        <TR>
          <TD class=classBold width="100%">
            <P>
            <%=strDescription(intPointer)%>
            </TD></TR></TBODY></TABLE>
</FORM>
<DIV id=logoQA><IMG src="login_files/QA.gif">
</DIV>

<SCRIPT LANGUAGE=javascript>
<!--
document.loginform.login.focus();
-->
</SCRIPT>
</BODY>
</HTML>
```

Troubleshooting Form-Based Authentication

For information on troubleshooting, see "Troubleshooting Oracle Access Manager" on page E-1.

B

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 are using 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:** 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 "Configuring Logout for an Identity System Resource" on page 3-29 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 Configuring a Single Sign-On Logout URL on page 2-6 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. See "Logout From a Single Domain Single Sign-On Session" on page 7-8 for details.

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 "AccessGate Configuration Parameters" on page 3-19 for details.

5. Configure the SSO Logout URL.

   See Configuring a Single Sign-On Logout URL on page 2-6 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

When you configure single sign-on between Oracle Access Manager and another product, logging out of the third-party product does 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 logged out of Oracle Access Manager as well.

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 B–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 B–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 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 B–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>
```

C

Oracle Access Manager Parameter Files

Oracle Access Manager provides a simple means for users to modify the way it operates, by changing the content of specified parameter files, also called catalog files. This appendix describes the file format, provides a list of the files, and describes values within them that you can change to customize Oracle Access Manager system operation.

File Categories

All of the parameter files are located relative to the Identity System installation directory, which could be, for example:

on Windows:

```
c:\COREid\identity\oblix
```

on UNIX:

```
/var/COREid/identity/oblix
```

The parameter files can be viewed as belonging to one of several categories, distinguished by the type of parameters they contain:

- Parameters that affect the administrative applications: *User Manager Admin; Group Manager Admin; Organization Manager Admin*.
- Parameters that affect the user applications: *User Manager; Group Manager; Organization Manager; Asynch Mailer; Password Management; Query Builder; Selector*.
- Parameters whose effect is common across many applications: the user applications, the administrative applications and the *Comm Server* (a binary streaming data module).
- Parameters that affect Oracle Access Manager interaction with the directory database (DB), further subcategorized as follows: *user, group, organization, application, configuration, workflow, and LDAP referential integrity* .
- Parameters that affect Oracle Access Manager multitier architecture (for example, the WebPass Web application, or the Identity Server engine).

For More Information on the Parameter Files

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

D

Using Oracle Access Manager with IPv6 Clients

This appendix provides the following information about using Oracle Access Manager with Internet Protocol Version 6 clients:

- About Oracle Access Manager and IPv6
- Prerequisites
- Configuring IPv6 with Simple Authentication
- Configuring IPv6 with an Authenticating WebGate and Challenge Redirect
- Configuring IPv6: Separate Proxy for Authentication and Resource WebGates

About Oracle Access Manager and IPv6

Oracle Access Manager supports Internet Protocol Version 4 (IPv4). Oracle Fusion Middleware supports Internet Protocol Version 4 (IPv4) and Internet Protocol Version 6 (IPv6).

Among other features, IPv6 supports a larger address space (128 bits) than IPv4 (32 bits), providing an exponential increase in the number of computers that can be addressable on the Web. IPv6 is enabled with Oracle HTTP Server with the mod_wl_ohs plug-in.

You can configure Oracle Access Manager to work with clients that support IPv6 by setting up a reverse proxy server. Several scenarios are provided here. Be sure to choose the right configuration for your environment.

- Supported Topologies
- Simple Authentication with IPv6
- Configuring IPv6 with an Authenticating WebGate and Challenge Redirect
- Considerations

Supported Topologies

The following topologies are supported with using IPv6 alone or in combination with IPv4 and various Oracle applications:

Note: Only item 6 pertains to Oracle Access Manager, as one of the single sign-on (SSO) solutions.

1. IPv4 (Oracle Database) -- Dual Stack (WebLogic Server) -- IPv4 Client, IPv6 Client
2. IPv4 (Oracle Database) -- Dual Stack (WebLogic Server, SOA Suite, BAM, WebCenter, Enterprise Manager) -- IPv6 (Oracle HTTP Server with mod_wl_ohs) -- IPv6 Client
3. IPv6 (Database like MySQL) -- IPv6 (WLS) -- IPv6 Client
4. IPv4 (Database) -- Dual Stack (SOA Suite, BAM, WebCenter, Identity Management, Enterprise Manager) -- IPv4 Client, IPv6 Client
5. IPv4 Classic (Forms, Reports, Discoverer, Portal + 10.1.4.3 SSO) -- Dual Stack (Oracle HTTP Server with mod_proxy) -- IPv6 Client
6. IPv4 (Oracle Access Manager 10.1.4.3 with applications like SOA Suite) -- Dual Stack (Oracle HTTP Server with mod_proxy) -- IPv6 Client
7. IPv4 (WebLogic Server, SOA Suite, BAM, WebCenter, Enterprise Manager) -- Dual Stack (Oracle HTTP Server with mod_wl_ohs) -- IPv6 Client

> **See Also:** *Oracle Fusion Middleware Administrator's Guide* (E10105-01) for more information about IPv6

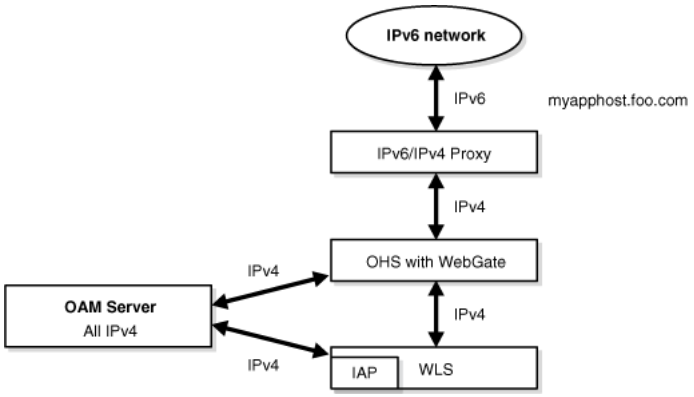
The remainder of this appendix focuses on using IPv6 with Oracle Access Manager.

Simple Authentication with IPv6

Figure D–1 illustrates simple authentication with Oracle Access Manager configured to use the IPv6/IPv4 proxy.

> **Note:** In a WebGate profile, an IPv6 address cannot be specified. In a WebGate profile, the virtual host name must be specified as a host name, for example, *myapphost.foo.com*, not as an IP address.

Figure D–1 Simple Authentication with the IPv6/IPv4 Proxy



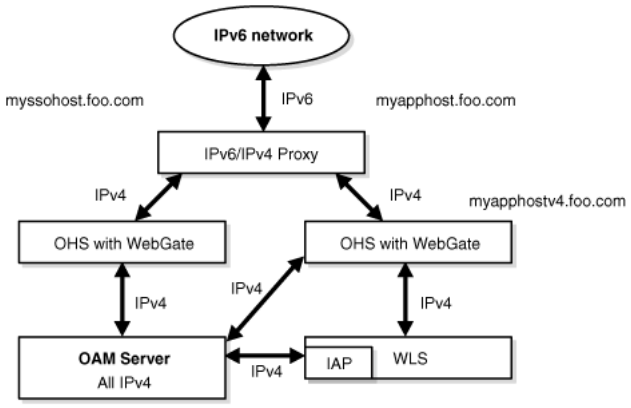
As illustrated in Figure D–1, the IPv6 network communicates with the IPv6/IPv4 proxy, which in turn communicates with the Oracle HTTP Server and WebGate using IPv4. WebGate, Oracle Access Manager servers, and Oracle WebLogic Server with the Authentication provider all communicate with each other using IPV4.

Configuring IPv6 with an Authenticating WebGate and Challenge Redirect

Figure D–2 illustrates configuration with a single IPv6 to IPv4 Proxy (even though *myssohost* and *myapphost* could use separate proxies).

> **Note:** In a WebGate profile, the virtual host name must be specified as a host name, for example, *myapphost.foo.com*, not as an IP address. The redirect host name, for example, *myssohost.foo.com* must also be specified as a host name and not an IP address. The IPv6 address cannot be specified in a WebGate profile.

Figure D–2 IPv6 with an Authenticating WebGate and Challenge Redirect



As illustrated in Figure D–2, the IPv6 network communicates with the IPv6/IPv4 proxy, which in turn communicates with the Oracle HTTP Server using IPv4. WebGate, Oracle Access Manager server, and Oracle WebLogic Server with the Identity Asserter all communicate with each other using IPV4.

You should be able to access the application from a browser on the IPv4 network directly to the IPv4 server host name and have login with redirect to IPv6 *myssohost.foo*.com.

Considerations

The following considerations apply to each intended usage scenario:

- IP validation does not work by default. To enable IP validation, you must add the IP address of the Proxy server as the WebGate's IPValidationException parameter value in the Access System Console.
- IP address-based authorization does not work because all requests come through one IP (proxy IP) that would not serve its purpose.

Prerequisites

Regardless of the manner in which you plan to use Oracle Access Manager with IPv6 Clients, the following tasks should be completed before you start:

- An Oracle HTTP Server instance must be installed to act as a reverse proxy to the Web server (required for WebGate).

- Oracle Access Manager must be installed and initial setup (Identity Server, WebPass, Policy Manager, Access Server, WebGate) as described in other chapters in this manual.

  See Also:

  - *Oracle Fusion Middleware Installation Guide for Web Tier* (E14260-01)
  - *Oracle HTTP Server Administrator's Guide* (E10144-01)

Configuring IPv6 with Simple Authentication

Configuring your environment for simple authentication with Oracle Access Manager using the IPv6/IPv4 proxy is described in the following procedure.

See Also: Figure D–1, "Simple Authentication with the IPv6/IPv4 Proxy"

The configuration in this procedure is an example only. In the example, *OHS_host* and *OHS_port* are the host name and port of the actual Oracle HTTP Server with WebGate. You must use values for your environment.

Note: For this configuration you must use the Web server on which the WebGate is deployed as the Preferred HTTP host in the WebGate profile. You cannot use the IPv6 proxy name.

To configure IPv6 with simple authentication

1. Configure Oracle HTTP Server 11g Release 1 (11.1.1) or any other server to enable reverse proxy:

   a. Stop Oracle HTTP Server with the following command:

   ```
   opmnctl stopproc ias-component=HTTP_Server
   ```

   b. Edit the following file

   ```
   UNIX: ORACLE_INSTANCE/config/OHS/ohs_name/httpd.conf
   Windows: ORACLE_INSTANCE\config\OHS\ohs_name\httpd.conf
   ```

   c. Append the following to the httpd.conf file:

   ```
   #---Added for Mod Proxy
   <IfModule mod_proxy.c>

   ProxyRequests Off
   ProxyPreserveHost On

   ProxyPass /http://OHS_host:OHS_port/
   ProxyPassReverse /http://OHS_host:OHS_port/

   </IfModule>
   ```

   d. Restart Oracle HTTP Server with the following command:

   ```
   opmnctl startproc ias-component=HTTP_Server
   ```

2. Log in to the Access System Console. For example:

   ```
   http://hostname:port/access/oblix
   ```

where *hostname* refers to computer that hosts the WebPass Web server; port refers to the HTTP port number of the WebPass Web server instance; /access/oblix connects to the Access System Console.

The Access System main page appears.

3. Click **Access System Configuration**, and then click **AccessGate Configuration**.

   The Search for AccessGates page appears. The Search list contains a selection of attributes that can be searched. Remaining fields allow you to specify search criteria that are appropriate for the selected attribute.

4. Select the search attribute and condition from the lists (or click the All button to find all AccessGates), and then click Go.
5. Click an AccessGate's name to view its details.
6. Click the Modify button.
7. **Preferred HTTP Host**: Specify the Web server name on which WebGate is deployed as it appears in all HTTP requests. The host name within the HTTP request is translated into the value entered into this field regardless of the way it was defined in a user's HTTP request.
8. To enable IP validation, add the IP address of the proxy server as the value of the **IPValidationException** parameter.
9. Click Save.

Configuring IPv6 with an Authenticating WebGate and Challenge Redirect

Use the following procedure to configure your environment to use Oracle Access Manager with the IPv6/IPv4 proxy and an authenticating WebGate and challenge redirect.

> **See Also:** Figure D–2, "IPv6 with an Authenticating WebGate and Challenge Redirect"

The following procedure presumes a common proxy for both form-based authentication and the resource WebGate. For example, suppose you have the following configuration:

Resource WebGate is installed on http://*myapphostv4.foo*.com/
Resource is on http://*myapphostv4.foo*.com/testing.html

Authenticating WebGate is on http://*myssohostv4.foo*.com/
Login form is http://*myssohostv4.foo*.com/oamsso/login.html

Reverse Proxy URL is http://*myapphost.foo*.com/

Note: For this configuration, the Preferred HTTP host must be the name of the Oracle HTTP Server Web server that is configured for this WebGate. For instance, a WebGate deployed on *myapphost4.foo*.com must use *myapphost4.foo*.com as the Preferred HTTP host. You cannot use the IPv6 proxy name.

The proxy configuration to be used for this example is described in the following procedure. You must configure the Oracle HTTP Server, configure WebGate profiles to use the corresponding Oracle HTTP Server as the Preferred HTTP host, and configure the form-based authentication scheme with a challenge redirect value of the reverse proxy server URL (`http://myapphost.foo.com/` in this example).

Be sure to use values for your own environment.

To configure IPv6 with an authenticating WebGate and challenge redirect

1. Configure Oracle HTTP Server 11g Release 1 (11.1.1) or any other server, as follows:

   a. Stop Oracle HTTP Server with the following command:

   ```
   opmnctl stopproc ias-component=ias-component=HTTP_Server
   ```

   b. Edit the following file

   ```
   UNIX: ORACLE_INSTANCE/config/OHS/ohs_name/httpd.conf
   Windows: ORACLE_INSTANCE\config\OHS\ohs_name\httpd.conf
   ```

   c. Append the following information for your environment to the httpd.conf file. For example:

   ```
   <IfModule mod_proxy.c>
   ProxyRequests On
   ProxyPreserveHost On
   #Redirect login form requests and redirection requests to Authentication
   WebGate

   ProxyPass /obrareq.cgi     http://myssohostv4.foo.com/obrareq.cgi
   ProxyPassReverse /obrareq.cgi    http://myssohostv4.foo.com/obrareq.cgi

   ProxyPass /oamsso/login.html   http://myssohostv4.foo.com/oamsso/login.html
   ProxyPassReverse /oamsso/login.html http://myssohostv4.foo.com/oamsso/login
   .html

   ProxyPass /access/sso     http://myssohostv4.foo.com/ /access/sso
   ProxyPassReverse /access/sso http://myssohostv4.foo.com/access/sso

   # Redirect resource requests to Resource WG
   ProxyPass /http://myapphostv4.foo.com /
   ProxyPassReverse /http://myapphostv4.foo.com /

   </IfModule>
   ```

   d. Restart Oracle HTTP Server with the following command:

   ```
   opmnctl startproc ias-component=ias-component=HTTP_Server
   ```

2. In the Access System Console, set the Preferred HTTP host for each WebGate as follows:

   a. Log in to the Access System Console. For example:

   ```
   http://hostname:port/access/oblix
   ```

   where *hostname* refers to computer that hosts the WebPass Web server; port refers to the HTTP port number of the WebPass Web server instance; /access/oblix connects to the Access System Console.

   The Access System main page appears.

b. Click **Access System Configuration**, and then click **AccessGate Configuration**.

   The Search for AccessGates page appears. The Search list contains a selection of attributes that can be searched. Remaining fields allow you to specify search criteria that are appropriate for the selected attribute.

c. Select the search attribute and condition from the lists (or click the All button to find all AccessGates), and then click Go.

d. Click an AccessGate's name to view its details.

e. Click the Modify button.

f. **Preferred HTTP Host**: The name of the Oracle HTTP Server Web server that is configured for this WebGate. For instance, a WebGate deployed on *myapphostv4.foo*.com must use *myapphostv4.foo*.com as the Preferred HTTP host.

g. To enable IP validation, add the IP address of the Proxy server as the value of the **IPValidationException** parameter.

h. Click Save.

i. Repeat for each WebGate and specify name of the Oracle HTTP Server Web server that is configured for this WebGate.

3. From the Access System Console, modify the Form authentication scheme to include a challenge redirect to the Proxy server, as follows:

   a. Click **Access System Configuration**, and then click **Authentication Management**.

   b. Click the name of the scheme to modify, and then click Modify.

   c. Configure the challenge redirect value to the Proxy server URL. In this example, the Proxy server URL is `http://myapphost.foo.com/`

   d. Click Save.

Configuring IPv6: Separate Proxy for Authentication and Resource WebGates

In this configuration you have multiple proxies: for example a separate proxy for the authentication WebGate and another proxy for the resource WebGate. You can access the application from a browser on the IPv4 network directly to an IPv4 server host name with a login redirect to an IPv6 host. For example:

Resource WebGate is on http://*myapphostv4.foo*.com/
Authenticating WebGate is on http://*myssohostv4.foo*.com

Proxy used for *myapphostv4.foo*.com should be *myapphostv4.foo*.com
Proxy used for *myssohostv4.foo*.com should be *myssohostv4*.com

---

Note: You cannot use the IPv6 proxy name as the Preferred HTTP host in a WebGate profile.

In the example, *OHS_host* and *OHS_port* are the host name and port of the actual Oracle HTTP Server that is configured for WebGate. Be sure to use values for your own environment.

To configure IPv6 with a separate proxy for authentication and resource WebGates

1. Configure Oracle HTTP Server 11g Release 1 (11.1.1) or any other server for multiple proxies, as follows:

   a. Stop Oracle HTTP Server with the following command:

   ```
   opmnctl stopproc ias-component=ias-component=HTTP_Server
   ```

   b. Edit the following file

   ```
   UNIX: ORACLE_INSTANCE/config/OHS/ohs_name/httpd.conf
   Windows: ORACLE_INSTANCE\config\OHS\ohs_name\httpd.conf
   ```

   c. Append the following information for your environment to the httpd.conf file. For example:

   ```
   <IfModule mod_proxy.c>
   ProxyRequests Off
   ProxyPreserveHost On

   ProxyPass /http://OHS_host:OHS_port
   ProxyPassReverse /http://OHS_host:OHS_port

   </IfModule>
   ```

   d. Restart Oracle HTTP Server with the following command:

   ```
   opmnctl startproc ias-component=ias-component=HTTP_Server
   ```

2. In the Access System Console, set the Preferred HTTP host for each WebGate as follows:

   a. Log in to the Access System Console. For example:

   ```
   http://hostname:port/access/oblix
   ```

   where *hostname* refers to computer that hosts the WebPass Web server; port refers to the HTTP port number of the WebPass Web server instance; /access/oblix connects to the Access System Console.

   The Access System main page appears.

   b. Click **Access System Configuration**, and then click **AccessGate Configuration**.

   The Search for AccessGates page appears. The Search list contains a selection of attributes that can be searched. Remaining fields allow you to specify search criteria that are appropriate for the selected attribute.

   c. Select the search attribute and condition from the lists (or click the All button to find all AccessGates), and then click Go.

   d. Click an AccessGate's name to view its details.

   e. Click the Modify button.

   f. **Preferred HTTP Host**: The name of the Oracle HTTP Server Web server that is configured for this WebGate. For instance, a WebGate deployed on

myapphostv4.foo.com must use *myapphostv4.foo*.com as the Preferred HTTP host.

g. To enable IP validation, add the IP address of the Proxy server as the value of the **IPValidationException** parameter.

h. Click Save.

i. Repeat for each WebGate and specify name of the Oracle HTTP Server Web server that is configured for this WebGate.

3. From the Access System Console, modify the Form authentication scheme to include a challenge redirect to the Proxy server, as follows:

a. Click **Access System Configuration**, and then click **Authentication Management**.

b. Click the name of the scheme to modify, and then click Modify.

c. Configure the challenge redirect value to the Proxy server URL that acts as a reverse proxy for the authentication WebGate. In this example, the Proxy server URL is `http://myssohost.foo.com/`

d. Click Save.

E

Troubleshooting Oracle Access Manager

This appendix explains typical problems that you could encounter while running or installing Oracle Access Manager. It contains these sections:

- Problems and Solutions
- Capturing Diagnostic Information
- Need More Help?

Problems and Solutions

This section describes common Oracle Access Manager error messages, problems and solutions. It contains the following topics:

- Access Server Issues
- Authentication and Authorization Issues
- Cache Flush Issues with Active Directory
- Directory Server Issues
- File Ownership and Command Line Tools
- Form-Based Authentication Issues
- Policy Manager Hangs During Cache Flush from Policy Manager to Access Server
- Single Sign-On Issues
- WebGate Issues

Access Server Issues

This section discusses the following issues:

- Access Server Hangs on Windows 2003
- The Access Server Is Not Sending Audit Data to the Database

Access Server Hangs on Windows 2003

On Microsoft Windows 2003, an Access Server or an Identity Server that has cache flushes to the Access Server may hang. This problem may be due to security changes in Microsoft Windows 2003 that cause the queue (Q) parameter that is sent to the Access Server to not take effect. The inability to set the number of queues leads to an insufficient number of available threads for the Access Server.

Requests are queued as they are sent to an Access Server. A thread processes each request. For example, if you have two request queues and 60 threads, the Access Server spawns 120 threads. The number of queues determines how many sets of threads are created for the Access Server process and determines the processing power of the Access Server.

Check to see if the `Q` parameter is being used in the startup command. You specify this parameter in the Services panel Start parameters field or via a registry key in the `ImagePath` parameter, as follows:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\ObAAAServer-Access_Server_ID
```

If you use the queue (`Q`) parameter in your startup command, multiply the current number of threads in the Access Server configuration by the queue size, then update the number of threads using the result. For example, if you currently use 2 queues with 30 threads in the Access Server, change the number of threads to 60. You set this parameter from the Access Server configuration details page.

To configure the number of threads for the Access Server

1. Launch the Access System Console.
2. Click Access System Configuration, then select Access Server Configuration.

   The existing Access Servers are listed on the page.
3. Select an Access Server to view its configuration.

   The configuration details of the Access Server appear.
4. In the Number of Threads field, configure the maximum number of threads allowed for the Access Server.

The Access Server Is Not Sending Audit Data to the Database

The auditing feature for the Access Server is not working when auditing to a database. However, file-based auditing is working.

Problem

This problem can occur for various reasons when creating an RDBMS profile, as described in the chapter on configuring global settings in the in the *Oracle Access Manager Identity and Common Administration Guide*.

You may discover the problem after doing the following:

1. Create a new database instance and create an oblix_audit_events table in it, as specified in the chapter on auditing in the *Oracle Access Manager Identity and Common Administration Guide*.
2. Create an RDBMS profile in System Console, as specified in the chapter on auditing in the *Oracle Access Manager Identity and Common Administration Guide*.
3. In the Access System Console, click Access System Configuration, then click Access Server Configuration, then click the link for the Access Server that you want to modify.
4. Click Modify and ensure that you have turned on auditing to database and to a file.
5. In the Access System Console, click Access System Configuration, then click Common Information Configuration, then click Master Audit Rule.

Select the authentication success, authentication failure, authorization success and authorization failure events in the master audit rule.

6. Log in to Oracle Access Manager through a WebGate and check whether authentication and authorization success and failure events are being logged.

   For example, you can check if a message, "The database audit writer was unable to perform the write," is logged in the Access Server's oblog.log file at the ERROR level.

   You can also run the following command in either the SQL Server's Query Analyzer window or Oracle's iSQL*Plus window:

   ```
   select * from oblix_audit_events
   ```

   This command displays a list of existing audit records in oblix_audit_events table. If the list does not contain the new authentication and authorization events, it means that auditing to the database has failed.

Solution

The problem often occurs because of an incorrect value for the SQLDBType parameter in the following file:

AccessServer_install_dir/identity/apps/common/bin/globalparams.xml

Where *AccessServer_install_dir* is the location where the Access Server was installed. Set the value for the SQLDBType parameter as follows:

- For an ODBC connection type, set the value to `Oracle`.
- For an OCI connection type, set the value to `Oracle_OCI`.
- For SQL Server database, set the value to `SQLServer`.

Authentication and Authorization Issues

This section discusses the following issues

Authentication Scheme Is Well Formed but User is Not Authenticated

The authentication scheme appears to be well-formed, and requests are being forwarded to the Access Server, but the user is not being authenticated.

Problem

After configuring an authentication scheme and testing it using the Access Tester, it appears that the scheme should work properly. However, users are not being authenticated when they should be. Debug logs of the directory server do not show any sign of the Access Server performing a search in the directory server.

Solution

When adding the credential mapping plug-in to an authentication scheme, be sure that the credential_mapping plug-in is placed before the validate_password plug-in. The authentication scheme must process the plug-in that validates the attribute for the login ID before validating the password.

Oracle Access Manager Fails While Authenticating (User Data in Netscape/Sun Directory Server)

Problem

Oracle Access Manager fails while authentication a user when user data is stored in Netscape/Sun Directory Server and the login attribute is indexed using index ldif import.

Cause

This is an issue with the Netscape/Sun Directory Server. To index any attribute, the Sun ONE Sever Console should be used.

Solution

Follow the steps here.

To index an attribute using the Sun ONE Server Console

1. Log in to Sun ONE Server Console, and open related directory server instance.
2. Go to the Configuration tab, expand the Data node, and select the searchbase node (the suffix under which attributes are to be indexed). For example:

   ```
   dc=us,dc=oracle,dc=com
   ```

3. In the right-hand panel, view the list of System Indexes and Additional Indexes.
4. Click the Add Attribute button to add attributes to be indexed and configure these for equality, presence, and substring matches.
5. Click Save which displays a warning.
6. Select Reindex Suffix and select attributes that you have recently added.
7. Click OK.

Problem with Large Authorization Expressions

Problem

You might experience a problem that prevents adding large authorization expressions (for example, if you are using Oracle Internet Directory, the limit is 4000 characters). After adding and saving a large expression, you might see the following message:

```
No Authorization Defined
```

Cause

The source of the problem is a directory server limitation. However, some directory vendors support increasing the maximum attribute length.

Solution

Before you add a large authorization expression, the Policy Manager globalparams.xml file requires the addition of the `policyDSMaxAttrValueLength` parameter. The value that you add should be the maximum limit imposed on an attribute length by the directory server. For Oracle Internet Directory, this value should not exceed 4000 characters. For other directory servers, consult the vendor-specific documentation for further details.

If you are adding the policy using an API in the Access Manager SDK, then you need to add the `policyDSMaxAttrValueLength` parameter to the associated Access Server globalparams.xml file.

> **Note:** If the directory limit cannot be changed or cannot accommodate the size of the expression, the `policyDSMaxAttrValueLength` parameter will instruct Oracle Access Manager to compensate for the limit and allow creation of larger expressions.

To compensate for the limit and allow creation of larger expressions

1. Locate the globalparams.xml file in the following path:

   For changes from the API of Access Manager SDK:

   AccessServer_install_dir/access/oblix/apps/common/bin/globalparams.xml

   For changes from Policy Manager:

   PolicyManager_install_dir/access/oblix/apps/common/bin/globalparams.xml

2. Add the following lines anywhere in the globalparams.xml file:

```
<SimpleList>
   <NameValPair
      ParamName="policyDSMaxAttrValueLength"
      Value="4000">
   </NameValPair>
</SimpleList>
```

3. Restart the Access Server, or the Policy Manager Web server.
4. Add authorization expressions to Oracle Access Manager.

Cache Flush Issues with Active Directory

As described in Chapter 8, an Oracle Access Manager deployment can include multiple instances of the Identity Server, Access Server, and Policy Manager. To ensure behavioral consistency, all changes to a user profile or policy information (or configuration changes for Access Servers) are written to the directory server and must be propagated across all components.

Problem

On Access Server 10*g* (10.1.4.2.0) or later with Active Directory as the backend Directory Server, cache flush operations are not taking effect on the Access System. Additionally, schema violation error logs are observed in the Access Sever log file during cache flush operations.

Cause

A schema limitation in Active Directory.

Solution

With Active Directory as a backend directory server, you must set the value of the `UserMgmtNodeEnabled` parameter to `false` in the Access Server globalparams.xml file.

> **See Also:** The chapter on parameters in the *Oracle Access Manager Customization Guide*

To ensure proper cache synchronization with Active Directory

1. Locate the globalparams.xml file for Access Server:

   AccessServer_install_dir`\access\oblix\apps\common\bin\globalparams.xml`

2. Confirm (or change) the value of the `UserMgmtNodeEnabled` parameter is set to `false`.
3. Restart the Access Server.
4. Repeat these steps for all Access Servers in the deployment.

> **Caution:** With original 10*g* (10.1.4.2.0) release, this was enabled by default. With bundle patch 10*g* (10.1.4.2.0)-07, the default is disabled (`false`) as it was for 10g (10.1.4.01).
>
> With Oracle Access Manager 10*g* (10.1.4.3) (or 10*g* (10.1.4.2.0) bundle patch 07), `UserMgmtNodeEnabled` operates as expected: A value of `true` enables the function and a value of `false` disables it.

Directory Server Issues

This section discusses the following topics:

- Error Message to Check if the Directory Server is Running or Responding
- Memory Usage Rises After Configuring a Directory Server Profile
- Search Halts When Using Active Directory or .Net

Error Message to Check if the Directory Server is Running or Responding

During normal operations, you can receive an error indicating that the directory server may not be operational.

Problem

The Access Server or Policy Manager may issue one of the following error messages:

- "Please verify that the Directory Server is running."
- "Please verify that the Directory Server is responding."

These error messages are generated when the Oracle Access Manager component does not receive a response from the directory server within a user-configurable amount of time.

Solution

The following are possible solutions to the problem:

- Check the value for the LDAPOperationTimeout parameter in globalparams.xml.

This parameter enables the Oracle Access Manager component to fail over to a secondary directory server when the primary one takes too long to respond. See the appendix on parameter files in the *Oracle Access Manager Customization Guide* for details.

- Ensure that failover has been configured for this directory server.

  See the information on configuring directory server profiles in the *Oracle Access Manager Identity and Common Administration Guide*. Also see the chapter on failover in the *Oracle Access Manager Deployment Guide*.

Memory Usage Rises After Configuring a Directory Server Profile

After configuring a directory server profile, the memory usage for the Access Server or Policy Manager becomes too high.

Problem When you configure a directory server profile, you are prompted to provide a maximum session time. The default value for the session time is 0 (unlimited). This may cause a performance issue, because the size of the caches for LDAP connections to the Access Server and Policy Manager increase over time. Oracle Access Manager does not control these caches directly.

Solution To prevent the cache size from causing a performance problem, set the value of the Maximum Session Time (Minutes) for the directory server profile to a finite value, for example, 10 hours, as follows:

1. From the Identity System Console click System Configuration, then click Directory Profiles.
2. Click the link for the profile that you want to modify.
3. In the Max. Session Time (Min.) field, set the value to 600.

Search Halts When Using Active Directory or .Net

When conducting a search, after the Search icon is selected an error page appears stating, "The following messages were produced by the product. Please contact your webmaster to fix the problem." The limitAMPolicyDomainResourceDisplay parameter is set to true in the Policy Manager globalparams.xml file.

Problem

The number of policy domains exceeds the current limit of 350. 400 policy domains were created in the Access System, each with 10 resources and 10 policies.

Solution

Do not exceed 350 policy domains with Active Directory.

File Ownership and Command Line Tools

All command line utilities and tools must be run as the user who installed the product, as described in the *Oracle Access Manager Installation Guide*. Oracle recommends that you do not attempt to change ownership or permissions on files after installation.

Form-Based Authentication Issues

This section discusses the following issues:

- The Login Form Appears Repeatedly

The Login Form Appears Repeatedly

The login form repeatedly reappears after the user enters credentials.

Problem

The login form may continue to pop up due to the configuration of the parameters for login credentials, the configuration of the authentication plug-ins, and the configuration of the authentication scheme.

Solution

One of the following solutions can be applied to this problem.

Make sure the credentials in the creds challenge parameter for the form scheme match the input fields in the form.

Make sure the authentication plug-ins for the form scheme are correct.

If you are using IIS and the form action is not the webgate.dll, make sure the WebGate filter post processing is enabled by the Registry entry

```
HKEY_LOCAL_MACHINE\SOFTWARE\Oblix\Oblix COREid\version\WebGate\postdata="yes"
```

where *version* is the version number of the installed Oracle Access Manager product.

To make sure that the authentication scheme is set properly, you can attempt to access a resource protected with that authentication scheme, adding the credentials as query string parameters. This simulates a form whose method is GET without actually using the form.

For example, suppose the authentication scheme uses the following creds challenge parameter:

```
creds:login password
```

In this example, if the protected URL is http://server/protected/page.html, you could launch a browser instance and type the following:

```
http://server/protected/page.html?login=jsmith&password=MyPwd
```

If jsmith and MyPwd are valid credentials, after you press Enter the page is displayed instead of the login form if the authentication scheme is working correctly but something is wrong in the form's HTML code or in the registry (in the case of IIS servers).

To verify whether a user has a valid session, you can type the following in the browser's location:

```
javascript:alert(document.cookie)
```

The window that pops up should contain the current cookies associated with the browser, including the ObSSOCookie. This can also help determine if the cookie domain or invalid logout situations are affecting the login process.

302 response with Apache/Oracle HTTP Server WebGate

In either of the following situations, a 302 response appears in the Web browser when WebGate is installed with the Apache/Oracle HTTP Server 1.3.x Web server and you are accessing a resource that is protected with a form-based authentication scheme:

- Either a challenge redirect is used
- Or a reverse-proxy server is used to hide the Web server protected by WebGate

Problem

The 302 response contains the Location:url that is used, per the HTTP protocol, for the next request. The browser displays the following message:

```
Found

The document has moved here
```

Cause

The extra data is being populated by Apache/Oracle HTTP Server 1.3.x because of a protocol mismatch between the Proxy Server (or client) and Apache/Oracle HTTP Server. Oracle HTTP Server is based on Apache 1.x.x and the keepalive and HTTP/1.1 on Apache 1.3.27 are not properly implemented. In this case, the data is being transferred to the Proxy Server/client (browser) in the form of chunks. All the chunks are being collected in a temporary cache and the cache link is shown to the user.

Workarounds

There are several workarounds. Item #2 is the best possible solution and is Oracle's recommendation. There is a performance penalty with using HTTP 1.0 as opposed to HTTP 1.1. Workarounds 3, 4, and 5 should be used only if workarounds #1 and 2 are not feasible or do not work. Choose the workaround that is best for your environment:

1. Upgrade the APACHE/Oracle HTTP Server 1.3.X to a higher version.
2. Add the following line to the httpd.conf file of the WebGate Web server:

   ```
   ErrorDocument 302 "
   ```

3. Add BrowserMatch directives in the "mod_setenvif" module of the httpd.conf file of the WebGate Web server to disable the keepalive for particular browsers.
4. Use two environment variables in the reverse-proxy httpd.conf file for protocol adjustments that can force the request to use HTTP/1.0 with no keepalive. For example:

   ```
   1)force-proxy-request-1.0
   2)proxy-nokeepalive
   ```

5. Configure your client browser setting to use the HTTP /1.0 protocol.

Other Form-Based Authentication Issues

After submitting a login form, you receive an error or the login form stops responding.

Problem

After you submit the login form, one or more of the following messages appears:

- 500 Internal Server Error
- You receive an new login challenge (for example, a basic login dialog box)

If the form stops responding, the redirection action may be misconfigured.

Solution

If you receive an error message, ensure that the form's action URL is protected by a policy domain, and ensure that the action challenge parameter of the form scheme matches the form action URL.

> **Note:** Because of the way Access System updates the Access Manager SDK caches that the WebGate uses, a corrected authentication scheme is not available until after that WebGate has made another request to the Access Server. Consequently, a form login problem may occur one more time after the correction.

If the form stops responding after successful authentication, be sure that the redirection action does not send the user back to the same login form.

Policy Manager Hangs During Cache Flush from Policy Manager to Access Server

Problem

This problem can occur in a new installation with default settings when you have multiple Access Servers configured with a single directory server. If one Access Server hangs during a cache flush request from the Policy Manager, the Policy Manager can hang when the Update Cache option is enabled and any parameter is modified and saved using the Policy Manager console. For instance, if you change a host identifier or Policy Domain, or URL prefix with the Update Cache option enabled.

Soution

Add the `CacheFlushTimeOut` parameter in the Policy Manager globalparams.xml file. For details, see "Configuring Synchronous Cache Flush Requests Between Multiple Access Servers with a Wait Period" in the *Oracle Access Manager Deployment Guide*.

Single Sign-On Issues

This section discusses the following issues

- Error with Single Sign-On Between Identity and Access Systems
- Other Single Sign-On Problems

Error with Single Sign-On Between Identity and Access Systems

After logging in to one system (for example, the Access System) you may receive an error message similar to the following when you try to access the other system (for example, the Identity System):

The Identity Server logged you in but the Access System (Policy Manager or System Console) logged you out.

Problem

This may be due to one or more of the following problems:

- Identity and Access Servers are running on different computers, and the clocks are set to a different time.
- You have protected the Identity System in a policy domain, but not the Access System, or visa versa.

- The loginslack parameter in the oblixbaseparams.xml file is not configured correctly.

Solution

Apply one or more of the following solutions:

- Synchronize the Identity and Access Servers system clocks.
- Ensure that policy domains have been created for both the Identity System and the Access System.

  See "Protecting Resources with Policy Domains" on page 4-1 for details.
- Open the following file and edit the value for the `loginslack` parameter:

  `PolicyManager_install_dir``/access/oblix/apps/common/bin/oblixbaseparams.xml`

  The `loginslack` parameter controls the time difference that is tolerated between the Policy Manager host computer and the Identity host computer. This parameter affects the WebPass, which controls single sign-on between the Policy Manager and the Identity System. It does not affect single sign-on provided by WebGate. The `loginslack` parameter is useful only if the WebGate is not protecting the Policy Manager or WebPass, and WebGate is not being used for single sign-on. In this type of scenario, the Policy Manager and WebPass use a cookie for login and single sign-on that is different from the ObSSOCookie. This cookie has time stamp.

  The default value is 60 seconds.

Other Single Sign-On Problems

Users can experience problems with single sign-on, including:

- Unexpected session timeout.
- Single sign-on failure (login prompts always are presented).
- Authentication fails.
- Users are authenticated initially, but their authorization fails when they access a resource on a second host.
- After authenticating to a protected Web site, single sign-on does not work when accessing a second site that has the same authentication level.

Problem

The following may be causes for one or more problems:

- Unexpected session timeout: the session timeout parameters are not configured correctly, or the system clocks on the hosts are not synchronized.
- Single sign-on failure: The cookie definition may contain the wrong domain name, or the WebGates may not have the same primary HTTP cookie domain or shared secret.
- The user's authentication fails: The user's identity or domain name does not match the authentication rules specified for the domain, or the authentication schemes to enable multi-domain single sign-on do not specify Challenge Redirect.
- Users authorization fails when they access a resource on a second host: The authentication scheme configured for the second host be higher scheme than the one on the first host, or the shared secret may have been corrupted.

Solution

One or more of the following solutions may fix the problem:

- Timeout problem:

  Increase the value of the session timeout parameters Maximum User Session Time and Idle Session Time. See AccessGate Configuration Parameters on page 3-19 for details.

  Synchronize the system clocks on each host involved in single sign-on.

  If the system clocks on the hosts are not synchronized, the cookie may be in the future or in the past and the session timeout rule may be triggered even though in reality there is no timeout issue

- Single sign-on failure:

  Check the definitions for the ObSSOCookie. The cookie definition may contain the wrong domain name.

  Be sure that both WebGates have the same primary HTTP cookie domain.

  Be sure the two WebGates are in configured for the same Access Server.

- The user's authentication fails.

  Be sure this user's identity matches the authentication rules specified for the domain.

  Also be sure the user supplied a fully qualified domain name. You can configure multiple ways for a user to specify the fully qualified domain name. See "Using Host Identifiers and Host Contexts" on page 4-31 for details.

  Finally, verify that Challenge Redirect is set on the authentication schemes to enable multi-domain single sign-on.

- Users are authenticated initially, but their authorization fails when they access a resource on a second host.

  The authentication rule configured for the second host could deny the requester access to the resource. A user can go from a higher scheme to a lower scheme, but not from a lower one to a higher one. For example, if a user is authenticated to access a resource that requires a Basic Over LDAP authentication scheme, that user can access other resources having the same or a less stringent scheme. However, if the same user tries to access a resource with a more stringent authentication challenge, such as Client Certificate, the user must re-authenticate. Even if a user tries to access a resource protected by the same scheme (but with a higher level authentication challenge), the user must re-authenticate.

- Single sign-on does not work when accessing a second site that has the same authentication level.

  The shared secret may have been corrupted. Regenerate the shared secret and restart the Web servers and Access Servers.

WebGate Issues

This section discusses the following issues:

- Invalid or Missing Preferred HTTP Host Identifier in WebGate Profile
- WebGate Diagnostics URL Incorrectly Report that the Access Server Is Down
- WebGate Cannot Connect to its Associated Access Server

- Mismatch Between Content and Content Length in a Web Server 401 Response

Invalid or Missing Preferred HTTP Host Identifier in WebGate Profile

New parameters are provided that you can use to leave the Preferred HTTP Host field empty, and to validate the entry in the field to ensure there are no errors:

- `AllowEmptyPreferredHost`: When the value is `true`, the Preferred HTTP Host field can be empty in a WebGate configuration profile in the Access System Console.

  When the value is `false` (or absent by default) the Preferred HTTP Host field cannot be empty. If it is empty, an error occurs when you save the profile.

- `PreferredHostValidityCheckEnabled`: When the value is `true` (or not present by default) the value in the Preferred HTTP Host field is validated to catch any typographical errors.

> **See Also:** Knowledge base article 416329.1 on Oracle Metalink at `https://metalink.oracle.com`. This note provides guidelines for deciding if you want to use the Preferred HTTP Host feature, and how to bypass it if you decide that enabling virtual hosting is more important in your environment than using the preferred host.

To allow an empty Preferred HTTP Host field or check the validity of the value

1. Locate the wrsc_admin.js file and remove the validation from this script
2. Locate the Policy Manager globalparams.xml file in:

   PolicyManager_install_dir/access/oblix/apps/common/bin/

3. **Allow an Empty Preferred HTTP Host Field**: Set `AllowEmptyPreferredHost` to `true` and save the file:

```
<SimpleList>
   <NameValPair
      ParamName="AllowEmptyPreferredHost" Value="true">
   </NameValPair>
</SimpleList>
```

4. **Enable a Validity Check of the Preferred HTTP Host**: Set `PreferredHostValidityCheckEnabled` to `true` and save the file:

```
<SimpleList>
   <NameValPair
      ParamName="PreferredHostValidityCheckEnabled" Value="true">
   </NameValPair>
</SimpleList>
```

WebGate Diagnostics URL Incorrectly Report that the Access Server Is Down

The WebGate diagnostics URL reports the status of the Access Server or Servers to which the WebGate is connected. In some cases, the landing page for this URL can report that the Access Server or Servers are down when in the servers actually are running.

Problem

This problem occurs when the number of Access Servers that are associated with a WebGate is higher than the value of WebGate's Maximum Connections property. In

this type of situation, the WebGate diagnostics page displays a status of Down for all Access Servers that exceed the Maximum Connections irrespective of their status.

For example, suppose that you set the Maximum Connections value for WebGate A to 1 and you associate three Access Servers with it, AAA1, AAA2, and AAA3. The diagnostics page will indicate that AAA1 is up and AAA2 and AAA3 are down. If AAA1 is down, the page will indicate that AAA2 is up and AAA3 is down.

Solution

To fix this problem, ensure that there are more connections configured between the WebGate and the Access Servers than there are Access Servers. See "Viewing AccessGate Profiles" on page 3-17 and "AccessGate Configuration Parameters" on page 3-19 for details.

WebGate Cannot Connect to its Associated Access Server

If you have installed a WebPass or a WebGate on IIS 6 and enabled logging, the WebPass or WebGate may be unable to connect to its associated Identity or Access Server.

Problem

This problem occurs when you send logs to an MPFileLogWriter. It does not occur when you send logs to a FileLogWriter.

The problem occurs with the MPFileLogWriter when there is no anonymous user with access to the directory that contains the log files. MPFileLogWriter uses a file named <*logfile name*>.lck to synchronize multiple processes that write to the corresponding log file. The MPFileLogWriter write-locks the.lck file before writing to the oblog.log file.

Solution

Configure an anonymous user with access to the directory that contains the log files. In some circumstances, the user context used to acquire the write-lock will be the IIS Anonymous Web user. By default, this user is named `IUSR_<computer name>`, but you can configure any anonymous user for this purpose.

Mismatch Between Content and Content Length in a Web Server 401 Response

It is possible that a Web server will send informative data with a 401 Web server response. A mismatch between the content and content length result in either no data displayed in the browser or an error message in the browser. After accessing the product URL with valid credentials on Mozilla, the following error is displayed:

```
HTTP Error 401
401.1 Unauthorized: Logon Failed
This error indicates that the credentials passed to the server do not
match the credentials   required to log on to the server.
Please contact the Web server's administrator to verify that you have
permission to access the requested resource
```

However, when the browser is refreshed, the product home page is displayed.

With Miscosoft Internet Explorer, if the product URL is accessed with valid credentials a blank page is displayed, but after refreshing the page Product home page is displayed.

To configure WebGate to work with particular browsers, proxies, and so on, you can set specific user-defined parameters. A new WebGate user-defined parameter can be used to set the Content-Length to 0 for all 401 responses:

Parameter: `ContentLengthFor401Response`

Value: `0` (any other value will be ignored)

Note: WebGate is affected and must be patched.

To add the ContentLengthFor401Response parameter to WebGate configuration

1. Launch the Access System Console and click Access System Configuration.
2. Click AccessGate Configuration.
3. Enter your search criteria for the WebGate, and then click Go.
4. In the Search Results table, click a WebGate name.
5. At the bottom of the Details for AccessGate page, click Modify.
6. On the Modify AccessGate page, locate the User Defined Parameters section of the page, enter the following parameter, and value, and then click the Add button:

   Parameter: `ContentLengthFor401Response`

   Value: `0`
7. Click the Add button if you want to add more user-defined parameters.
8. Save to save this new information.
9. Repeat for each WebGate in your deployment.

Capturing Diagnostic Information

Oracle Access Manager supports the capture of core dump information to the log files. You can also run and capture stack trace information. See the troubleshooting appendix in the *Oracle Access Manager Identity and Common Administration Guide* for details.

Need More Help?

You can find more solutions on My Oracle Support (formerly MetaLink), https://metalink.oracle.com. If you do not find a solution for your problem, log a service request.

Index

A

B

C

D

E

F

G

H

I

L

M

N

O

P

R

S

T

U

V

W