# Oracle® Access Manager

Deployment Guide

10*g* (10.1.4.0.1)

**B25344-01**

July 2006

This guide provides capacity planning, performance tuning, failover, load balancing, caching, and migration planning for people who plan and manage the environment.

ORACLE®

Oracle Access Manager Deployment Guide, 10*g* (10.1.4.0.1)

B25344-01



Primary Author: Nina Wishbow

Contributing Author: Gail Tiberi

# Contents

## 3 Failover and Load Balancing



## 4 Caching and Cloning

## 5 Reconfiguring the System



## 6 Synchronizing System Clocks Across Time Zones

# Preface

This Deployment Guide provides information for people who plan and manage the environment in which Oracle Access Manager will run. This guide covers capacity planning, network topologies and system tuning.

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

This Preface covers the following topics:



## Audience

This guide targets the knowledge and skill requirements of system, network, or Oracle Access Manager administrators who are responsible for optimizing the implementation.

This document assumes that you are familiar with your network architecture, your LDAP directory, and firewall and internet security.

## Documentation Accessibility

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

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

### Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

### Accessibility of Links to External Web Sites in Documentation

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

### TTY Access to Oracle Support Services

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

## Related Documents

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

- *Oracle Access Manager Introduction*—Provides an introduction to Oracle Access Manager, a road map to the manuals, and a glossary of terms.
- *Oracle Application Server Release Notes*—Read these for the latest Oracle Access Manager updates. The release notes are available with the platform-specific documentation. The most current version of the release notes is available on Oracle Technology Network at: `http://www.oracle.com/technology/documentation`.
- *Oracle Access Manager Installation Guide*—Describes how to install and set up the Oracle Access Manager components.
- *Oracle Access Manager Upgrade Guide*—Explains how to upgrade earlier versions of Oracle Access Manager to the latest version.
- *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 operation by making changes to operating systems, Web servers, directory servers, directory content, or by connecting CGI files or JavaScripts to Oracle Access Manager screens. This guide also describes the Access Manager API and the authorization and authentication plug-in APIs.
- *Oracle Access Manager Developer Guide*—Explains how to access Identity System functionality programmatically using IdentityXML and WSDL, how to create custom WebGates (known as AccessGates), and how to develop plug-ins. This guide also provides information to be aware of when creating CGI files or JavaScripts for Oracle Access Manager.
- *Oracle Access Manager Integration Guide*—Explains how to set up Oracle Access Manager to run with third-party products such as BEA WebLogic, the Plumtree portal, and IBM Websphere.
- *Oracle Access Manager Schema Description*—Provides details about the schema.
- Also, read the Oracle Application Server Release Notes for the latest updates. The release notes are available with the platform-specific documentation. The most current version of the release notes is available on Oracle Technology Network (`http://www.oracle.com/technology/documentation`).

## 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 Oracle Access Manager 10*g* (10.1.4.0.1) and provides pointers to additional information within this book. Information from previous releases is also retained to help those users migrating to the current release.

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

- Product and Component Name Changes
- Tuning the Directory
- Tuning the Access Server
- Tuning Workflows
- Tuning Your Network
- Failover and Load Balancing
- Reconfiguring Oracle Access Manager

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

## Product and Component Name Changes

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

| Item | Was | Is |
|---|---|---|
| Product Name | Oblix NetPoint<br>Oracle COREid | Oracle Access Manager |
| Product Name | Oblix SHAREid<br>NetPoint SAML Services | Oracle Identity Federation |
| Product Name | OctetString Virtual Directory Engine (VDE) | Oracle Virtual Directory |
| Product Release | Oracle COREid 7.0.4 | Also available as part of Oracle Application Server 10g Release 2 (10.1.2). |

| Item | Was | Is |
|---|---|---|
| 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<br>COREid Administrator | Master Administrator |
| Directory Tree | Oblix tree | Configuration tree |
| Data | Oblix data | Configuration data |
| Software Developer Kit | Access Server SDK<br>ASDK | Access Manager SDK |
| API | Access Server API<br>Access API | Access Manager API |
| API | Access Management API<br>Access Manager API | Policy Manager API |
| Default Policy Domains | NetPoint Identity Domain<br>COREid Identity Domain | Identity Domain |
| Default Policy Domains | NetPoint Access Manager<br>COREid Access Manager | Access Domain |
| Default Authentication Schemes | NetPoint None Authentication<br>COREid None Authentication | Anonymous Authentication |
| Default Authentication Schemes | NetPoint Basic Over LDAP<br>COREid Basic Over LDAP | Oracle Access and Identity Basic Over LDAP |
| Default Authentication Schemes | NetPoint Basic Over LDAP for AD Forest<br>COREid Basic Over LDAP for AD Forest | Oracle Access and Identity for AD Forest |
| Access System Service | AM Service State | Policy Manager API Support Mode |

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

## Tuning the Directory

- To optimize performance, you should ensure that your directory performance is optimal. In this release, information on directory tuning has been enhanced. This release provides a new parameter for clearing the LDAP connection cache. Also, new guidelines for configuring the directory connection pool size has been added.

**See:** "Guidelines for Directory Tuning" on page 2-1, "Check the Performance of the Directory" on page 2-2, "Directory Connection Pool Size" on page 2-2.

## Tuning the Access Server

- Guidelines have been provided on configuring threads and queues.

  **See:** "Changing the Number of Request Queues and Threads" on page 2-18.

## Tuning Workflows

- There are best practices for optimizing workflow performance.

  To minimize the impact that workflows have on server performance, you can tune various parameters in workflowdbparams.xml. You can also tune various workflow search parameters to enhance performance.

  **See:** "Tuning Workflows" on page 2-27.

## Tuning Your Network

- There are best practices for optimizing network and Oracle Access Manager performance.

  **See:** "Tuning Your Network" on page 2-28.

## Failover and Load Balancing

- Information has been added on load balancing of LDAP data.

  **See:** "About Load Balancing of LDAP Data" on page 3-1.

- A "heartbeat" polling mechanism facilitates immediate failover to a secondary directory server when the number of connections in the connection pool falls below the specified threshold level. Information has been added on setting the polling interval for failover.

  **See:** "Setting the Polling Interval" on page 3-11.

## Reconfiguring Oracle Access Manager

- You can change basic components that you specified during Oracle Access Manager installation, such as the person object class or the directory server host.

  **See:** "Reconfiguring the System" on page 5-1.

# 1

# Capacity Planning

Capacity planning consists of selecting the right equipment for an Oracle Access Manager deployment based on anticipated usage. For most deployments, Oracle Access Manager functions well using standard off-the-shelf equipment.

When planning for an Oracle Access Manager installation, you want to be sure that your equipment is adequate for handling peak loads.

This chapter provides information on a medium-sized deployment of 20,000 users and a large deployment of 100,000-2,000,000 users.

This chapter includes the following topics:



For an overview of Oracle Access Manager, see the *Oracle Access Manager Introduction* manual.

## About Capacity Planning

Capacity planning is the process of determining your hardware and memory requirements. The goal of capacity planning is to maintain good system performance during times of peak load. There are no rules when it comes to capacity planning. A quote from an anonymous author on the Web states: The reasonable answer to any performance question begins with "it depends."

Having said that, it is important to use the right equipment for the task. In general, it is cost-effective to have more equipment than you need than it is to try to make do with inadequate hardware. The cost of extra hardware is low compared to the effort required to maintain an under-powered system. Oracle Access Manager components tend to function well on standard hardware such as the machines listed in this section. In fact, if you use the information in this section as the basis for your capacity planning, this may be entirely adequate for your environment.

If you want to take a more methodical approach to equipment planning, a general guideline for capacity planning is to estimate your peak load and to purchase equipment that is capable of handling peak loads. This chapter presents methods of estimating peak usage.

**Note:** You can spend a considerable amount of time calculating your requirements. However, following the recommendations for typical hardware configurations in this chapter may ultimately be the simplest and best choice.

A key point: Remember that it is hard to predict all of the variables in your network. As a result, most calculations of peak usage are error-prone and may ultimately be less reliable than following a few simple guidelines.

# Estimating Peak Load

To ensure that you have adequate equipment for your environment, your machines need to be able to handle the maximum number of operations that can be expected in a particular time interval. The equipment that you use in your Oracle Access Manager installation should be able to accommodate your users during times of peak load.

The information in this section is divided into the following topics:

- General guidelines for estimating peak load

  These guidelines are somewhat reductionist, but they are reasonable in light of the fact that network performance tends to be unpredictable. If you can estimate your peak system throughput, you generally can predict what class of server you need.

- Specific methods

  You can use more labor-intensive methods of predicting requirements by analyzing estimated system usage. These more complex methods, however, may or may not result in better overall predictions of hardware requirements.

## General Guidelines for Estimating Peak Load

A simple method for estimating peak usage is as follows:

- Measure usage over at least a day, and preferably over several weeks.

  If usage tends to spike during particular weeks of the year, try to obtain measurements from the busiest weeks.

- Use the highest value seen in production.

- Estimate the what a typical busy load is.

  To estimate a typical busy load, multiply the value of an average heavy load by a small integer such as 2 or 3. This allows for usage patterns that are two or three standard deviations higher than an average heavy load, assuming a Gaussian distribution (bell curve) of loads.

If you have a multi-site deployment, you may want to create a chart of peak usage for all sites and estimate peak load based on total estimated usage across the sites. For instance, you can record the number of logged-in users at different times of the day. The following table illustrates this method of estimation:

| Peak hours GMT | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Mexico | 0 | 0 | 0 | 3 | 75 | 150 | 175 | 225 | 225 | 225 | 225 | 225 | 225 | 225 | 225 | 175 |
| Spain | 35 | 50 | 75 | 50 | 25 | 35 | 50 | 75 | 75 | 75 | 75 | 35 | 20 | 0 | 0 | 0 |

| Peak hours GMT | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Egypt | 45 | 45 | 50 | 50 | 50 | 50 | 50 | 55 | 55 | 45 | 30 | 10 | 0 | 0 | 0 | 0 |
| U.S. | 0 | 2 | 5 | 10 | 30 | 80 | 95 | 95 | 95 | 95 | 86 | 80 | 80 | 90 | 75 | 60 |
| Columbia | 0 | 2 | 7 | 15 | 30 | 45 | 45 | 50 | 50 | 50 | 55 | 55 | 55 | 55 | 45 | 35 |
| Costa Rica | 0 | 0 | 0 | 0 | 2 | 5 | 10 | 10 | 10 | 12 | 12 | 12 | 12 | 12 | 12 | 12 |
| Indonesia | 60 | 45 | 30 | 10 | 4 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 5 |
| Taiwan | 125 | 115 | 100 | 60 | 30 | 5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 10 | 25 | 75 |
| Total | 265 | 269 | 267 | 198 | 372 | 425 | 425 | 510 | 510 | 502 | 483 | 417 | 392 | 392 | 385 | 362 |

If you can estimate the number of transactions per user during your peak hours, you have an estimate of your overall system capacity. You can compare your estimates of transactions-per-second to your equipment manufacturer's estimates for your hardware. Based on these comparisons, you can determine if the machines you already have are adequate for supporting the estimated load, and if not, you can base your equipment choices in part on your throughput requirements.

**Note:** Even this method of estimation may be more rigorous than is required for a deployment of fewer than 20,000 users. Standard server class hardware is adequate for most deployments.

## Complex Estimates of Peak Load

As an alternative to taking simple measurements of peak usage patterns, you can use an estimation method. This may be practical if there is no simple way to measure usage--for instance, if you have a geographically distributed environment. For a geographically distributed environment, you can identify the hours of peak usage and estimate the number of users who are present during those times. It may be more practical to make these estimates rather than trying to gather statistics on who is logged in at each office.

For instance, suppose your company has offices in a variety of countries. You can create a chart of regular office hours plotted against the GMT time, as illustrated below. The table below allows you to estimate the times (GMT) when the majority of your offices are busy, as indicated by the shaded area in the above table. This is a good indication of peak load hours:

| GMT | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 |
| Mexico | 19 | 20 | 21 | 22 | 23 | 24 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 |
| Spain | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 1 |
| Egypt | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 1 | 2 |
| U.S. | 20 | 21 | 22 | 23 | 24 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 |
| Columbia | 20 | 21 | 22 | 23 | 24 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 |
| Costa Rica | 19 | 20 | 21 | 22 | 23 | 24 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 |
| Indonesia | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| Taiwan | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |

Using Human Resources data on the number of users per office, you can estimate the total number of users accessing your resources during peak hours.

| Country | Full time | Part time | Contract | Total |
|---|---|---|---|---|
| Mexico | 3021 | 496 | 35 | 3552 |
| Spain | 755 | 356 | 5 | 1116 |
| Egypt | 329 | 275 | 0 | 604 |
| U.S. | 134 | 25 | 55 | 214 |
| Columbia | 1290 | 245 | 11 | 1546 |
| Costa Rica | 175 | 130 | 0 | 305 |

Again, you can create a simple estimate of throughput based on an estimate of the maximum number of users that you expect to be logged in at the same time, the estimated number of transactions per user for a given time period, and ultimately the estimated transactions-per-second that need to be supported. You can compare your transactions-per-second estimates with claims made by your hardware vendors.

## Predictions of Concurrency and Throughput

You can calculate peak requirements by estimating either concurrency or throughput. Oracle Access Manager is a stateless system. As a result, a discussion of concurrency is not entirely meaningful and estimates of throughput are more useful. For example, a user can log in but let their account remain idle until they time out. This user is accessing the account concurrently with other users, but their effect on throughput may be minimal.

However, you may want to use estimates of concurrency as a partial method of predicting system requirements. Based on an estimate of concurrency and an estimate of average number of transactions per user per minute, you can estimate system load.

For example, the Response-Time Law states that the number of requests per second ($X_o$) is equal to the number of concurrently logged-in users divided by response time and think time:

$X_o = N/R+Z$

For example, for a community of 5,000 users with an estimated response time of 3 seconds and an estimated user wait or think time of 2 seconds, the estimated requests per second would be 1,000.

Little's Law states that the number of simultaneous connections is equal to the throughput times the response time. That is:

N (concurrency) = Xo (requests per second) * Rsite (response time for the web site)

Based on a network response time of 1 second, the Web site time would be 2 seconds (3-1), and the estimated concurrency would be 1,000 * 2 or 2,000 users. Assuming this is an estimate of concurrency during average load, you would want to multiply this concurrency by 2 or 3, for a peak load estimate of 6,000 users. This can be the basis for estimating your hardware requirements.

**Note:** Little's Law is a simple equation to calculate. However, it may also be relatively simple to base the calculation on an incorrect assumption.

You can also use information about the number of simultaneously logged in users to construct a poisson distribution of the estimated number of concurrent users. A poisson distribution is often used as a model for the number of events, such as the number of telephone calls at a business or the number of accidents at an intersection, in a specific time period.

The following are input parameters for a poisson distribution of concurrent users:

- The total number of users who may be logged in during peak hours.
- The time period for critical usage.
- The estimated duration of a user session.

Information on calculating a poisson distribution is readily available on the Web and in textbooks and will not be covered further in this chapter.

**Note:** Keep in mind that although a complex analysis may help you feel more confident about system requirements estimates, there may be a number of unpredictable elements on your network that interfere with the accuracy of your estimates. As a result, using the simpler guidelines and recommendations in this chapter may, in fact, be the most efficient method for estimating peak usage.

## Capacity Planning for Multiple Environments

A standard Oracle Access Manager deployment consists of several instances of various components installed for different purposes:

- Development and Test

  This is an area where you configure and test a fully deployable system. On a development or test server, a smaller amount of RAM may be required. For development and test systems, you can run the Web server on the same system as the Identity Server, assuming that only a small development team is using the machines rather than the whole user population.

- Staging

  This is an area where new application rollouts, software upgrades, and performance benchmarking can be performed without affecting your production and development environments. This environment can closely resemble the production environment, though it may be a scaled down image of it. For instance, a smaller amount of RAM may be used.

- Production

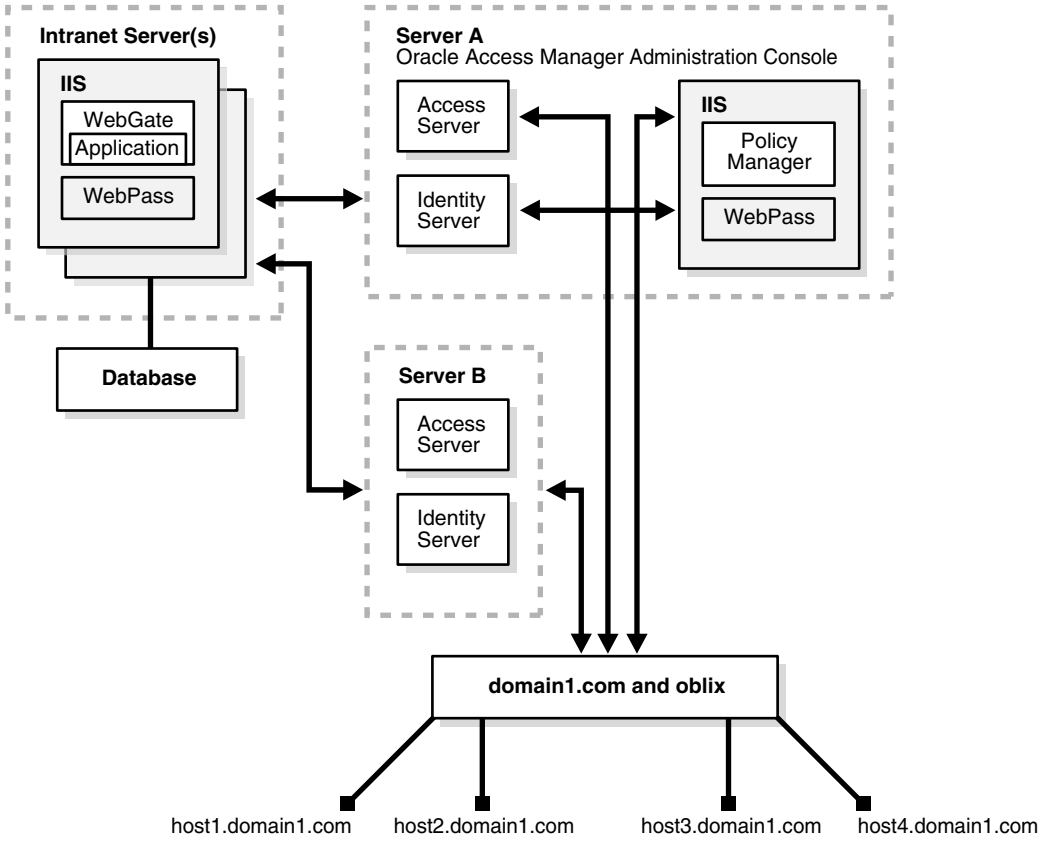
  This is an area where your end users can access the deployed system. Oracle recommends you use a replicated directory and Oracle Access Manager's native load-balancing features. You may also want to configure the Oracle Access Manager Servers for failover and load balancing and have separate machines for your Web servers. As an alternative, you can install multiple Web server instances on a single machine.

## Example from an Actual Deployment

The following diagram Figure 1–1 illustrates hardware and software choices in an actual mid-sized (10,000-20,000 user) deployment. Using a poisson distribution, the

estimated concurrency was predicted to be 74 users, with a 0.01% probability of having 100 or more users.:

***Figure 1–1 Hardware and Software Choices in an Actual Mid-Sized Deployment***




For the production environment:

- The servers are running Windows and Active Directory, IIS, and Oracle Access Manager.
- Each server has two CPUs and 2 GB of RAM, and about 10 GB of disk space.
- There are two Identity Servers running on different machines.
- Multiple WebPass instances are installed in several intranet IIS servers, with at least one server acting as the administration console for the Identity System.
- There are two Access Servers running on different machines.
- There is one WebGate installed on each IIS intranet server.
- The administrative console for the Access System is installed on an IIS server on one of the two Access and Identity Server boxes.

This environment could be scaled as follows:

- The number of Oracle Access Manager Servers can be increased to spread the load and offer higher redundancy.
- The number of CPUs, memory, and disk space can be increased to improve server performance.

- The Active Directory environment supports the addition of new servers for cases where this is desired to improve performance and reliability.

# 2

# Performance Tuning

Once you have installed and configured Oracle Access Manager, you want to be sure to get the best possible performance out of the product.

This chapter provides information for experienced Oracle Access Manager administrators on how to optimize product performance. This chapter includes the following topics:



> **Note:** For details about Policy Manager tuning factors for Apache Web servers, see the *Oracle Access Manager Installation Guide.*

## Guidelines for Directory Tuning

Oracle Access Manager stores its data in an LDAP directory. When diagnosing problems that appear to be due to Oracle Access Manager, you may want to check the performance of the directory. Additionally, many performance issues can be resolved by avoiding trips to the directory, especially for write operations.

The following sections discuss these topics:

- Configuring the Searchbase
- Applying Search Constraints
- Increasing Connections to the Directory in the Identity System
- Changing Directory Content
- Adjusting Cache Settings
- Deleting ObSyncRecord Entries from the Directory

## Check the Performance of the Directory

If directory performance is slow, it will affect the performance of the Access Server or Identity Server. 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.

## Directory Connection Pool Size

The Identity Server and the Access Server open a configurable number of connections to LDAP servers. If you perform a large number of time-consuming searches of user profiles, you can increase the database connection pool size to improve performance.

You configure the connection pool size in the Identity System Console or Access System Console. Note that you specify the details for connections used for accessing configuration data and policy data in configuration files, not through the System console. At startup, a number of connections are opened based on an Initial Connections setting. As needed, more connections are opened until the Maximum Connections setting is reached. Connections remain open until the Identity or Access Server shuts down or the directory server stops responding.

### Differences Between Configured and Actual Connection Pool Size

The number of connections that the Identity or Access Server opens to the directory is different from the number of connections that you specify in the Identity System Console or Access System Console. This is because certain connection details are picked up from configuration files. Connections are specified in the following files:

- **Connections for accessing configuration data:** Three database configuration files reside in *Oracle_Access_Manager_install_dir*/oblix/config/ldap. The setting for the number of connections applies to each of the configuration files. The default value is 1, which means at any time at least three connections are open.
- **Connections for accessing policy information:** Four locator files are used to manage policy definitions. These locators reside in *Identity_Server_install_dir*/oblix/config/ldap. The setting for the number of connections to be opened is on a per-locator-file basis. The default value is 1, which means at any time at least four connections are open.
- **Connections for database profiles that are created during setup:** During product setup, Oracle Access Manager automatically creates one or two database profiles, depending on if the configuration base and search base are the same or disjoint. Each database profile has one instance and the number of connections is set to 1. If the database profile supports authentication operations, a separate connection pool is used for authentication operations. For example, if a database profile has one database instance and the number of connections is set to 1, two connections may be opened—one for authentication and one for other LDAP operations.

To illustrate the difference between the configured and actual number of connections, suppose that you have performed product setup without configuring disjoint domains. In this case, nine connections are opened by default:

- One each, for a total of three, for the configuration data.
- One each, for a total of four, for policy data
- One each, for a total of two, for a database profile with one database instance and one configured connection.

As another example, suppose that you configure two database profiles, each with one database instance and an Initial Connections setting of three, and a Maximum Connections setting of five. All database instances apply to the same directory, and configuration and user data are stored in the same directory. Authentication operations are supported. In this example, the minimum connections opened to the directory is 19, as follows:

- One each, for a total of three, for configuration data.
- One each, for a total of four, for policy data
- Three each, for a total of twelve, applies to the two database profiles.

  Each database profile will have six connections, since there will be two connection pools with three connections each. There will be one pool for authentication requests and another for other operations.

In this scenario, if you set the maximum number of connections to 27, the number of connections to the directory will range from 19 to 27.

### Configuring the Connection Pool

The following procedure explains how to configure the number of LDAP connections.

#### To increase the connection pool size for user data

1. Navigate to Identity System Console, System Configuration, Directory Profiles.
2. Click the name of a user data directory server instance in the Configure LDAP Directory Server Profiles list on the Configure Profiles page.
3. Click the name of a directory server instance in the Database Instances list of the Modify Directory Server Profiles page.
4. On the Modify Database Instance page, modify the two parameters below:
   - **Maximum Connections**: Increases the number of available connections in the pool. The default value is 1. No upper limit is enforced; you can experiment with this setting to find a suitable value.
   - **Initial Connections**: Specifies how many connections are opened at database initialization. The default value is 1. There is no upper limit.

## Storing Workflow Tickets in the Directory

As described in the *Oracle Access Manager Administration Guide*, by default each workflow step generates a ticket and Oracle Access Manager writes the ticket to the directory.

One way to avoid unnecessary directory write operations is to minimize the number of steps in a workflow.

Another way to avoid unnecessary directory write operations is to change Oracle Access Manager default behavior of creating tickets for every step in a workflow.

Tickets are of little value when:

- The information in a step has little value for the next step.
- A participant triggers the workflow but there are no participants for other steps.

#### *Workflow Example*

Suppose you create a workflow consisting of the following steps:

1. **Initiate**: The user initiates a self-registration.
2. **External action**: The request is passed to an external process.
3. **Enable**: The workflow is enabled.

If no one participates in steps 2 and 3, the workflow may not require a ticket.

### Writing Workflow Tickets to the Directory

The WFInstanceNotRequired flag determines whether every workflow step generates a ticket. This flag is set to false by default, which means all workflow instances are written to the directory. When set to true, workflow instances are only written to the directory server when:

- A user action is required.
- Any errors are encountered (for example, the commit action fails).
- Any subflows are triggered.

Setting the WFInstanceNotRequired flag to true reduces the directory size and speeds up the workflow process. The disadvantage of setting this flag to true is that, because the instances are not being stored, you will not able to see all of the workflows that were executed from the Oracle Access Manager Workflow Monitor.

### To avoid create tickets for every workflow step

1. Set the WFInstanceNotRequired flag to true in the following file:

   *IdentityServer_install_dir*/identity/oblix/data/common/workflowdbparams.xml

2. Restart the Identity Server.

## Indexing Attributes in the Directory

Indexing improves search performance in the directory, although at the cost of slower database modification and creation operations and disk space. Oracle Access Manager automatically indexes key attributes when you run the setup program. The index files are specific to your directory server type, and are stored in:

*IdentityServer_install_dir*/identity/oblix/data/common

where *IdentityServer_install_dir* is the directory where the Identity Server is installed.

Index types include the following:

- An equality index improves searches for directory entries that contain a specific attribute value.
- A presence index improves searches for directory entries that contain a specific attribute.

- A substring index improves searches for entries that contain specific text strings.

You can enhance performance if you index attributes that are read during various operations. These include:

- Attributes used in searches that are triggered *indirectly* by user interaction.
- Attributes used in searches that are explicitly invoked by users.
- Attributes used in mapping filters for authentication schemes.
- Attributes in filters for dynamic member definitions in Group Manager.

Your directory documentation describes how to index attributes.

Oracle Access Manager provides an index file fragment showing how Oracle Access Manager-specific attributes can be indexed for each supported directory server. For example, the Oracle Access Manager attributes that can be indexed to improve performance in a directory managed by Sun (formerly iPlanet) directory server are listed in the file:

*IdentityServer_install_dir*/identity/oblix/data/common/iPlanet5_oblix_index_add.ldif

An example of an index entry in this file is:

```
index obclass pres,eq,sub
```

This means that the attribute obclass can be indexed for presence (*pres*), equality (*eq),* or substring (*sub)* .

> **Note:** The iPlanet5_oblix_index_add.ldif file needs to be loaded to the directory server using ldapmodify.

### Limitations of Indexing

You should carefully weigh the benefits of indexing directory attributes for search operations against the performance hit incurred when these attributes are modified. When an attribute value is modified, indexes for that attribute may need to be rebuilt by the directory server. However, this is accomplished varies by directory server.

You should read the manufacturer's guidelines for indexing. Avoid over-indexing attributes.

### Indexing and User Deactivation

After you deactivate a significant number of users, Oracle Access Manager performance can start to degrade. If you experience this problem but want to maintain all deactivated users in the directory, use an equality index for the following attributes:

- ObIndirectManager
- ObUniqueMemberstr

An equality index allows you to search efficiently for entries containing a specific attribute value.

### Indexing and Workflows

If it takes a long time to delete a workflow, index any attributes that the system checks during a delete operation. Performance issues with the delete workflow function usually result from attributes that need to be indexed.

Index the following attributes using the types of equality, presence, and substring:

- ObWorkflowID
- ObContainerID
- ObWFStepID
- ObWFTargetID
- ObIsWorkflowProvisioned
- ObLocationDN
- OblixGID
- Manager
- Secretary

An equality index type improves searches for directory entries that contain a specific attribute value. A presence index type improves searches for entries that contain a specific attribute. A substring index type improves searches for entries that contain specific strings.

For example, if you search for an attribute with a substring index type as follows:

```
cn=*lane
```

The search would match common names containing these strings:

```
John Lane Jane Tulane
```

If you conducted this search:

```
telephonenumber= *123*
```

The search would return all entries with telephone numbers that contain 123.

### Indexing and Groups

The following attributes are used for group expansion and could be indexed to improve performance:

- Any attribute configured with the obSDynamicMember semantic type
- The obGroupExpandedDynamic attribute of the oblixAdvancedGroup object class
- All user attributes used in the dynamic filters of the groups to be expanded.

### Indexing and Search Constraints

You can limit the performance impact of searches by forcing users to use a minimum number of characters for a search. This is controlled through the searchStringMinimumLength parameter. The default value for this parameter is 0.

### To set a minimum number of search characters

1. Locate seachStringMinimumLength in:

   *IdentityServer_install_dir*/identity/oblix/apps/common/bin/oblixappparams.xml

2. Set its value to the minimum number of characters for a search.

   For instance, if you set it to 6, users could not search for "Smith" because it has only 5 characters. The constraint only applies to the longest search string supplied by the user. For example, if the search is on both surname and job title, and the user enters "manager" for job title, the longest string ("manager") has 7 characters, the search is allowed. A value of 0 turns off checking.

The searchStringMinimumLength constraint is enforced for searches that are conducted using the Oracle Access Manager user interface and for clients using other interfaces to Oracle Access Manager (for example, Identity XML clients).

### To enforce a per-attribute minimum number of characters

1. Set the searchStringMinimumLength to 0 in the catalog.
2. In the JavaScript code, find the function validateSearchAndSubmit() in the file misc.js.
3. Modify the JavaScript code to handle the per-attribute checking you require.

> **Note:** This technique does not enforce the constraint on users of Identity XML clients.

As with other parameters, you can set searchStringMinimumLength to one value in the global oblixappparams.xml catalog, and to a different value in one or more of the application-specific catalogs. For example, to override the global value and set searchStringMinimumLength to a different value for User Manager, specify the parameter and its value for User Manager only in:

*IdentityServer_install_dir*/identity/oblix/apps/userservcenter/bin/userservcenterparams.xml

## Changing the Number of Access Server-to-Directory Server Connections

By default, each Access Server opens only one connection to a primary directory server and one connection to a failover directory server (if failover is configured). This may not be optimal for systems with heavy loads.

To configure additional Access Server-to-directory server connections for Oracle Access Manager configuration and policy data, use the command line tool configureAAAServer. The *Oracle Access Manager Access Administration Guide* provides information on this tool. In environments with very high loads (perhaps 1,000 hits/second on the Access Server), creating twenty connections has significantly improved performance.

> **Note:** If your directory is running on a low-powered system, the directory itself may be the limiting factor. In this case, increasing the number of connections may have little benefit. If the machine running the directory is the problem, increase the number of directory server instances and configure load balancing among them.

## Deleting and Archiving Workflows

To prevent the Oblix tree from getting too large, periodically delete or archive workflows. Deleting a workflow instance removes the directory entries associated with that instance. Archiving a workflow instance keeps a record of the instance in an LDIF file and deletes the instance from the directory.

The frequency for archiving or deleting depends on how frequently your workflows are used.

Archived workflows are stored in LDIF format. The default archive file is:

*IdentityServer_install_dir*/identity/oblix/data/common/wfinstance.ldif

Multiple archive operations add information to this file. You can change the archive file by changing entries in the following files:

- User Manager

  *IdentityServer_install_dir*/identity/oblix/apps/userservcenter/bin/usc_wf_params.xml

- Group Manager

  *IdentityServer_install_dir*/identity/oblix/apps/groupservcenter/gsc_wf_params.xml

- Org Manager

  *IdentityServer_install_dir*/identity/oblix/apps/objservcenter/osc_wf_params.xml

The entry to change is similar to the following:

```
<NameValPair ParamName="archiveFileName" Value="wfinstance.ldif"/>
```

### To delete or archive a workflow

1. In User, Group, or Organization Manager, click Requests.
2. Click Monitor Requests.

   For subflows, if the first step has not been processed, the Date Processed field will be empty.
3. In the Search fields, select your search criteria
4. Click Go.

   The results appear below the search fields.
5. Click individual check boxes or the Select All button.
6. Click Next or Previous as necessary to see other results.
7. Click the Delete or the Archive button.

## Setting Read and Write Permissions for Administrators

By default, administrators can view all attributes in the directory so that they can perform initial setup of Oracle Access Manager. Because end users only have read permissions for specific attributes, there is a difference in performance for an administrator and an end user.

The parameter BypassAccessControlForDirAdmin controls whether the Master Identity Administrator or a delegated administrator is permitted to view all attributes in the directory. By default, the value of the parameter BypassAccessControlForDirAdmin is set to true. You can set the BypassAccessControlForDirAdmin parameter to false in the following file:

*IdentityServer_install_dir*/identity/oblix/apps/common/bin/globalparams.xml

If the BypassAccessControlForDirAdmin flag is set to false, performance is the same for non-administrative and administrative users. This is because attribute read and write permissions are applied to the administrative users.

The following is an example of the parameter entry:

```
<SimpleList>
  <NameValPair ParamName="BypassAccessControlForDirAdmin" Value="true">
  </NameValPair>
</SimpleList>
```

## Configuring the Searchbase

Searchbase configuration can affect Oracle Access Manager performance.

Guidelines for searchbase configuration include:

- Set the searchbase for the user object class at a point in the people branch where all users can be found, for example, the root.
- Minimize the number of search bases that you configure per user, group, or organization object class.
- It is more efficient to configure attribute access controls than it is to limit user access by setting multiple search bases.
- Use the default searchbase filter objectclass=* whenever possible.
- Instead of defining searchbase filters, configure read and write permissions for attributes.
- Configure workflow rules and roles to control the user's ability to view and modify attribute values.

### Setting a Searchbase Filter

When configuring a searchbase, if you add a filter and enter a filter other than the default (objectclass=*), the Resource Filter Search scope takes effect. The Resource Filter Search scope has no effect unless you define a filter.

The default searchbase filter (objectclass=*) instructs Oracle Access Manager to apply the user's search criteria to the entire section of the directory tree that has been selected as the searchbase. If you specify other criteria, for example, (telephone=408*), Oracle Access Manager retrieves all users who satisfy that criteria, then applies your search criteria to the subset specified by the filter instead of to the entire tree.

This can degrade performance, especially if the filter retrieves a large number of entries. For example, if you specify (objectclass=wwmOrgPerson) in the filter, Oracle Access Manager may recover all users (assuming all users satisfy this filter) under the specified tree before applying your search criteria. This can seriously degrade performance. You do not have to specify (objectclass=wwmOrgPerson) because the searchbase is already set for that object class. In general, setting read and write privileges for attributes is a better way of controlling user access than setting a searchbase filter. To optimize directory performance, avoid defining complex filters for the searchbase. In the case of searchbases for a tab, only objects of the class which that tab applies to are searched. There is no need to mention (objectclass=*) explicitly.

If you must enter a filter, you can limit the performance impact by setting the resourceSearchFilter scope parameter to 1.

## Applying Search Constraints

The larger the number of entries that a search actually or potentially returns, the longer the search takes. For an interactive application such as Oracle Access Manager, large result sets may be unmanageable for the end user.

Your directory may allow you to limit the time that the directory server spends on a search, the size of the result, or both.

## Increasing Connections to the Directory in the Identity System

The LDAP Database Instance Maximum Connections is the maximum number of connections allowed from the Identity Server to the directory servers. This value

defaults to 1, but you may see a performance improvement by setting this value to more than 1. (With a SQL profile, the default is 5.)

There is no optimal value for the maximum connections. There are variables to consider such as directory configuration and hardware. You may want to consider setting the maximum directory connections no higher than the number of threads set for a Identity Server. For example, if the Identity Server is configured with only 20 threads, there is no benefit in having more than 20 connections because no Identity worker threads can take advantage of the additional connections.

You may want to increase the maximum connections by increments of 5, and monitor your system performance. If performance is worse, decrease the number of connections. The Initial Connections and the Maximum Connections settings work together. When an Identity Server starts, it opens the number of connections to the directory as specified in the Directory Profile Initial Connections. Those connections are then pooled and used by the Identity Server.

---

**Note:** Additional connections can introduce overhead that in turn can hurt performance. For example, if you restart the directory server, the Identity Server has to reconnect the configured number of connections.

---

## Changing Directory Content

This section describes modifications that can be made to the directory content to affect Oracle Access Manager operation. For a discussion of the tools needed to make these changes, see "LDAP Tools" on page 2-12.

### Ordering the Columns in a Search Results List

A search of the Policy Manager for policy domain names or policy names returns a default set of columns in a default order. You can display different columns or change the order of columns. While this does not affect performance in terms of response times, it may improve your satisfaction with the results returned from a search.

### To modify results for a policy or policy domain names search

1. Locate the DN, for example:

   ```
   obname=SDSearchColumnList, obapp=WRSC, o=Oblix, o=Company, c=US2
   ```

2. Under this DN, find the current column list.

   The column list consists of the values for obsearchresultcolumns.

   Possible values for a search of policy names are:

   - `WRORname`: Policy Name
   - `AuthentPolicyName`: Authentication Rule Name
   - `AuthorPolicyName`: Authorization Rule Name
   - `AdminPolicyName`: Auditing Rule Name
   - `URLPrefix`: URL for the domain controlled by the policy

   Possible values for a search of policy domain names are:

   - *SDName*: Policy Domain Name
   - *AuthentPolicyName:* Authentication Rule Name

- *AuthorPolicyName*: Authorization Rule Name
- *AdminPolicyName*: Auditing Rule Name
- *AbsPathPattern*: Path to the controlled domain

For example, the following is an LDIF extract that shows the policy name, authentication rule name, authorization rule name, and URL for the domain controlled by the policy:

```
dn: obname=SDSearchColumnList, obapp=WRSC, o=Oblix, o=Company, c=US
objectclass: top
objectclass: OblixWRSSearchResultColumns
obname: SDSearchColumnList
obSearchResultColumns: WRORName
obSearchResultColumns: AuthentPolicyName
obSearchResultColumns: AuthorPolicyName
obSearchResultColumns: URLPrefix
```

To change the order or content of the displayed search results, modify this information and store it back to the directory.

### Changing the Bind DN

Using Directory Manager as the bind DN bypasses search limits such as look through limit, size limit, and time limit. Large searches can tie up the directory server and increase the amount of processing that is required by the Identity Server.

You can create another user to use as a bind DN.

> **Note:** This procedure illustrates the technique for Sun (formerly Netscape and iPlanet) directories. Consult your directory server's administration manual for instructions on how to create a directory user with the appropriate permissions.

Create your new user, for example orcladmin. Use the LDAP directory to give the user permissions to act as an administrator for Oracle Access Manager. For the searchbase, configuration base, and all branches underneath them, this user needs the following permissions:

- Read
- Write
- Add
- Delete
- Search
- Compare
- Selfwrite

### To change bind DN permissions

1. From the Sun LDAP Server Admin Console, navigate to the directory server instance and open it.
2. Choose the Directory tab, then locate and right-click the branch for which you want to set permissions.

   For example:

for o=Company, c=US, you would find the Company node and right-click to get a menu.

3. Choose Set Access Permissions in the menu.
4. Add a new access permission, or edit the one already in place.

   There should be two lines when you are done. The first is a default deny for everyone, the second is the allow statement.
5. Choose Allow, then search users and groups to find the new administrative user.
6. Set the rights for this user.

If you make this change after installing Oracle Access Manager, change the bind DN for the Identity Server to match the value you created in the directory.

### Adjusting Cache Settings

Caching avoids the latency of unnecessary lookups or calculations. If they keep the results of recent requests close to the consumer, producers can respond quickly by returning cached results rather than recalculating them. The cost of a cache is usually extra RAM or disk space.

A directory that manages large entries will likely hit a maximum cache size limit first, while a directory of small entries might hit the maximum number of entries limit first. If you only care about one of these two criteria, set the other to an unrealistically large number, or match their limits by setting the maximum cache size first, then dividing that number by the size of each entry to get the number you should use for maximum entries in cache.

### Deleting ObSyncRecord Entries from the Directory

Every time a change is made to a user entry in Oracle Access Manager, a directory entry called ObSyncRecord is created. This directory entry enables the user cache for different Oracle Access Manager Servers to stay in sync. If many updates are made to user entries over a short time interval, the number of ObSyncRecord entries that are written can cause a directory performance issue.

You may want to delete ObSyncRecord entries at a regular interval. If you do this, you may want to check each entry before deleting it to ensure that the entry has been in the directory for a time period that is greater than the cache-flush interval.

## LDAP Tools

Directory applications use Lightweight Directory Access Protocol (LDAP) as a standard tool to create, modify, and report data stored in the directory. Tools are available to allow easy manipulation of this data. This section introduces these tools. More detail is available from the manufacturer of your server application.

### Viewing Directory Content in LDIF Files

The structure of a directory, and the data contained within it, is represented by the content of an LDAP Data Interchange Format (LDIF) file. The file can be *output*, the formatted result of a request made to the directory by an LDAP reporting tool, such as LDAPSEARCH. It can also be *input*, data that is intended for insertion to the directory, either as entirely new data, or as an update to existing data, using an updating tool such as LDAPMODIFY.

The following is an example, part of an LDIF file taken from a Oracle Access Manager Demo Directory:

```
dn: cn=John Kramer, ou=Sales, o=Company, c=US
objectclass: top
objectclass: person
objectclass: organizationalPerson
objectclass: inetOrgPerson
objectclass: companyOrgPerson
cn: John Kramer
sn: Kramer
telephonenumber: 415-555-5269
facsimiletelephonenumber: 415-333-1005
title: Account Manager
departmentnumber: 1204
employeetype: Fulltime
employeenumber: 521-321-4560
givenname: John
.
.
Reporting Directory Content with LDAPSEARCH
```

LDAPSEARCH is one possible tool that can be used to report directory content. There are others, which use a different syntax, but the concepts are the same.

LDAPSEARCH can be used in either a command line or interactive mode. The command line approach is preferable, as it allows users to provide the text of the report request by means of a input file. It is easy to verify the content of this file before making the request. You can correct errors by changing a few characters in the file rather than retyping the full request, which would be necessary in the interactive mode.

### LDAPSEARCH Command-Line Format

The command-line format for LDAPSEARCH is:

```
ldapsearch params filter attr_list
```

> **Note:** Each of the three categories shown between <> is optional; if all are omitted, LDAPSEARCH drops into interactive mode, which is not discussed here.

The categories are as follows:

- **<params>**: P*arameters* tell LDAPSEARCH how to operate. One of them, *-f* , is used to specify a filter file. If instead the search filter is provided on the command line, all parameters must be stated before the filter is stated.
- **<filter>**: The *filter* instructs LDAPSEARCH to provide a subset of the data that would otherwise be provided. For example, a filter could require that only names beginning with N be reported. A filter provided on the command line must be enclosed in quotes.
- **<attr_list>:** The *attributelist*, if included on the command line, overrides the default attribute listing. The default list shows all attributes belonging to the directory entry, except operational attributes. If you wish to see only some of these attributes listed, provide their names in the command line, following the filter, and separated by spaces. If you want to see operational attributes, provide their names

in the command line. If you follow the operational attributes with a * you get the default list of attributes as well.

### LDAPSEARCH Command-Line Parameters

Parameters are always provided in the form:

```
-p pdata
```

where *p* is the parameter, preceded by a dash, and pdata is the information (data) required for the parameter, if any. If the data contains one or more spaces, it must be enclosed in double quotes:

```
-p "pdata with spaces"
```

Following is a list of commonly used parameters. See the reference document for your version of LDAPSEARCH, or use the parameter `/?` to see them listed:

- **-A**: Retrieve the attribute names only, not the attribute values.
- **-b**: Searchbase, the starting point for the search. The value specified here must be a distinguished name that is in the directory. Data provided for this parameter MUST be in double quotation marks. For example:

  ```
  -b "cn=Barbara Jensen, ou=Development, o=Oblix.com"
  ```

- **-D**: Distinguished name of the server administrator, or other user authorized to search for entries. This parameter is optional if anonymous access is supported by your server. For example:

  ```
  -D "uid=j.smith, o=Oblix.com"
  ```

- **-f**: Specifies the file containing the search filters to be used in the search. For example:

  ```
  -f filterfile
  ```

- **-h**: Host name or IP address of the machine on which the directory server is installed. This entry is optional; if no host name is provided, LDAPSEARCH uses the local host. For example:*-h myserver.com*
- **-H**: This generates a list of all possible LDAPSEARCH parameters.
- **-p**: Port number that the directory server listens at. For example:

  ```
  -p 1049
  ```

- **-s**: Scope of the search. The data provided for the scope must be one of the following:
  - **base:** Search only the entry specified in the -b option.
  - **one**: Search only the immediate children of the entry specified in the -b parameter; do not search the actual entry specified in the -b parameter.
  - **sub**: Search the entry specified in the -b parameter, and all of its descendants. That is, perform a subtree search starting at the point identified in the -b parameter. This is the default, if the -s parameter is not used.
  - **-S**: Designates the attributes to use as sort criteria, controlling the order in which the results are displayed. You can use multiple -S arguments if you want to sort on multiple criteria. The default behavior is not to sort the returned entries. In the following example, the search results are sorted first by surname and then, within surname, by first name:*-S sn -S firstname*

- **-w**: Password associated with the distinguished name that is specified in the -D option. If you do not specify this parameter, anonymous access is used. For example:*-w password*
- **-x**: Specifies that the search results are sorted on the server rather than on the client. This is useful if you want to sort according to a matching rule, as with an international search. In general, it is faster to sort on the server than on the client.
- **-z:** Specifies the maximum number of entries to return in response to a search request. For example:*-z 1000*

### LDAPSEARCH Examples

To get the surname (sn), common name (cn), and given name for every employee named John in the Sales organization, from the directory server listening at port 392, entirely from the command line, you could provide the following information:

```
ldapsearch -p 392 -b "ou=sales, o=company, c=US" -s sub "givenname=John" sn cn
givenname
```

Results could be something like:

```
dn: cn=John Jackson, ou=Sales, o=Company, c=US
sn: Jackson
cn: John Jackson
givenname: John
dn: cn=John Kramer, ou=Sales, o=Company, c=US
sn: Kramer
cn: John Kramer
givenname: John
```

You can get the same results by using a filter file. For example, a file called namejohn containing the filter:

```
givenname=John
```

can be used, with the following command line:

```
ldapsearch -p 392 -b "ou=sales, o=company, c=US" -s sub -f namejohn sn cn
givenname
```

## Changing Directory Content with LDAPMODIFY

The LDAPMODIFY tool changes or adds directory content. There are other tools, but the concepts are similar. LDAPMODIFY opens a connection to the specified server, using the distinguished name and password you supply, and modifies the entries based on LDIF update statements contained in a specified file. LDAPMODIFY can also be run in interactive mode, a method that is not discussed here.

If schema checking is active when you use LDAPMODIFY, the server performs schema checking for the entire entry before applying it to the directory. If the directory server detects an attribute or object class in the entry that is not known to the schema, then the entire modify operation fails. Also, if a value for a required attribute is not present, the modify operation fails. Failure occurs even if the value for the problem object class or attribute is not being modified.

**Note:** Keep schema checking turned on at all times to avoid accidently adding data to the directory that could be unusable or cause schema violations when schema checking is turned back on. Schema checking is controlled at the directory administration server console and is generally on by default.

### LDAPMODIFY Command-Line Format

The command line format for LDAPMODIFY is:

- `ldapmodify <params>`

**Note:** The params category is optional; if it is omitted, LDAPMODIFY drops into interactive mode, which is not discussed here.

<*params*> These *parameters* tell LDAPMODIFY how to operate. One of them, *-f* , can be used to specify a file describing modifications to be made to the directory.

### LDAPMODIFY Command-Line Parameters

Parameters are always provided in the form:

```
-p pdata
```

where p is the parameter, preceded by a dash and followed by a space, and pdata is the information required for the parameter, if any. If the data contains one or more spaces, it must be completely enclosed in double quotes:

```
-p "pdata with spaces"
```

Following is a partial list of commonly used parameters. Use the parameter `/?` to see all of them.

- **-a**: Allows you to add LDIF entries to the directory without requiring the *changetype:add* LDIF update statement, which is necessary in the interactive mode. This provides a simplified method of adding entries to the directory; in particular, this allows you to directly add a file created by LDAPSEARCH and modified to make changes.
- **-c**: Forces the utility to run in continuous operation mode. Errors are reported, but the utility continues with modifications. Default is to quit after reporting an error.
- **-D**: Distinguished name of the server administrator or other user authorized to change directory entries. This parameter is optional if anonymous access is supported by your server. For example:*-D "uid=j.smith, o=Oblix.com"*
- **-f**: Provides the name of the file containing the LDIF update statements used to define the directory modifications. For example:*-f changestomake.txt*
- **-h**: Hostname or IP address of the machine on which the directory server is installed. This entry is optional; if no hostname is provided, LDAPSEARCH uses the localhost. For example:*-h mozilla*
- **-H**: Lists all possible LDAPMODIFY parameters.
- **-p**: Port number that the directory server uses. For example:*-p 1049*

- **-w**: Password associated with the distinguished name that is specified in the -D option. If you do not specify this parameter, anonymous access is used. For example: *-w password*

### LDAPMODIFY Examples

Suppose you want to change the stored given name of John Kramer, as reported under the discussion of LDAPSEARCH. The data reported back was:

```
dn: cn=John Kramer, ou=Sales, o=Company, c=US
sn: Kramer
cn: John Kramer
givenname: John
```

This output can be used to derive an input file, ToHarvey, whose content might be:

```
dn: cn=John Kramer, ou=Sales, o=Company, c=US
changetype:modify
replace:givenname
givenname: Harvey
```

The command line would then be:

```
ldapmodify - p 392 -f ToHarvey
```

If you were to now search the directory with the command line:

```
ldapsearch -p 392 -b "ou=sales, o=company, c=US" -s sub "givenname=Harvey" sn cn
givenname
```

The response would be:

```
dn: cn=John Kramer, ou=Sales, o=Company, c=US
sn: Kramer
cn: John Kramer
givenname: Harvey
```

# Access Server Performance Tuning

This section describes ways to tune the performance of the Access Server, including:

- Configuring Password Validation by the Access Server.
- Changing the Number of Request Queues and Threads.
- Performance Considerations when Using ObMyGroups.
- Limiting the Number of Authorization Queries from WebGate.

## Configuring Password Validation by the Access Server

By default, when Oracle Access Manager validates a user password as part of the validate_password plug-in, it passes the request to the user directory server. The directory server validates the password and returns the result to Oracle Access Manager. This operation is slow. In an environment with many authentications, it degrades Access Server performance.

You can control whether to use the Access Server or the directory server for password validation on a scheme-by-scheme basis.

### Process overview: When using Access Server password validation

1. The first time a user's password is validated, the Access Server goes to the directory server for validation.

   If the password is valid, the Access Server caches an MD5 hash of the password. The Access Server never caches a clear text password.

2. The next time the user's password needs to be validated, the Access Server creates an MD5 hash of the supplied password and compares it to the hash of the cached password.

   - If the two hashes match, the user's password is considered valid.
   - If the two hashes don't match, the Access Server validates the password against the directory. If the directory validates the password, the Access Server hashes it and replaces the old hash in the cache.

### The ObCredValidationByAS Parameter

To configure an authentication scheme so that the Access Server validates passwords, add the parameter ObCredValidationByAS with a value of true to the validate_password plug-in parameter list. To control the timeout of the cached password on a scheme-by-scheme basis, use the Time To Live parameter. The default value of the Time To Live parameter is 1800 seconds (30 minutes). To control the interval at which the Access Server goes to the directory for password validation, add the parameter obPwdHashTTL with a value equal to the number of seconds required.

The following is an example of the out-of-the-box validate_password plugin:

```
validate_password obCredentialPassword="password", obAnonUser="cn=anonymous,
o=Company, c=US"
```

The following is an example of validate_password configured to use Access Server password validation with the default Time To Live parameter:

```
validate_password obCredentialPassword="password", ,obCredValidationByAS="true',
obAnonUser="cn=anonymous,o=Company, c=US"
```

The following is an example of validate_password set to use the Access Server password validation with Time To Live set to 100 seconds:

```
validate_password obCredentialPassword="password", ,obCredValidationByAS="true" ,
obAnonUser="cn=anonymous,o=Company, c=US", obPwdHashTTL="100"
```

## Changing the Number of Request Queues and Threads

The Access Server uses request queues to create a sequence of incoming requests that are processed by worker threads.

You can tune the performance of the Access Server by increasing the number of request queues from the default of 1. Multiple request queues can reduce contention between service threads and message threads. This can be particularly beneficial on multi-processor computers. See "To change the number of request queues" on page 2-20 for details. If you change the number of request queues, you should also change the number of threads per queue as recommended in "Estimating the Required Number of Threads and Queues" on page 2-19.

### About Threads and Queues

The request queue holds requests from AccessGates until they are processed. By default, there is one request queue.

The Access Server uses three types of threads:

- Message threads

  Message threads accept new requests from AccessGates and append them to the request queue. The number of message threads is equal to the number of connections opened between the AccessGate and the Access Server.

- Service threads

  Service threads remove requests from the queue and process them. Each request queue has a fixed number of service threads. The number of service threads is configured in the Access System Console. See the *Oracle Access Manager Access Administration Guide* for details. By default, there are 60 service threads per request queue. The total number of service threads in an Access Server is equal to the number of request queues times the number of service threads per queue.

- Utility threads

  These perform housekeeping activities. There are usually between 20-40 utility threads, depending on the number of directory profiles configured for the Access Server, the number of file log writers defined, and so on.

### Estimating the Current Number of Threads

To estimate the total number of threads, take the totals for each type of thread, as follows:

- Total message threads

  The netstat command enables you to determine the number of connections opened by AccessGates to the Access Server. Multiply this number by the number of message threads. For example, if there are 50 WebGates, each with 3 connections to an Access Server, there are 3 * 50 or 150 message threads.

- Total service threads

  The number of service threads is configured in the Access System Console. See the *Oracle Access Manager Access Administration Guide* for details.

- Total utility threads

  An average of 30 can be considered safe, using the guidelines in the previous paragraphs.

For example, if there are 50 WebGates, each with 3 connections to AccessGates, a value of 60 configured for the number of service threads in the Access System Console, and an estimated 30 utility threads, there are a total of 150 + 60 + 30 or 240 threads.

### Estimating the Required Number of Threads and Queues

Having many queues with a relatively small number of threads per queue can cause problems if the requests come primarily on one queue. It may be helpful to configure more than the minimum number of threads per queue. You can control the number of worker threads from the command line or from the Access Server configuration page.

If your Access Server handles more than 800 requests per second, you may need to change the number of request queues. A good rule of thumb:

- Set the number of queues equal to the number-of-peak-requests-per-second/800.
- Set the number threads per queue to 60/number-of-request-queues.

These recommendations are based on benchmark and performance tests. For example, if the peak request rate for an Access Server is expected to be 2000 requests per second,

the number of queues should be 2000/800, or approximately 2. The number of threads per queue should be 60/2 = 30.

Avoid having too few or too many threads per queue. Four threads is adequate for benchmarking, but if the directory server is slow in responding you might need more. The optimal number for the total number of threads may be closer to 100 for a modest improvement in performance on a large, fast system. However, performance is not excessively sensitive to the number.

For example, an environment with 16 queues and 16 threads apiece (256 total threads) produced about 9,000 TPS during a lab test with directory access turned off. An environment with 16 queues and 4 threads apiece (64 total threads) produced about 9,400 TPS on the same configuration. For even the largest deployments, 8 queues should be enough, and 2-4 queues may be sufficient.

### To change the number of request queues

1. When starting the Access Server from the command line, type the following:

   ```
   aaa_server -i install_dir -Q n
   ```

   where *n* is the number of request queues. The number of queues must be an integer to a maximum of 1024.

   When using Access Server service on Windows NT or Win2K the -Q option must be specified as a startup parameter to the service. Alternatively, you can modify the following script to include this parameter:

   *AccessServer_install_dir*/access/oblix/apps/common/bin/start_access_server

2. Restart the Access Server for this parameter to take effect.

## Performance Considerations when Using ObMyGroups

There are several situations where the use of obmygroups proves to be a powerful approach for further integrating the Access System with applications to provide role-based information on the given user. For example, a given application could be modeled as the same set of URLs whose access is restricted only to the members of a few groups in LDAP. However to drive the navigation or presentation of the application, the application would need to know which groups the particular user belongs to because specific menu items or functions would be presented based on group membership. In this case, setting an action with ObMyGroups would be a seamless way to supply such information to the application, thus isolating the application from having to query LDAP directly or resolve whether the group is static, nested, or dynamic.

Using ObMyGroups as an authentication or authorization action requires you to consider the performance implication that resolving user group membership could have in the system for the LDAP directory and Access Server, and consequentially the Web server and application being accessed.

### Guidelines for ObMyGroups

Following are some guidelines and considerations that apply to this feature:

When using ObMyGroups, by default, Oracle Access Manager searches for all group objects within the searchbase and then builds a user/group relationship cache in the Access Server, called the Group Query Cache.The searchbase used is the Access System searchbase. No Oracle Access Manager ACLs would apply to the query. For

example, all groups the user belongs to are supplied, regardless of whether or not the user has read rights to those groups' class attribute.

The user/group relationship cache is built by checking group membership in this order:

1. Static Groups Membership
2. Dynamic Group Membership
3. Nested Group Membership

The Group Query Cache expires every 10 minutes. This timeout is not currently configurable. Contrary to user entries in the Access Server cache, there is no flush mechanism for the Group Query Cache other than waiting for the cache timeout or restarting the Access Servers.

---

**Note:** In general the user/group evaluation is an expensive operation from an Access Server perspective. Oracle strongly recommends that these always be configured with a qualifying LDAP filter. For example, enter `obmygroups:ldap:///o=company,c=us??sub?(group_type=role)`.

---

Depending on the number of groups being searched, ObMyGroups processing may take a significant amount of time. It is best to specify obmygroups in an authentication rule rather than an authorization action and, if possible, have the action be a cookie so that the data is available to other applications under the same Web server without incurring an additional toll.

---

**Note:** When ObMyGroups is used in an authorization rule, limit its use to as few resources (URLs) as practical.

---

Even though user/group relationship information is in the Group Query Cache, the entire cache must be evaluated for each resource protected by a policy that contains an authorization action. The entire cache must be evaluated because all groups must be checked to see if the user is in that group. So even though it's a cache lookup, it's an expensive lookup. You need to consider the Access Server performance impact of using ObMyGroups for authorization actions.

---

**Note:** Oracle strongly recommends that a thorough performance test, with the specific use cases relevant to ObMyGroups actions, be designed and executed prior to rollout. This allows you to benchmark, tune, and understand the actual performance and response times involved in your specific environment.

---

## Limiting the Number of Authorization Queries from WebGate

When you define access policies for a policy domain, the WebGate by default will query the Access Server every time a user attempts to access resources in that domain. The more broad the policy domain, the more often the Access Server is queried. For example, if you configure root (/) as the policy domain in the Policy Manager, the WebGate will contact the Access Server every time someone tries to access a resource on the entire Web site.

To minimize the number of times the WebGate queries the Access Server, you can configure the DenyOnNotProtected flag. 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. See *Oracle Access Manager Access Administration Guide* for details.

# Tuning the Caches

You can tune three groups of Access System caches:

- The cache for policy domains and policies
- The cache for user data
- The cache for URL Prefix

For additional information, see Chapter 4, "Caching and Cloning" on page 4-1.

## Tuning the Policy Cache

A policy cache contains information about policy domains, excluding URLs, policy descriptions, and display names. A policy cache element contains the following:

- Rules
- Actions associated with rules
- Filters, groups, and roles associated with rules
- Policy conditions for rules
- All policy domains
- All policies
- All authentication schemes

You tune the policy cache by setting the Maximum Elements in Policy Cache and Policy Cache Timeout parameters.

---

**Note:** These parameters are also documented in the *Oracle Access Manager Access Administration Guide.*

---

### Calculating Maximum Elements in a Policy Cache

To estimate the requirements, calculate the total number of authorization rules in all policy domains and policies. To do this, search the Oblix directory tree using the following filter:

```
"(objectclass=oblixpolicyrule)"
```

When determining the number of elements in a policy cache, be sure to account for future growth of the number of rules.

### Calculating Memory Requirements for the Policy Cache Elements

To calculate the memory requirements for the Access Server's policy cache elements, check the memory requirements of the directory used to store the policies.

This value is roughly the value you should provide on the Access Server.

### Calculating Policy Cache Timeout

When a policy domain, rule, or policy is created or modified, the administrator has the ability on each screen to Update Cache. When these screens are saved, this forces an immediate flush of the relevant policy cache, ensuring that the change takes effect immediately.

You can have the caches time out on a regular basis as a security precaution or to clean up if the administrator does not press the Update Cache button during a policy change. The *Oracle Access Manager Access Administration Guide* discusses automatic flushing of the Access System cache.

## User Cache Tuning

The tuning parameters available for this are Maximum Elements in User Cache and User Cache Timeout.

### Calculating the User Cache Timeout

The user cache contains all user attributes and values used in audit and HTTP actions. You need to determine the shortest time acceptable for not changing bulk user attribute values. If the value is too large, values that are considered important from a risk-management perspective may not make their way into Oracle Access Manager until the cache is flushed. This can be a problem if the administrator forgets to update the cache after making changes. The administrator, delegated identity administrator, or user may change a user value in the directory and think it has been implemented when, in fact, there will be a delay until the value in the cache is flushed.

On the other hand, you should avoid setting the timeout or the cache size so small that data is flushed while a user is accessing a site, since this forces another trip to the directory.

To estimate a reasonable interval for updating user values, you can track average session times over a 24-hour period. The User Cache Timeout value can probably be set to this value or slightly higher than this value.

### Calculating Maximum Elements in the User Cache

If you know the value for User Cache Timeout, you should next measure the number of users who access resources during this time interval.

When you know the maximum number of concurrent users during the time interval, the number represents the maximum number of cache elements. This is because each element contains a user DN and their attributes and values used in the audit logs and HTTP actions.

### Calculating Memory Requirements for User Caches

If you know the user cache timeout and the maximum number of elements in the user cache, you can calculate hardware requirements.

To begin, calculate the requirements per cache element. An element in the user cache refers to all user attributes and their values used in all the rules. The formula for this is:

```
cache element size = size of DN + size of all attribute values to be cached + 50
bytes for overhead
```

You can now calculate total memory requirements as follows:

```
Max. elements in user cache memory requirements = Max. elements x cache element
```

```
size
```

## Tuning the URL Prefix Cache

The URL prefix cache sets the interval for flushing a URL prefix. If a URL prefix is added to a policy domain or policy, the administrator can click the Update Cache button to force an automatic cache flush. The URL Prefix reload period is an additional safeguard that provides automatic flushing.

Choose a time interval that you feel comfortable with between automatic cache flushes. The default value is 7200 seconds, or two hours.

## WebGate Cache Tuning

WebGate caches information on authentication and on whether or not a resource is protected. WebGate cache tuning refers to the total number of unique URLs expected over the timeout interval.

The chances of an authentication scheme changing quickly are very low. The chances of a URL prefix being changed or added are low. If an administrator adds, changes, or deletes a URL prefix, the screens contain an Update Cache button for forcing an immediate cache flush. As a result, the default value for the WebGate cache timeout is zero. This means that the cache is not automatically updated and is flushed only when the administrator clicks the Update Cache button. If you are not comfortable with the default, choose an appropriate interval before the URL is automatically flushed from the cache.

## Sizing the Maximum Elements in Cache

WebGate can cache URLs to avoid trips to the Access Server or directory server when determining if a URL requires protection. The default value for Maximum Elements is 100,000. To estimate an appropriate value for this parameter, examine the number of URLs that the Web server is protecting and the HTTP operations associated with them.

Web browsers place an internal limit of 4096 bytes on URLs. Most URLs are smaller than 4096 bytes. You may want to determine upper limits for the cache and choose the same size or a smaller size than 4096, depending on what you believe is an average URL size.

To calculate memory requirements per cache element:

```
memory per cache element = URL (4096 bytes) + overhead (128 bytes) = 4224 bytes
```

To calculate the memory requirements for maximum cache elements:

```
memory required = 100,000 elements x 4224 bytes/element = 422,400,000 bytes = 422
MB of memory
```

Memory requirements may be smaller if the maximum number of elements is downsized according to the needs of the Web server that the WebGate is on and the average size of the URLs.

# Tuning the Identity System

It is a good practice to use at least two Identity Servers running in a pooled primary configuration. Pooled primary means using multiple Identity Servers that run as primary servers, with one or more WebPass instances connecting to the primary Identity Servers.

You can use separate Identity Servers as secondary servers when using a pooled primary approach. If you have only two servers, a pooled primary configuration is recommended over using one primary and one secondary server. When running a pooled primary configuration, it is best to use identical but separate hardware for the Identity Servers.

#### Advantages of pooled primary mode

- Increased performance through load balancing
- Increased availability through multiple servers
- Automatic failover

#### Disadvantages of pooled primary mode

- The cost of additional hardware.

  If there are no secondary servers, each primary server needs to be sized to handle the total expected load if the other primary servers are unavailable.
- Additional system configuration.

> **Note:** Identity Server configuration and stylesheet files must be identical on all servers. This applies to all configurations that use multiple Identity Servers. You should configure all Identity Servers from a file system level, that is, ensure that all directory and file system structures are identical.

## Tuning the Group Manager

The Group Manager is a Oracle Access Manager application. The Group Manager has unique performance tuning requirements. In particular, group size can adversely affect performance. For instance, if you have groups of more than 100,000 users, operations involving these groups may be slow. Similarly, nested groups can result in slow performance during evaluation due to the large number of members in the nested groups. Where possible, use dynamic groups instead of nested groups.

In addition, three Group Manager pages can be tuned to optimize performance:

- My Groups page
- Group Expansion page
- View Members page

### Tuning the My Groups Page

You can tune the performance of the My Groups page as follows.

#### To tune the My Groups page

1. In the Identity System Console, navigate to Group Manager Configuration, Group Manager Options.

   On this screen are several Boolean flags that you can set by clicking Modify. The *Oracle Access Manager Administration Guide* describes these settings. As an example of how these flags might influence the performance of your system, consider the setting labeled Show nested groups. Showing nested groups can be a time-consuming operation, depending on the complexity of the group hierarchy.

Setting this option to false limits group display to a simpler format, but can significantly improve the performance of the page.

The types of attributes in step 2 are used in the My Groups profile.

2. To improve directory performance, index the following:
   - Attributes configured with the ObSDynamicMember semantic type
   - Attributes configured as the ObSStaticMember semantic type
   - All user attributes used in group dynamic filters
3. Group Manager can use an optional filter--the extra group filter--to further qualify results shown in the My Groups profile.

   The filter can be used to further qualify results under any searchbase, but would most likely be used in a FAT tree scenario where all entries are located under one container. The parameters that control the use of this filter are extra_group_filter and use_extra_group _filter_mygroups, found in the groupdbparams.xml catalog. The extra filter can be any LDAP filter and, in addition, may contain an Oblix rule substitution ($ $). When using a rule substitution, the substituted value comes from the user entry. This provides a means to link the values of attributes in a user entry with those in group entries. For example, a filter such as ou=$ou$ would specify, that in order for a group entry to qualify, the *ou* of the group must be the same as that for the user.
4. The default stylesheet for the My Groups page uses DHTML and layers to render the groups in a browseable tree format. If the My Groups page has to show many groups, the browser may take a long time to render the page. You can replace the default stylesheet, gsc_myprofile.xsl, with gsc_myprofile_simple.xsl. This version of the stylesheet has a simpler, non-browseable interface and does not use DHTML and layers as much. Both stylesheets are located in the directories:

   *IdentityServer_install_dir*/identity/oblix/lang/en-us/style0/ (stylesheet template)
   *IdentityServer_install_dir*/identity/oblix/shared/ (wrapper stylesheet)

For details about stylesheets and styles, see the *Oracle Access Manager Customization Guide.*

### To use gsc_myprofile_simple.xsl

1. Change the XML registry settings in the registry file:

   *IdentityServer_install_dir*/identity/oblix/apps/groupservcenter/bin/groupservcenterreg.xml
2. In this file, change the following line:

   ```
   <ObStyleSheet name="gsc_myprofile.xsl"/>
   ```

   to

   ```
   <ObStyleSheet name="gsc_myprofile_simple.xsl"/>
   ```
3. Then restart the Identity Server and Web server.

## Tuning the View Members Page

You can tune the performance of the View Members page in the following ways:

- You can control the behavior of the View Members page using three Identity System Console options.

These options can be turned on and off in the System Configuration, Group Manager Configuration, Group Manager Options page. See the *Oracle Access Manager Administration Guide* for information on these flags.

- You can constrain the search that can be done in the View Members page. Locate the groupMemberSearchStringMinimumLength parameter in the catalog:

  *IdentityServer_install_dir*/identity/oblix/apps/groupservcenter/bin/groupservcenterparams.xml

  This parameter controls the minimum number of characters that the user must type to be able to search for members of a group. Other than being restricted to group membership searches, its usage is identical to that of the searchStringMinimumLength parameter (see "Indexing and Search Constraints" on page 2-6 for details).

- You can index any user attributes used in group dynamic filters.

### Tuning the Group Expansion Page

The following attributes are used in group expansion and should be indexed to improve performance:

- Any attributes configured with the *ObSDynamicMember* semantic type.
- The obgroupexpandeddynamic attribute of the oblixadvancedgroup objectclass.
- All user attributes used in the dynamic filters of the groups to be expanded.
- As discussed under see "Tuning the My Groups Page" on page 2-25 you should use the Set Searchbase feature to localize access to sub-domains appropriately. This may also improve the performance of the Group Expansion page.

The performance of the Group Expansion page may be improved by using the extra group filter feature, discussed under see "Tuning the My Groups Page" on page 2-25.

## Tuning Workflows

Workflow performance can be tuned in several ways:

- Tuning workflowdbparams.xml
- Configuring Workflow Search Parameters.

Workflows are also mentioned in the sections on directory tuning and indexing. See "Storing Workflow Tickets in the Directory" on page 2-3 and "Indexing and Workflows" on page 2-5 for details.

### Tuning workflowdbparams.xml

The workflowdbparams.xml file resides in the following location:

`Identity_install_dir/identity/oblix/data/common`

The following parameters can be tuned to assist with workflow performance. You must restart the Identity Server for changes to these parameters to take effect:

- **WfDefCacheMaxNoOfElements**—This parameter sets the size of the workflow definition cache. Set the value of this parameter to be higher than the number of defined workflows.
- **WfDefMaxNumStepDefFiltersPerSearch**—This parameter controls how many searches the Identity System performs on instance data. If users are participants in

a large number of workflow steps, increasing the value of this parameter can reduce the number of times the directory is searched. To experiment with this parameter, increase its value in increments of 20. A higher number reduces searches to the directory, but increases the LDAP filter length. You can monitor the directory CPU utilization to determine an optimum value.

### Configuring Workflow Search Parameters

The following guidelines will help you tune workflow search performance:

- Check the number of search results that are returned per page. The default is 20. A higher number of results per page may cause slow performance.
- If workflow participants are specified as a filter, performance will be slower. See if participants can be specified using a static group.

## Tuning Your Network

The performance of the overall network, or network latency, is a major factor in the performance of the system. A reduction in network latency will be reflected in the performance of Oracle Access Manager.

It is not necessary, and not even desirable, for you to deploy load balancers between Oracle Access Manager components, even if you use load balancers for other application deployments. You should avoid placing load balancers between an Access or Identity Server and any of the following components:

- WebGates or AccessGates
- WebPass
- Directory servers

Oracle Access Manager uses the LDAP protocol to perform update operations, and it provides keep alive, failover, and failback functionality to handle LDAP and network outages, replication, and related activities. The built-in features of Oracle Access Manager are often the same or better than similar features provided by a load balancer.

In some cases, you can negatively affect the performance of Oracle Access Manager by placing a load balancer between its components. For example, a load balancer may terminate a connection that it interprets as idle without triggering a response that Oracle Access Manager can detect and adjust to. This can cause outages.

The following are guidelines for tuning your network:

- You can consider adding an SSL accelerator or load balancer outside of the Oracle Access Manager system to improve the performance of your network.

  Deploying a load balancer in front of the Web servers or application servers is a best practice for increasing availability and performance of Web-based applications, including Oracle Access Manager. However, load balancers are not recommended between the Oracle Access Manager components themselves.

- You may want to set the DenyOnNotProtected flag to True in the WebGate configuration page (or in the WebGateStatic.lst file for pre-10.1.4 versions of Oracle Access Manager).

  This reduces trips between the Access Server and the directory. The DenyOnNotProtected flag forces the WebGate check its own cache to determine if a URL is protected.

- Reduce the latency between devices in high-traffic parts of the network.

You can place the Identity and Access servers closer to client applications than to the directory. During normal operations there is more traffic between WebGates and Access Servers than between Access Servers and the directory. Similarly, there is more traffic between WebPasses and Identity Servers than between Identity Servers and the directory. These servers have the greatest interaction with the directory at startup. After that time, much of the configuration information that these servers need can be read from their caches. One exception is if the administrators are performing system configuration, for example, if they are defining and testing workflows. In this case, placing the Access and Identity servers in the hosting site can help the response time for the administrators.

### Be Sure Your Machines Are Working Properly

If you experience performance issues, you may want to check system CPU usage. For example, if a domain controller is not working well, performance will suffer.

## Resource-Intensive Operations

Resource-intensive operations include login forms, password management, and plug-ins.

The following are issues you should examine to optimize performance.

### Login Forms

Login forms increase the overall load on Web servers. This may mean that you need more Web servers, depending on the size of the server and the form content. Dynamic contents and graphics adversely affect performance. An extremely high number of logins can cause network latency.

### Password Management

Password management causes many directory write operations. When you implement password management, be aware that there are performance implications if you expect it to be used frequently.

### Plug-Ins

Oracle Access Manager plug-ins and .exe files can affect performance. When you develop customizations for Oracle Access Manager, be aware of the impact of these customizations. To minimize performance impact:

- When creating an action for the Identity Event API, because an action incurs overhead, you should identify all the possible events to attach the action to and choose the least frequently used one that yields the desired result.
- Evaluate the sequence in which actions are executed.

  For example, suppose that you develop a .exe file to send an email message through the SMTP server when a user performs a particular action. Depending on how you write this program, the Identity Server may be unable to perform further actions until it receives a reply from the SMTP server. If it is the case that the SMTP server is down, there may be a large impact on performance.
- When comparing EXEs and LIBs, note that LIB actions execute quickly because they are binary code modules compiled from C or C++ source code. However, their perceived speed depends on the function they perform.

# 3

# Failover and Load Balancing

Failover and load-balancing are vital considerations when you performance-tune your Oracle Access Manager environment. This chapter contains the following topics:



## About Load Balancing with Oracle Access Manager

Oracle Access Manager uses load balancing to maximize performance by distributing server requests across multiple physical servers. You configure load balancing among the components listed below.

Load balancing between Oracle Access Manager Web components and servers includes:

- WebPass requests to one or more Identity Servers
- WebGate requests to one or more Access Servers

  You cannot use hardware load balancers to load balance requests from WebPass and/or WebGates to the Identity and/or Access Servers because the hardware load balancers do not understand the proprietary Oracle Access Manager protocols. Hardware load balancers can load balance only at the connection level, not the request level. The connections to the Oracle Access Manager Servers are persistent connections.

Load balancing among Oracle Access Manager Servers and directory servers includes:

- Identity Server to one or more directory servers
- Access Server to one or more directory servers

### About Load Balancing of LDAP Data

Oracle Access Manager supports multi-master LDAP environments. Failover and load balancing are supported for user and policy data, not configuration data.

Before you configure load balancing for LDAP writes of user and group data, note that LDAP replication is likely to produce undesirable behavior. An update to one instance may take some time to be reflected on the another instance. This limitation is inherent to all LDAP directory services since LDAP replication does not guarantee transaction integrity.

For user and group data, clustering or segmenting may be useful for distributing the load for read and write operations. This is particularly the case if separate user populations exist in separate branches of the DIT, for example, if one server stores partner data under the ou=partners branch, and another server stores supplier data under the ou=suppliers branch. In cases like this, you can maintain each set of data on a different primary LDAP server for read and write operations. Also, for availability purposes, each server cluster can fail over to the other LDAP servers in a cross-over fashion. This load-balances read and write operations across LDAP multi-masters.

Some LDAP servers, for example Oracle Internet Directory, provide true load balancing capability for reads and writes. These servers guarantee immediate availability when an update occurs on any of the multi-mastered LDAP servers that are configured in this fashion. When using these types of servers, you can configure Oracle Access Manager to provide load-balancing for read and write operations for user and group data.

Oracle Access Manager does not support round robin write operations on either the user and group directories, nor the configuration branch containing Oracle Access Manager meta-data. Data corruption could result when WebGates and AccessGates are found in a round robin configuration with various Access Servers even if each Access Server points to a single, but different, instance of the configuration data hosted on separate, but multi-mastered directories. This is particularly important when Policy Management is turned on and the automatic cache flush is enabled on the Identity Server (or both). Every cache update triggers at least one write operation in the Policy Manager configuration data.

## Configuring Load Balancing for Web Components

Oracle Access Manager supports both simple round-robin and weighted round-robin load balancing of requests from its Web components (WebPass and WebGate) to their associated servers (Identity and Access). You configure load balancing of Web component requests to Oracle Access Manager Servers from the perspective of the web component using two key fields:

- **Maximum Connections**: The maximum connections you want opened for a given instance of the Web component. This is the total number of live connections you want established to one or more primary Oracle Access Manager Servers at any given time. If the Web component cannot establish a connection to one of its primary servers, it will try to establish a connection to a secondary server.
- **Initial Connections**: The initial connections from the Web component to its associated Oracle Access Manager Servers. This applies to both primary and secondary Oracle Access Manager Servers.

There are several procedures in this section, to use depending upon your environment:

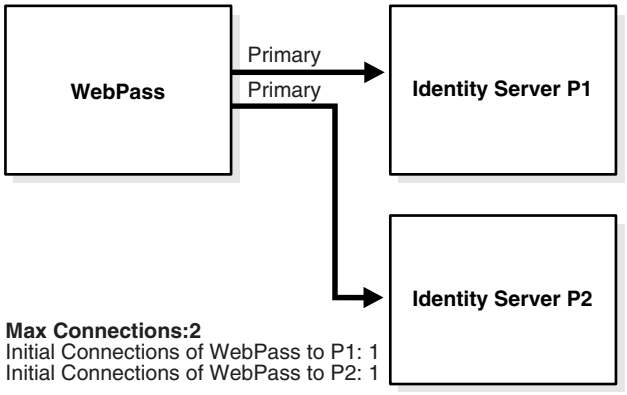
- Configuring Simple Round-Robin Load Balancing
- Configuring Weighted Round-Robin Load Balancing

## Configuring Simple Round-Robin Load Balancing

Configuring simple round-robin load balancing of Web component requests means that a Web component opens a single connection to each of its associated primary Oracle Access Manager Servers in the order they are listed, and distributes the requests evenly among them.

For example, assume that you have a single WebPass and two primary Identity Servers as shown in Figure 3–1. In this case, WebPass opens a connection to each Identity Server in the order they are listed. WebPass sends request1 to Identity Server P1, request2 to Identity Server P2, request 3 to Identity Server P1 and so on.

***Figure 3–1 Simple Round-Robin Load Balancing of Web Component Requests***




### To configure simple round-robin load balancing

1. Access the Web component configuration whose requests you want to load balance.

   For example:

   - From the Identity System Console, select System Configuration, WebPass.
   - From the Access System Console, select Access System Configuration, AccessGate Configuration.

   See the *Oracle Access Manager Administration Guide* and *Oracle Access Manager Access Administration Guide* for more information about Web component configuration.

2. Enter the Maximum Connections you want opened for this WebPass or WebGate.

   This is the total number of live connections you want established to one or more primary Identity or Access Servers at any given time. If the Web component cannot establish a connection to one of its primary servers, it tries to establish a connection to a secondary server.

3. Leave the Initial Connections to each associated Oracle Access Manager Server at the default, which is 1.

   This applies to both primary and secondary Oracle Access Manager Servers.

## Configuring Weighted Round-Robin Load Balancing

You may want to configure weighted round-robin load balancing of Web component requests if your Oracle Access Manager Servers have varying performance capacities.

The primary difference when configuring weighted load balancing is that you adjust the initial connections you want established to each server.

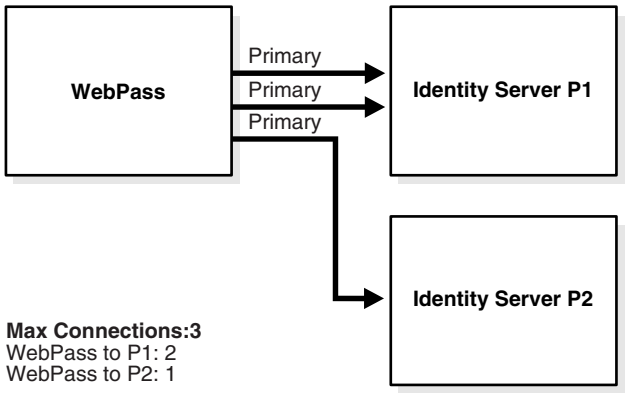
Figure 3–2 provides an example. Assume you have two primary Identity Servers as shown in Figure 3–2. However, Identity Server P1 can handle additional load. Then you may want to configure WebPass to open two connections to Identity Server P1, and one connection to Identity Server P2. The Maximum Connections for that WebPass would be 3: Identity Server P1, request2 to Identity Server P2, request 3 to Identity Server P1 and so on.

---

**Note:** When you associate an AccessGate with an Access Server cluster, Oracle Access Manager automatically configures the number of connections between the AccessGate and all the Access Servers in the cluster based on the maximum number of connections that is specified for the cluster. Load balancing is dynamically configured and Oracle Access Manager ensures that the AccessGate routes requests to the most lightly loaded Access Servers in the cluster.

---

***Figure 3–2 Weighted Load Balancing Layout Using Two Servers and no Failover***




### To configure weighted round-robin load balancing of Web component requests

1. Access the Web component where you will configure load balancing.

   For example:

   - From the Identity System Console, select System Admin, System Configuration, Configure WebPass.
   - From the Access System Console, select Access System Configuration, AccessGate Configuration.

2. Enter the Maximum Connections you want opened for a given WebPass.

   Again, this is the total number of live connections you want established to one or more primary Identity Servers at any given time. If WebPass cannot establish a connection to one of its primary servers, it tries to establish a connection to another primary server.

3. Enter the Initial Connections to each associated Oracle Access Manager server to reflect that server's capacity.

   Again, this applies to both primary and secondary servers. When you associate an AccessGate with an Access Server cluster, Oracle Access Manager automatically

configures the number of connections between the AccessGate and all the Access Servers in the cluster based on the maximum number of connections specified for the cluster. Load balancing is dynamically configured and Oracle Access Manager ensures that the AccessGate routes requests to the most lightly loaded Access Servers in the cluster.

## Configuring Load Balancing among Oracle Access Manager and Directory Servers

Simple round-robin load balancing of Oracle Access Manager server requests to two or more directory servers is supported.

**Note:** Oracle Access Manager generally does not support load balancing for configuration data because many functions are dependent on data carried over from the previous request. In a load balanced environment, this data may not yet be available to the replicated server

The instructions for configuring load balancing for directory servers vary depending on the type of component (Identity Server, Access Server, Policy Manager), and whether you are configuring load balancing for user data or configuration data. See Table 3–1.

Previous versions of Oracle Access Manager managed directory connection information solely through XML configuration files. Recently, Oracle Access Manager provided the ability to manage this information through the interface using the Directory Profile screen in the Identity System Console and the Access System Console. However, some configuration and policy data is still managed through the XML files. Therefore, you will find yourself using combined methods for configuring load balancing.

**Note:** The Identity Server depends on a profile configured for the policy tree to do referential integrity for the policy directory. During Policy Manager setup, this profile is created for the policy directory for the Identity Server component. Whenever a user makes DN changes from the Identity System, the Identity Server uses this profile to update any DN references in the policy directory.

***Table 3–1 Configuring load balancing for directory servers***

| Component | Data Store | Operation |
|---|---|---|
| Identity Server | Users | READ—Configured in the Directory Profile. See "Configuring Load Balancing for User Data" on page 3-6. |
| | | WRITE—Not supported |
| | Configuration | READ—Not Supported |
| | | WRITE—Not Supported |

*Table 3–1 (Cont.) Configuring load balancing for directory servers*

| Component | Data Store | Operation |
|---|---|---|
| Access Server | User | READ—Configured in the Directory Profile.<br>WRITE—Not supported Note: Write operations apply to the Access Server only if password policy is enabled. |
| | Configuration | READ—Configured via the ConfigureAAAServer command line tool. See "Configuring Load Balancing of Configuration & Policy Data" on page 3-7.<br>WRITE—n.a. |
| | Policy | READ—Configured via the ConfigureAAAServer command line tool.<br>WRITE—n.a.<br>Note: If you turn on the Access Management Service for that Access Server, you cannot load balance requests from an Access Server to the data store containing policy information. |
| Policy Manager | User | READ—Configured in the Directory Profile |

This section includes the following procedures:

- Configuring Load Balancing for User Data
- Configuring Load Balancing of Configuration & Policy Data
- Adjusting Connection Pooling for a Directory Server Instance

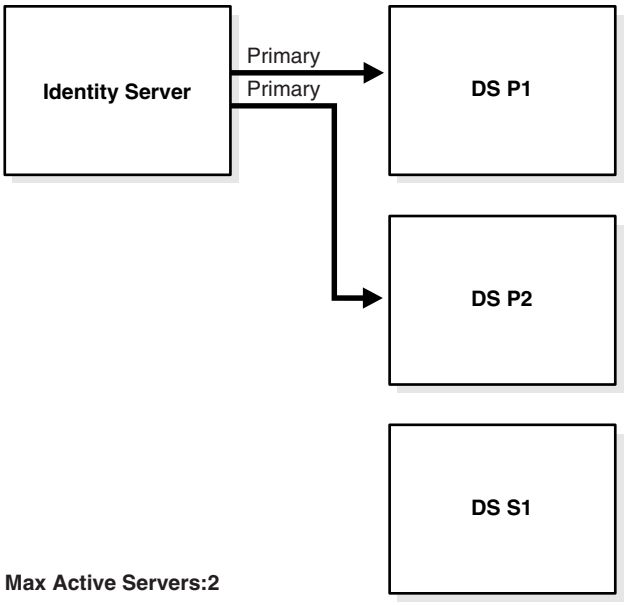
## Configuring Load Balancing for User Data

By default, Oracle Access Manager creates a directory profile for each installed component. When configuring load balancing for Oracle Access Manager requests to directory servers containing user data, you use the Directory Profile page. For more information on directory profiles, see the *Oracle Access Manager Administration Guide.*

**Maximum Active Servers**: The total number of live primary directory servers you want up and running at all times. Requests are distributed evenly across these servers.

As shown in Figure 3–3 assume you have 2 primary directory servers and one secondary directory server. You want to load balance between the primary directory servers, so you should enter Max Active Servers = 2. The secondary directory server only comes into play during failover and does not impact this setting.

*Figure 3–3 Load Balancing of Oracle Access Manager Requests to Directory Servers*




### To configure load balancing for user data

1. Access the Directory Profile page.

   For example:

   - From the Identity System Console, select System Configuration, Directory Profiles.
   - From the Access System Console, select System Configuration, View Server Settings, Directory Options.

2. Under Configure LDAP Directory Server Profiles, select the name of the profile for the component and data where you want load balancing.
3. Enter the Maximum Active Servers available for load balancing.

## Configuring Load Balancing of Configuration & Policy Data

When you configure load balancing of Access Server requests to directory servers containing configuration and/or policy data, use the configureAAAserver tool. The following instructions assume that you have two or more primary directory servers set up.

> **Note:** Load balancing for configuration data is generally not supported for the Identity Server. These instructions address the Access Server only. The ConfigureAAAServer is an older tool that does not have the most current naming conventions. Maximum Connections as shown in the tool is equivalent to the Maximum Active Servers as explained earlier in see "Configuring Load Balancing among Oracle Access Manager and Directory Servers" on page 3-5

### To configure load balancing of configuration and policy data for the Access Server

1. From the command prompt, access the configureAAAServer tool located in *AccessServer_install_dir*/access/oblix/tools/configureAAAServer
2. Run configureAAAServer utility using the reconfig *install_dir* option.

   where *install_dir* is the name of the directory where your Access Server is installed.

   For example:

   ```
   configureAAAServer reconfig "c:\Program Files\COREid1014\access"
   ```

3. Enter the number that corresponds to the Access Server security mode. These are the Access Servers that will connect to the directory servers.
   - 1) Open
   - 2) Simple
   - 3) Cert

   You are then be asked if you want to specify failover information for Configuration or Policy.
4. Select Yes (Y).
5. Specify whether the data is stored in:
   - 1) Oblix tree
   - 2) Policy tree
6. Enter 1 to add a failover server at the following prompt.
   - 1) Add a failover server
   - 2) Modify a failover server
   - 3) Delete a failover server
   - 4) Modify common parameters
7. Enter the following information:
   - Directory server name
   - Directory server port

     For LDAP in an Active Directory forest environment, use port 3268 for Open mode and port 3269 for SSL mode. These two are the global catalog ports.
   - Directory server login DN
   - Directory server password
   - Directory Server security mode
     - 1) Open
     - 2) SSL
8. Enter 1 as the priority since you are configuring load balancing
   - 1) Primary
   - 2) Secondary
9. Enter 4 to modify common parameters.
10. Enter 1 to specify the Maximum Active Servers.

Maximum Connections as shown in the tool is equivalent to the Maximum Active Servers as explained earlier in see "Configuring Load Balancing among Oracle Access Manager and Directory Servers" on page 3-5.

11. Enter the total number of primary directory servers available for load balancing.
12. Enter 5 to Quit.
13. Enter 1 to commit changes.

## Adjusting Connection Pooling for a Directory Server Instance

In addition to specifying load balancing of requests evenly across directory server instances using the Max Active Servers parameter, you can also adjust connection pooling within a specific directory server instance by entering the Initial and Maximum Connections for that specific server instance. The request is sent on the connection with the least load.

Similar to configuring load balancing, you must adjust directory pool connections from the following places:

- **Directory Profile Page**: You use this page to configure directory pool connections for user data.

  See "To adjust directory connection pooling from the directory profile" on page 3-9 for details.

- **ConfigureAAAServer Tool**: You use this tool to configure directory pool connections for policy data.

  See "To adjust directory connection pooling using the ConfigureAAAServer tool" on page 3-10 for details.

### To adjust directory connection pooling from the directory profile

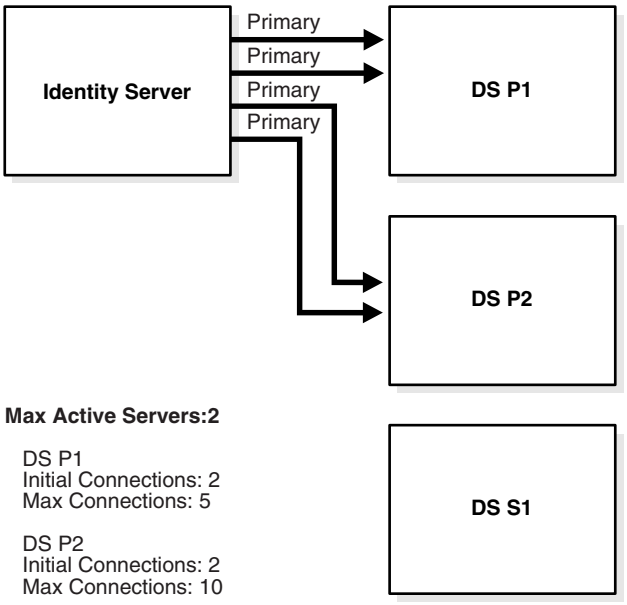
1. From the Directory Profile page, select a specific DB (directory) instance.
2. Enter the Initial Connections you want opened to this directory server.

   The default is 1.
3. Enter the Maximum Connections allowed to this directory server instance.

   If any of the connections to the directory server are lost, then Oracle Access Manager can attempt to open connections up to the maximum entered.

   When the first request is sent to a directory server, the initial number of connections is opened. When connections are lost, or when there are more service threads than connections, new connections are opened (up to the maximum). See Figure 3–4 for an example.

*Figure 3–4 Adjusting Connection Pooling for Directory Server Instances*




For example, assume there are 5 initial connections and 10 max connections to a directory server. When a service thread makes a request to the directory server, then 5 initial connections are created. Assume now, that there are 5 concurrent active service threads that require information from the directory server. They will use all of the existing 5 connections.

If the concurrent service threads increase beyond 5 more connections, Oracle Access Manager can create up to 10 max connections to that directory server. When the limit of 10 connections is reached, and there are 11 or more concurrent service threads, the 11th service thread will get one of the least loaded connections from the pool of 10 connections. The connection is then shared by more than one service thread.

---

**Note:** There is no general optimal value for the initial and maximum connections, given variables such as directory configuration, hardware, and so on. However, you should not set the number of connections higher than the number of threads for a given Oracle Access Manager server.

---

### To adjust directory connection pooling using the ConfigureAAAServer tool

1. See "To configure load balancing of configuration and policy data for the Access Server" on page 3-8.
2. Specify the Modify Common Parameters option when prompted.

## About Failover with Oracle Access Manager

Oracle Access Manager uses failover to provide uninterrupted service. Failover involves re-directing requests to another server when the original request destination fails.

Failover is accomplished by configuring primary and secondary servers and identifying specific parameters for the failover process. Failover can be configured between:

- Oracle Access Manager Web components to Oracle Access Manager Servers
  - WebPass requests to secondary Identity Servers
  - WebGate requests to secondary Access Servers
- Oracle Access Manager Servers to directory servers
  - Identity Server to secondary directory servers
  - Access Server to secondary directory servers
- Policy Manager to directory servers
  - Oracle Access Manager now provides the ability to perform failover from the Policy Manager to secondary directory servers.

### Primary Versus Secondary Servers

Oracle Access Manager components first attempt to connect to a primary server.

If the primary server is unavailable, a connection attempt may be made to a secondary server. Oracle Access Manager continues to attempt to connect to the primary server, and when the connection is re-established, the connection to the secondary server is dropped. Any server can be configured as a primary or a secondary server. For example, you could designate less powerful or geographically distant servers as secondary.

## Setting the Polling Interval

A "heartbeat" polling mechanism facilitates immediate failover to a secondary directory server when the number of connections in the connection pool falls below the specified threshold level. Additionally, a failback mechanism facilitates switches from the secondary directory server back to the primary server as soon as the preferred connection has been recovered.

The heartbeat feature polls all the primary directory server connections periodically to verify the availability of the directory service (and by implication, the network). You configure the polling interval by setting the Sleep For (Seconds) parameter for each directory profile.

When the host cannot be reached, further attempts to connect to that host are blocked for the specified Sleep For interval, rather than for the TCP timeout used previously.

A heartbeat_ldap_connection_timeout_in_millis parameter in globalparams.xml determines the timeout interval for establishing a connection. If the directory service is not available, the heartbeat mechanism immediately initiates failover to the secondary directory server. This enables failover without requiring an incoming directory service request and a subsequent TCP timeout.

The heartbeat_enabled parameter indicates whether the Identity and Access Servers should proactively identify when a directory server is down. Oracle recommends that you enable this function.

---

**Note:** If your network is slow and the heartbeat_ldap_ connection_ timeout_in_millis is set to a low value (for example, 10 milliseconds), the heartbeat mechanism can give an incorrect indication that directory is unreachable when it is up and working.

---

#### To set the polling interval in the Identity System

1. From the Identity System Console, select System Configuration, Directory Profiles.

   The Configure Profiles page appears.

2. In the Configure LDAP Directory Server Profiles section of the page, click the link for the profile that you want to modify.

   The Modify Directory Server Profile page appears. This directory server profile is used by the Oracle Access Manager servers that are selected in the Used By lists on this page.

3. Enter the interval in the Sleep For (Seconds) field.

#### To set the polling interval in the Access System

1. From the Access System Console, select System Configuration, then click Server Settings.
2. In the Configure LDAP Directory Server Profiles section of the page, click the link for the profile that you want to modify.

   The Modify Directory Server Profile page appears. This directory server profile is used by the Oracle Access Manager servers that are selected in the Used By lists on this page.

3. Enter the interval in the Sleep For (Seconds) field.

#### To set the connection timeout parameter

1. Open the following file:

   *Component_install_dir*/identity/apps/common/bin/globalparams.xml

   where *component_install_dir* is the location where the Access or Identity Server was installed.

2. Edit the value for the heartbeat_ldap_connection_timeout_in_millis parameter.

   Specifies the amount of time that Identity and Access Servers wait for a connection to be established with the directory server. If a connection with the directory server is not established within this time, the Identity and Access Servers assume that the directory is down or not reachable, and the servers start establishing connections with another directory server. The default value is 4000 (4 seconds). A value of -1 designates that the platform's connection timeout limit should be reached before attempting to establish a connection.

#### To turn the heartbeat mechanism on or off

1. Open the following file *Component_install_dir*/identity/apps/common/bin/globalparams.xml

   where *component_install_dir* is the location where the Access or Identity Server was installed.

2. Edit the value for the heartbeat_enabled parameter.

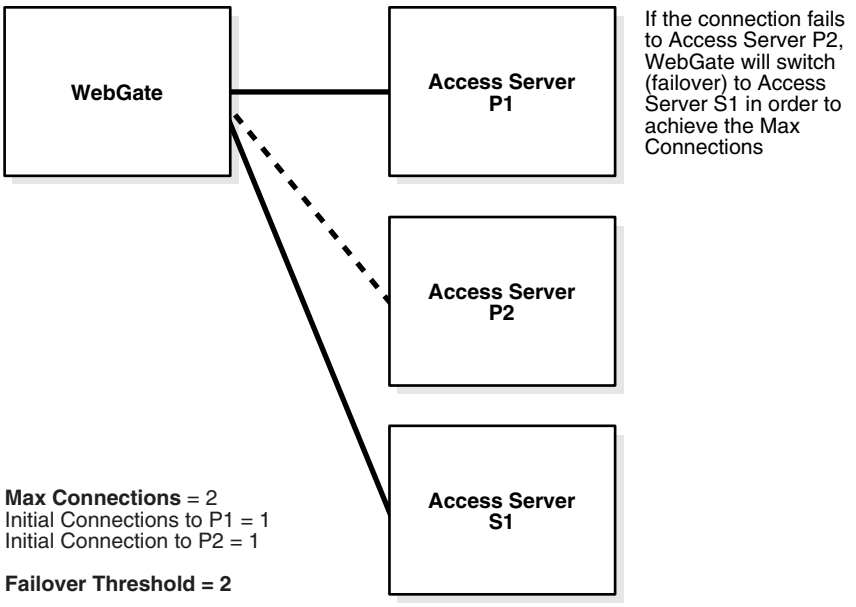
   This parameter activates or deactivates the heartbeat mechanism. By default, it is set to true (on). A value of false deactivates the mechanism.

## Configuring Failover of Web Components

Configuring failover enables a WebPass or WebGate to check the health of its connections, and failover to secondary Oracle Access Manager Servers in case one or

more primary servers go down. You configure failover from the perspective of the Web component. See Figure 3–5 for an example.

***Figure 3–5 Basic Failover Scenario between a WebGate and its Access Servers***




**Failover Threshold**: The key to configuring failover is the Failover Threshold field in the Web component configuration. This specifies the minimum number of live primary connections required. If the number of live connections drops below the failover threshold, then the Web component attempts to establish connections to its secondary servers in the order they are listed. The default is the maximum number of connections.

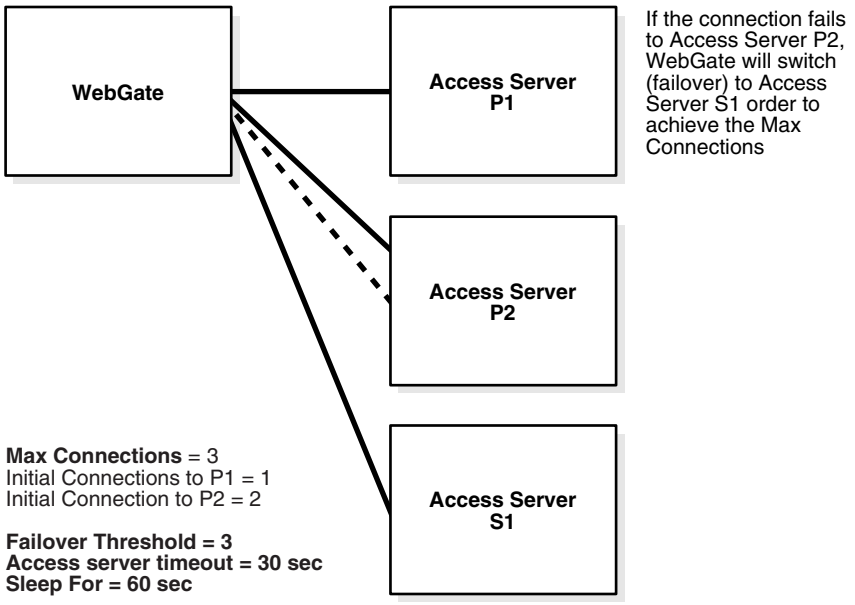
**Sleep For Interval**: The default interval is 60 seconds. After this interval, the WebPass or WebGate will check to see if the number of valid connections equals the maximum number of connections configured. If the number of valid connections does not equal the maximum number of connections (drops below the failover threshold), the Web component tries to establish connections to its secondary servers in the order they are configured. Then, at the interval specified, the Web component tries establishing a connection to the primary servers. When it establishes the connection, it drops the connections to the secondary servers after finishing the request it is servicing.

**Timeout Threshold**: Specifies how long (in seconds) the Web component waits for a non-responsive Oracle Access Manager server before it considers it unreachable and attempts to contact another. No value indicates there is no timeout, and the Web component will wait endlessly for a response from the Oracle Access Manager server. Leaving no timeout can potentially result in a hung session or default to a "TCP" timeout. For example:

- For an existing WebPass, enter a value for the Identity Server Timeout Threshold.
- For an existing AccessGate, enter a value for the Access Server Timeout Threshold.

In Figure 3–6 a WebGate communicates with two primary Access Servers (Access Server P1 and Access Server P2).

***Figure 3–6 Failover Scenario between a WebGate and its Associated Access Servers***




- The Maximum Connections is 3. Assume that Access Server P2 can handle additional load, and therefore two initial connections are established to it.
- The Failover Threshold is 3. If the number of valid connections drops below the failover threshold, WebGate tries to establish a connection to its secondary server, Access Server S1. Assume that Access Server P2 dropped its second connection.
- The Access Server Timeout Threshold parameter is set for 30 seconds. If the WebGate does not receive a response from the Access Server P2 in 30 seconds, it considers the server unreachable and attempts to failover to Access Server S2.
- The Sleep For parameter is set for 60 seconds. Every 60 seconds this WebGate checks whether the number of valid connections to primary servers equals the specified number of maximum connections. If not, WebGate continues its attempts to re-establish connections to failed primary Access Server P2.

### To configure failover for Web component requests

1. Access the WebPass or WebGate Web component where you will configure failover.

   For example:

   - From the Identity System Console, select System Admin, System Configuration, Configure WebPass, Name, Modify.
   - From the Access System Console, select Access System Configuration, AccessGate Configuration, All, Go, Name.

   See the *Oracle Access Manager Administration Guide* and the *Oracle Access Manager Access Administration Guide* for more information.

2. In the Failover Threshold field, enter the required number of live connections from the Web component to its primary Oracle Access Manager server.
3. Enter the Sleep For interval in seconds.

4. Enter a Timeout Threshold to specify how long (in seconds) the Web component waits for a non-responsive Oracle Access Manager server before it considers it unreachable and attempts to contact another:
   - **WebPass**: Enter a value for the Identity Server Timeout Threshold.
   - **AccessGate**: Enter a value for the Access Server Timeout Threshold.
5. Save your changes.

## About Failover Between Oracle Access Manager and Directory Servers

Failover for Oracle Access Manager Servers occurs when the number of live primary directory servers drops below the number in the Failover Threshold field. The instructions for configuring failover from Oracle Access Manager components to directory servers vary depending on the type of component (Identity Server, Access Server, Policy Manager), and whether you are configuring failover for user data or configuration data.

> **Note:** The Identity Server depends on a profile configured for the policy tree to do referential integrity for the policy directory. During Policy Manager setup, this profile is created for the policy directory for the Identity Server component. Whenever a user makes DN changes from the Identity System, the Identity Server uses this profile to update any DN references in the policy directory.

***Table 3–2 Supported Failover Configurations for Directory Servers***

| Component | Data Store | Operation |
|---|---|---|
| Identity Server | User | READ/WRITE—Configured in the directory profile. |
| | Configuration | READ/WRITE—Configured in the directory profile and XML configuration files. |
| Access Server | User | Read/WRITE—Configured in the directory profile. Write is only applicable if a password policy is enabled. |
| | Configuration | READ—Configured via the ConfigureAAAServer command line tool. |
| | Policy | READ—Configured via the ConfigureAAAServer command line tool. |
| Policy Manager | User | READ—Configured in the directory profile. |
| | Configuration | |
| | Policy | READ—XML configuration files. |

**Note:** The Identity Server depends on a profile configured for the policy tree to do referential integrity for the policy directory. During Policy Manager setup, this profile is created for the policy directory for the Identity Server component. Whenever a user makes DN changes from the Identity System, the Identity Server uses this profile to update any DN references in the policy directory.

## Configuring Directory Failover for User Data

You use the Directory Profile screen when configuring failover of Oracle Access Manager requests to directory servers containing user data.

**Failover Threshold**: The required number of live primary directory servers. If the number of primary servers drops below the failover threshold, Oracle Access Manager fails over to a secondary directory server.

When the Oracle Access Manager server sends a request on one of its directory connections, and the LDAP SDK returns a connection or server-down error, the directory server is assumed not to be available. If the number of primary directory servers drops below the failover threshold, then Oracle Access Manager attempts to establish connections to its secondary servers in the order they are listed.

If there is a primary server available when failover occurs, the Oracle Access Manager server fails over to the primary server first.

**Sleep For**: The number of seconds before the watcher thread wakes up and attempts to re-establish connections and create new connections if the connection was down. The watcher thread maintains a pool of good connections at all times.

By default, Oracle Access Manager creates a directory profile for each installed component. You need to access this page to configure directory failover. For more information on directory profiles, see the *Oracle Access Manager Administration Guide.*

### To configure directory failover for user data

1. Navigate to the System Console and access the Directory Profile page:

   For example:

   - From the Identity System Console, select System Configuration, Directory Profiles.
   - From the Access System Console, select System Configuration, Server Settings.

2. Select the link to the directory profile that contains connection information for the component and data where you want failover.
3. Enter the Failover Threshold.
4. In the Sleep For field, enter the number of seconds before the watcher thread wakes up and attempts to re-establish connections and create new connections if the connection was down.
5. Add the Database Instances and indicate their status as secondary servers.

## Configuring Directory Failover for Configuration and Policy Data

The instructions for configuring failover from Oracle Access Manager components to directory servers vary depending on the type of component (Identity Server, Access

Server, Policy Manager), and whether you are configuring failover for user data or configuration data. See Table 3–2 for details.

### Task overview: Configuring directory failover for configuration and policy data

1. See "Configuring Identity Server Failover for Configuration Data" on page 3-17 for details.
2. See "Configuring Access Server Directory Failover for Configuration and Policy Data" on page 3-18 for details.

## Configuring Identity Server Failover for Configuration Data

For the Identity Server, most configuration data is still managed through the XML configuration files. However, multi-language and referential-integrity data is managed through the Directory Profile page.

When the primary configuration data directory server is down, there is no way for the Identity Server to read any configuration entries. Therefore, the Identity Server reads failover.xml to obtain bootstrap secondary directory server information. See "Sample Failover.xml" on page 3-18 for an example.

### Task overview: Configuring Identity Server failover for Configuration data

1. Configure failover for configuration data.

   See "To configure Identity Server directory failover for configuration data" on page 3-17 for details.
2. Create the encrypted password.

   See "To create the encrypted password for the bind DN" on page 3-17 for details.
3. Configure failover.xml.

   See "To create failover.xml" on page 3-17 for details.

### To configure Identity Server directory failover for configuration data

1. From the Directory Profile page, enter failover specifications for the directory profile containing the configuration branch of the tree, as described in "Configuring Directory Failover for User Data" on page 3-16.
2. Create a file called failover.xml and add it to *IdentityServer_install_dir*/identity/oblix/config/ldap directory.

### To create the encrypted password for the bind DN

1. Locate the obencrypt tool in:

   AccessServer_install_dir/access/oblix/tools/ldap_tools/
2. Run obencrypt password.
3. Copy and paste the encrypted password into your failover file, as described in "To create failover.xml" on page 3-17.

### To create failover.xml

1. Copy and paste the existing sample_failover.xml template into the directory:

   *IdentityServer_install_dir*/identity/oblix/config/ldap

2. Open the copy with a text editor and add failover information for secondary servers using "Sample Failover.xml" on page 3-18 as a guide.
3. Copy and paste the encrypted password into your failover file.
4. Rename the copy to failover.xml.
5. Repeat as necessary for each applicable Identity Server.

### Sample Failover.xml

```
?xml version="1.0" encoding="ISO-8859-1"?>
<CompoundList xmlns="http://www.oblix.com" ListName="failover.xml">
<!-- # Max number of connections allowed to all the active ldap servers -- note
this is the same as Max Active Servers>
<SimpleList>
<NameValPair ParamName="maxConnections" Value="1"></NameValPair>
</SimpleList>
<!-- # Number of seconds after which we switch to a secondary or reconnect to a
restarted primary ldap server -->
<SimpleList>
<NameValPair ParamName="sleepFor" Value="60"></NameValPair>
</SimpleList>
<!-- # Max amount of time after which a connection to the ldap server will expire
-->
<SimpleList>
<NameValPair ParamName="maxSessionTime" Value="0"></NameValPair>
</SimpleList>
<!-- # Minimun number of active primary ldap servers after which failover to a
secondary server will occur -->
<SimpleList>
NameValPair ParamName="failoverThreshold" Value="1"></NameValPair>
</SimpleList>
<!-- # Specify the list of all secondary ldap servers here -->
<ValList xmlns="http://www.oblix.com" ListName="secondary_server_list">
<ValListMember Value="sec_ldap_server"></ValListMember>
</ValList>
<!-- # Specify the details of each secondary ldap server here -->
<ValNameList xmlns="http://www.oblix.com" ListName="sec_ldap_server">
<NameValPair ParamName="ldapSecurityMode" Value="Open"></NameValPair>
<NameValPair ParamName="ldapServerName"
Value="instructor.oblix.com"></NameValPair>
<NameValPair ParamName="ldapServerPort" Value="9002"></NameValPair>
<NameValPair ParamName="ldapRootDN" Value="cn=Directory Manager"></NameValPair>
<NameValPair ParamName="ldapRootPasswd" Value="000A0259585F5C564C"></NameValPair>
<NameValPair ParamName="ldapSizeLimit" Value="0"></NameValPair>
<NameValPair ParamName="ldapTimeLimit" Value="0"></NameValPair>
</ValNameList>
</CompoundList>
```

## Configuring Access Server Directory Failover for Configuration and Policy Data

The following procedures describe configuring failover from the Access Server to one or more directory servers containing configuration and/or policy data.

- To configure Access Server failover for configuration and policy data
- To add a failover directory server using the ConfigureAAAServer tool

### To configure Access Server failover for configuration and policy data

1. Using the Directory Profile page, enter the failover for the directory profile containing the Oblix branch of the tree.

   See "Configuring Directory Failover for User Data" on page 3-16 for general instructions.

2. Repeat the above procedure for the directory server containing policy data, if applicable.

   For the Access Server, most configuration data is still managed through the configuration files. However, multi-language and referential-integrity data is managed through the Directory Profile page.

3. Add failover information using the configureAAAServer tool, as described next.

### To add a failover directory server using the ConfigureAAAServer tool

1. From the command line, navigate to the folder where configureAAAserver tool is located.

   For example, the default location is:

   AccessServer_install_dir\access\oblix\tools\configureAAAServer

   where AccessServer_install_dir is the directory where the Access Server is located.

2. Run configureAAAServer tool using the following arguments:

   configureAAAServer reconfig AccessServer_install_dir

   For example:

   configureAAAServer reconfig "c:\Program Files\COREid1014\access"

3. Enter the number that corresponds to the Access Server security mode for Access Servers that will connect to the directory servers.

   - 1) Open
   - 2) Simple
   - 3) Cert

   You are then be asked if you want to specify failover information for configuration or Policy.

4. Select Yes (Y).

5. Specify whether the data is stored in the:

   - 1) Oblix tree
   - 2) Policy tree

6. Enter 1 to add a failover server at the following prompt.

   - 1) Add a failover server
   - 2) Modify a failover server
   - 3) Delete a failover server
   - 4) Modify common parameters
   - 5) Quit

7. Enter the following information:

   - Directory server name

- Directory server port

  For LDAP in an Active Directory forest environment, use port 3268 for Open mode and port 3269 for SSL mode. These two are the global catalog ports.

- Directory server login DN
- Directory server password
- Directory Server security mode
  - 1) Open
  - 2) SSL
- Priority Enter 2 as the priority
  - 1) Primary
  - 2) Secondary

8. Enter 5 to Quit.

   You are prompted to commit the changes.

9. Select Y to commit your changes.

ConfigureAAAServer automatically creates the following xml files in AccessServer_install_dir/access/oblix/config/ldap

- AppDBfailover.xml
- ConfigDBfailover.xml
- WebResrcDBfailoverxml

# 4

# Caching and Cloning

Caching information and cloning play key roles when you performance tune your Oracle Access Manager environment. This chapter contains the following topics:



## Cloned and Synchronized Components

Instead of using the command line or the installation GUI to install a Oracle Access Manager component, you can automatically install a component by cloning the configuration of an already installed component. *Cloning* creates a copy of a component on a remote system using an already-installed component as a template.

*Synchronizing* allows you to harmonize two installations of the same Oracle Access Manager component when one is more up-to-date than the other. Synchronization can be used to upgrade or repair installations on similar platforms.

See the *Oracle Access Manager Installation Guide* for more information on cloning and synchronizing.

## About Caching Recent Information

A cache is in-memory storage that keeps a copy of recently used information. Oracle Access Manager retrieves information that is used frequently from a cache instead of the directory. This allows information to be retrieved quickly. Oracle Access Manager uses several caches to improve performance.

The Identity and Access Systems contain different caches. The Oracle Access Manager system caches configuration settings and group information. The Access System caches information on password policies, policy domains, user credentials, and authentication schemes.

Some operations performed in the Oracle Access Manager impact the evaluation of access policies in the Access System. For example, if you deactivate a user in the Identity System, that change must be reflected in the Access System so the user does not have access to the resources that the Access System protects.

The optimum number of elements in a cache is a balance between:

- The number of elements required to be in the cache. This depends on the information being cached and system usage.

- The RAM available for caching.

The higher the number of elements in a cache, the greater the probability of finding the requested information and improving performance.

In a worst case scenario, if the cache uses a lot of memory and the machine on which the component runs does not have enough RAM, the operating system could spend too much time swapping pages in and out of memory. This would decrease performance.

The minimum cache size for optimal performance would be:

N = XR

where:

N = the number of entries in the cache. X = the number of new sessions per second. R = the average length of a session in seconds.

For example, if:

X = 100 sessions per second, and R = 600 seconds per session, then N = 60,000 number of entries in the cache

The default cache size for Oracle Access Manager components is sufficient for most installations.

## Triggering Cache Flush Events

Identity Servers can communicate with one another. They do so primarily for cache flush requests. When a cache is updated on one server, that server tells the other servers to update their caches. The Identity Server has several caches. Each can receive a cache flush request if data is modified:

- **Configuration Data**: Object classes, attributes, tabs, and panels. The Oracle Access Manager-specific data (OSD) cache is flushed automatically. In addition, you can request a cache flush as described in "To clear the OSD cache" on page 4-3.
- **Group Objects**: Automatic when you modify a group, or you can explicitly request a cache flush as described in "Caching Group Objects" on page 4-4.
- **DNs of User Objects**: This is the name cache. It is flushed automatically.
- **Read and Write Privileges for Attributes**: Automatic cache flush.
- **Delegated Identity Administration and Auditing**: Automatic cache flush.
- **Workflow Definitions and participants**: Automatic cache flush.
- **Basic Configuration Data Entered in Setup**: Configuration base (configuration DN) and searchbase. Automatic cache flush.
- **User-Defined Portal Inserts**: Invoked explicitly through a URL, as described in the *Oracle Access Manager Customization Guide.*

## Cache Timeout

The cache timeout specifies how long an element is held in the cache. When the time expires, the element is removed from the cache and must be retrieved from the directory.

If the information changes, but the cache does not, the Oracle Access Manager component uses outdated information until the information expires from the cache. Updating caches when you change the information is one way to avoid this problem.

If updating caches is not possible, such as when a user's information is changed through other software, then configuring a reasonable timeout for the cache is another solution.

A shorter timeout means information about the user is more recent, but the Identity Server and the Access Server must fetch data more often from the directory. This may reduce performance. Setting a long timeout means out-of-date information could remain in the cache for a longer period of time.

**Note:** In general, setting a lower cache timeout value means the data is more recent. However, a cache timeout value of 0 means the cache never expires.

## Caching System Information

Oracle Access Manager caches configuration information for specific data such as object classes, attributes, panels, and tabs used by Oracle Access Manager applications. This information is automatically cached during startup.

You must be a Master Administrator to view, reload, or clear the OSD cache.

### To view the OSD cache contents

1. Launch the Identity System Console.
2. Click the System Configuration tab and select View Server Settings.
3. In the View Server Settings page, click Cache.
4. In the Cache page, click View Cache Contents.

   The cached OSD information is displayed.

### To load the OSD cache

1. Launch the Identity System Console.
2. Click the System Configuration tab and select View Server Settings.
3. In the View Server Settings page, click Cache.
4. In the Cache page, click Load memory cache.

   The latest information is loaded into the cache.

To update existing information, you must first clear the cache and then reload the information.

### To clear the OSD cache

1. Launch the Identity System Console.
2. Click the System Configuration tab and select View Server Settings.
3. In the View Server Settings page, click Cache.
4. In the Cache page, click Clear memory cache.

   The cache is cleared.

The following sections provide more details on caching system information:

- Caching Group Objects
- Turning Off the Credential Mapping Cache

## Caching Group Objects

The Identity System provides a cache for group objects to boost performance for all group functions, especially those involving computation of parent (isMemberOf) groups, and children or nested groups.

A group is cached only when Oracle Access Manager makes its first request for that group. Subsequent requests for the group are directed to the cache. When you modify a group, it is removed (flushed) from the cache along with any parent, child, or nested groups. In a multi-server configuration, when a group is modified on one Identity Server, all other Identity Servers are notified so that they remove the group and all its dependent groups from their caches. When Oracle Access Manager receives another request for the modified group, it re-stores the group in the cache. This ensures that the cache has the most recent information for the group.

When you search for a group, Oracle Access Manager searches the directory, not the cache.

You manage the group cache using the groupdbparams.xml configuration file.

To configure group cache parameters

1. Navigate to the file groupdbparams.xml located in:

   *IdentityServer_install_dir* \identity\oblix\data\common

   where *IdentityServer_install_dir* is the directory where the Identity Server is installed.

2. Configure values for the parameters in the groupdbparams.xml file that control the cache-related functions described in Table 4–1.

***Table 4–1 Parameters in groupdbparams.xml***

| Parameter | Description |
|---|---|
| GroupCacheTimeout | The time period (in seconds) for which the object is valid.<br>By default, the Group Cache never times out.<br>Default = 0. |
| GroupCacheMax NumElements | The maximum number of group objects that can be stored in the cache.<br>Default = 10000. If the cache is at its maximum size, objects that are not accessed often are replaced by those that are more frequently accessed.<br>Consider the memory limitations of your system before setting the value for this parameter. |
| GroupCacheDisabled | Indicates whether the cache can be used or not.<br>Default = false.<br>A value of false indicates that the cache can be used. A value of true indicates that the cache cannot be used. If a cache is disabled, all reads of group objects will go to the directory. |
| GroupCacheRead FromMaster | Forces reads of groups so that the cache can do reads from the master replica in a replicated environment. This ensures that the cache does not contain old information in case a read from a consumer replica occurs before the consumer replica has received the updated information from the master replica.<br>Default = false<br>If value = true, it indicates that the cache should read from the master replica. |

On occasion, it may be necessary to remove a group from the cache to maintain cache integrity. For example, you may have to remove a group that has been modified using an application other than Oracle Access Manager to ensure that the updated group is read when the cache receives a read request for the group.

A Master Identity Administrator can clear the group cache through the Identity System Console, or with an Identity XML function named *flushGroupCache*. See the IdentityXML chapter in the *Oracle Access Manager Developer Guide* for more information.

#### To clear group caches from the Identity System Console

1. Launch the Identity System Console.
2. Click the Group Manager Configuration tab.
3. In the Group Manager Configuration page, click Group Cache.
4. In the Group Cache page, click the Clear group cache link.

   The group cache is cleared and a message appears confirming whether or not the operation was successful.

## Turning Off the Credential Mapping Cache

If a user is deactivated in Oracle Access Manager, you can deny that user access to protected resources. Because the Identity Server does not automatically flush the Access Server credential mapping cache when a user is deactivated, you must manually turn off the credential mapping cache.

When the cache is turned off, the credential mapping plug-in communicates directly with the LDAP directory whenever it must map a user to a user profile (DN). Otherwise, the plug-in uses the cached credentials for the user.

By default, the credential mapping cache is enabled.

To disable use of this cache, set the obEnableCredentialCache parameter to false in the credential_mapping plug-in. If you deactivate a user who is logged in, the user will still have access to resources based on policy information and prior authentication. However, if obEnableCredentialCache is set to false, when the user's session token expires or the user logs out, the user will not be allowed access to a protected resource the next time the user is authenticated.

Table 4–2 shows the possible values for the parameter.

***Table 4–2 Parameters for obEnableCredentialCache***

| Value | Meaning |
|---|---|
| no value | Credential mapping cache turned on |
| true | Credential mapping cache turned on |
| false | Credential mapping cache turned off |

Here is an example of the credential_mapping authentication plug-in with the credential mapping cache turned off:

- credential_mapping obMappingBase="%domain%",obMappingFilter="(&(&(objectclass=user)(samaccountname=%userid%))

(|(!(obuseraccountcontrol=*)) (obuseraccountcontrol=ACTIVATED)))",
obdomain="domain",obEnableCredentialCache="false"

To set the obEnableCredentialCache parameter

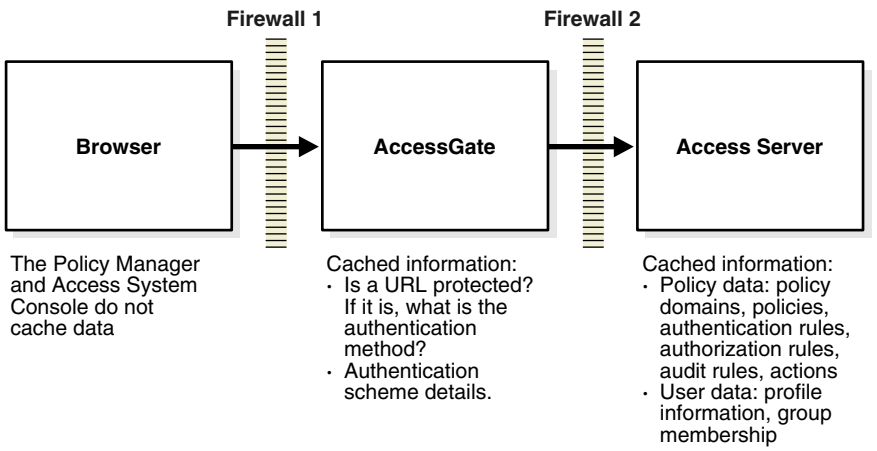
1. In the Access System Console, Access System Configuration, Authentication Management.
2. Select the authentication scheme you want to modify, then click Modify.
3. Click the Plugins tab.
4. Add the obEnableCredentialCache="false" parameter to the credential_mapping plug-in.

## Caching Access System Information

The Access System caches information on password policies, policy domains, user credentials, and authentication schemes.

Figure 4–1 illustrates how caching works in the Access System. It is not necessarily a deployment recommendation.

***Figure 4–1 Illustration of Caching in the Access System***




You can specify the maximum number of elements in a cache. The Access System ensures that the total number of elements in a cache never exceeds the maximum specified for that cache.

For example, suppose you have set the Access Server's user cache to contain a maximum of 100,000 elements. If a user is added to the system when the cache is full, the Access System removes the user element that has not been used in the longest time and adds information about the new user to the cache.

---

**Note:** WebGate and Access client configurations are cached in the Access Server. To reduce off-time network traffic between WebGate and Access Server and between Access Server and LDAP directory server, you can change the default configuration cache timeout. For details about reducing network traffic between components, see the *Oracle Access Manager Access Administration Guide.*

---

## Access Server Cache Configuration

The Access Server caches information that includes data on policy domains, policies, users, host identifiers, and password policies. When information is updated for any of these, ensure that the Update Cache box is selected to immediately update the Access Server caches.

### Information in a Policy Cache

An Access Server's policy cache contains information about policy domains, policies, authentication, authorization, audit rules, and actions.

An Access Server caches policy information for efficient runtime lookup. If policy information is not in the cache, the Access Server must obtain it from the directory. Obtaining data from the directory is slow when compared to the AccessGate obtaining information from the Access Server, so preferably all policy information should be cached.

When policy information is changed through Policy Manager or the Access System Console, Access Server caches can immediately be updated by selecting Update Cache. Policy cache timeout becomes important when this feature is not used. Cache timeout should be set to how quickly the policy information needs to be updated when Update Cache is not or cannot be used. The default policy cache timeout is two hours.

If the Access Server machine has sufficient memory to hold all the configured policy information, the maximum elements should be set according to whatever is the maximum of following:

- Total number of policy domains
- Total number of authentication rules, default or otherwise
- Total number of authorization rules, default or otherwise
- Total number of audit rules, default or otherwise

This configuration works for a typical deployment.

### Caching User Information

An Access Server's user cache includes information about:

- User Profile, which is required both in actions and rule-based access evaluation
- User's group-membership status, such as whether a user is member of a group or not

The Access Server caches a user's profile information and group membership information. The profile information includes both the information required in actions and the information required for authorization. For example, if the authorization rule for a policy allows access to anyone whose userLevel attribute is set to greater than 3, Access Server caches will include the userLevel attribute.

Similarly, an Access Server caches information about whether or not a user is a member of a group.

User and group information can be updated through the Identity Server or other applications. As user and group information changes, the Oracle Access Manager caches become out of date. For this reason, user cache timeout is crucial.

The timeout of the user cache should be proportional to the average time interval in which part of User Profile--those relevant to Access System--changes. Relevant parts of a user profile are:

- Attributes returned to AccessGate in actions
- Attributes used to control access
- Attributes used in dynamic group-membership definitions that may in turn be used to control access

The maximum elements should be equal to the number of different users that may access the system in the *Access Server user cache timeout* period of time.

You can configure the Identity Server to notify the Access Server automatically when user or group information changes. The Access Server's caches are then automatically flushed and updated with new information.

The following topics provide more details on caching:

- Automatically Flushing Access Server Caches
- Manually Flushing Access Server Caches
- Cache Configuration Using Replicated Directories
- AccessGate Cache Configuration

## Automatically Flushing Access Server Caches

The Identity System and the Access System use different user and group caches. An administrator may perform any of the following operations:

- Deactivating a user
- Changing user attributes
- Deleting a group
- Changing a password policy
- Changing redirect URLs

After any of the above operations take place, the directory server is updated with the new information. However, the Access Server may be unaware of these changes. For example, if you deactivate a user in Oracle Access Manager, that change should be reflected in the Access System so the user does not have access to its protected resources. To ensure that the Access Server is informed of changes in the Identity System, you can manually flush the Access Server's user cache. Or, you can configure the Identity Server to notify the Access Server of changes to user and group information. The Access Server caches are then automatically flushed and replaced with the latest information.

---

**Note:** The user cache is not automatically flushed when changes are made to group membership through a dynamic filter or through static membership.

---

### To flush the Access Server cache automatically

1. Navigate to *IdentityServer_install_dir*/identity/oblix/data/common/basedbparams.xml file

   where *IdentityServer_install_dir* is the directory where the Identity Server is installed.

2. In the basedbparams.xml file, locate the doAccessServerFlush parameter and set it to true.

3. Restart the Identity Server.
4. Add a dummy AccessGate using the configureAccessGate command line tool, as follows:

   configureAccessGate -i *IdentityServer_install_dir*/identity/AccessServerSDK -t AccessGate
5. Confirm that the Access Management Service for the AccessGate is turned on:
   a. From the Access System Console, click Access System Configuration, AccessGate Configuration, All, Go, click a link for the appropriate AccessGate, Modify.
   b. Set the Access Management Service to On, if needed.
6. Confirm that the Access Management Service for the Access Server is turned on:
   1. From the Access System Console, click Access System Configuration, Access Server Configuration, click a link for the appropriate Access Server, Modify.
   2. Set the Access Management Service to On, if needed.
7. Restart the Access Server.

## Manually Flushing Access Server Caches

If you choose not to implement automatic cache flushing for the Access Server, the Access Server's caches will contain outdated information until the cache timeout occurs. You can, however, manually flush the Access Server's user and password policy caches in the Access System Console. You can flush stored information on a specific password policy or on all password policies from the Access Server cache. You can also flush cached information on all redirect URLs.

### To flush the Access Server's user cache manually

1. Launch the Access System Console.
2. Navigate to Access System Configuration, User Access Configuration and click Flush User Cache.
3. To flush a specific user's profile and group information, click Select User.

   The Selector page appears.
4. To search for a user, enter the name and click Go.

   The search results are listed under Selector.
5. Click Add to select a user, or to select all listed users, click Add All.

   The user name is listed under Selected.
6. Click Done to leave the screen.

   The selected user's name appears on the Flush User Cache screen.
7. Click Flush Cache.

   A dialog box requesting confirmation appears.
8. Click OK to remove the user's cached information; click Cancel if you do not want to remove the user's cached information.

### To flush the password policy cache manually

1. Launch the Access System Console.

2. Navigate to Access System Configuration, Common Information Configuration and then click Flush Password Policy Cache.
3. If you changed any information for a password policy, select that policy from the list under Flush All Cached Information for a Specified Password Policy, then click Flush Cache.

   For information about the order of password policy evaluation, see the *Oracle Access Manager Administration Guide.*
4. To flush all of the password policies, or if you deleted a password policy from the Identity System, then click Flush Cache to delete cached information for all password policies.
5. If you changed any of the redirect URLs on the Password Policy Management screen, click Flush Redirect URL.

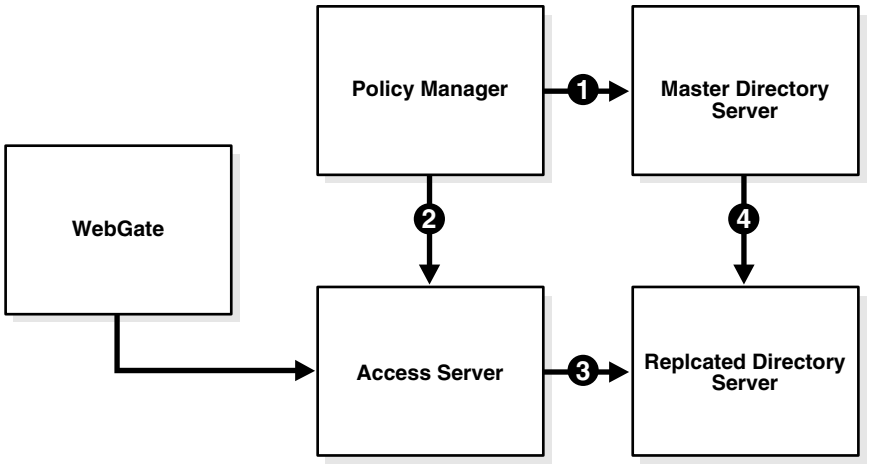
   For information about configuring the password redirect URLs, see the *Oracle Access Manager Administration Guide*.
6. Click OK to confirm your decision.

## Cache Configuration Using Replicated Directories

When you change data using the Policy Manager, it modifies data in the directory and notifies the Access Server to flush its cache. The Access Server flushes its cache and reads the updated information from the directory server.

In a replicated directory environment, modified data is passed from the master directory server to replicated directory servers. There is a time lag before the modified data is reflected in the replicated directory servers. If the Policy Manager communicates with the master directory server and the Access Server communicates with a replicated directory server, the Access Server may flush its cache and read data before the modified data is reflected in the replicated server. As a result, the Access Server may cache old data. Thus, in a replicated environment, policy evaluation by the Access Server can be incorrect if it is based on old data. Figure 4–2 illustrates a replicated environment.

***Figure 4–2 A Replicated Environment***




### Process overview: Replication

1. The Policy Manager modifies data in the master directory server.

2. The Policy Manager sends a signal to the Access Server to flush its cache.
3. The Access Server flushes its cache and reads data from the replicated directory server.
4. Modified data is replicated from the master directory server to the replicated directory server.

### Timeouts That Ensure Correct Behavior

To ensure that policy evaluation is accurate and that the Access Server reads the latest data, use the splTimeout parameter. The splTimeout parameter enables you to specify a timeout on the element being modified. It allows for the time lag between replication from the master directory server to the replicated directory server. The splTimeout parameter takes precedence over the cache timeouts specified in the Access Server Configuration page of the Access System Console.

The splTimeout value is specified in seconds.

If you do not specify a value for the splTimeout parameter, the Access Server flushes its cache and reads the data as soon as it receives a flush signal from the Policy Manager. The splTimeout parameter is in the file located in:

*AccessServer_install_dir*/access/oblix/apps/common/bin/globalparams.xml

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

## AccessGate Cache Configuration

When you configure an AccessGate, you can specify the maximum number of elements in the cache as well as the cache timeout.

Each AccessGate caches the following information:

- Details about an authentication scheme, such as:
  - How to challenge a user for credentials
  - Level of the scheme.

    For information about creating authentication schemes and their levels, see the *Oracle Access Manager Access Administration Guide*.
  - Redirection URL, if any
- Protection information about every resource--including the authentication scheme, URL, and operation--that is passed to the AccessGate, such as:
  - The authentication scheme key
  - The protected resource's URL
  - Whether the resource is protected in the system or not
  - If it is protected, the authentication method required for the resource.
- Details about an authorization scheme, such as:
  - The LDAP user attribute
  - The parameters specified in the authorization rules of the policy domain

When you configure or modify an authentication or authorization scheme, select Update Cache to update the AccessGate cache immediately with the updated scheme.

The number of URLs for which information is cached can be configured for each AccessGate. With out-of-the-box configuration, URL elements timeout every 30

minutes. When you make any policy changes that may change a URL's protection status or authentication method, select Update Cache to update the AccessGate cache immediately. This means the AccessGate's cache timeout period need not be short.

Assuming the system has sufficient memory, the maximum number of URLs cached should be at least equal to the number of distinct URLs processed by AccessGate in the amount of time specified in the Cache Timeout field.

For example http://www.myserver.com and http://myserver are treated as distinct URLs. If information about a URL is not in the cache, AccessGate makes a request to the Access Server. Fetching information from the Access Server takes longer than fetching information from a local cache, but not significantly longer.

# 5

# Reconfiguring the System

You can change basic components that you specified during Oracle Access Manager installation, such as the person object class or the directory server host. This chapter describes system-level reconfiguration.

This chapter includes the following topics:



## What Can Be Reconfigured

There are a number of basic system components that can be reconfigured:

- You can configure Oracle Access Manager against a different directory server (for configuration or policy data).
- You can specify a new person or group object class.
- You can change the class attribute for the person or group object class.
- You can reconfigure the following characteristics of the directory:
  - The host name
  - Port number
  - Domain name
  - Root DN
  - Root password
  - Configuration DN
  - Searchbase

## Performing the Reconfiguration

During installation, data that you specify is written to a number of areas, including the following:

- setup.xml
- configInfo.xml
- ois_server_config.xml
- The directory server

The following procedure describes how to reconfigure Oracle Access Manager so that it will work properly after you make any of the changes described in "What Can Be Reconfigured" on page 5-1.

### To update the system configuration

1. Shut down the Web server that runs the WebPass.
2. Stop the Identity Server Service.
3. Back up your directory configuration data by exporting it to an LDIF file.
4. Rename the following file to ensure that you have a backup copy:

   *Identity_Server_install_dir*/identity/oblix/config/ois_server_config.xml.bak
5. From the directory that you navigated to in the preceding step, back up and then delete the following files:
   - setup.xml
   - configInfo.xml
   - ois_server_config.xml
6. Copy the file ois_server__config.bak to ois_server__config.xml.

   This action allows you to change the configuration settings when you re-run the setup program later in this procedure. It causes the Identity Server to retrieve settings from *ois_server*__config.xml during setup instead of retrieving the settings from the directory. The information in ois_server__config.xml is migrated to the directory when the Identity Server is restarted.
7. In the branch of the directory where your policies are stored, locate the WebResrcDB container.
8. In the WebResrcDB container, delete the following entries:
   - The entry for WebPass.

     The cn for this entry is the ID that you supplied when installing WebPass. Example: wp1_50.
   - The entry for the Identity Server.

     The cn for this entry is the ID that you supplied when installing the Identity Server. Example: ois1_50.
   - The entry with a timestamp for its ID.

     Example: 20010815T16221897. This entry connects the WebPass and Identity Server components.
9. In the branch of the directory where your policies are stored, locate the DBAgents container and delete all entries under this container.
10. Restart the Identity Server Service.
11. Restart the Web server that runs the WebPass.
12. From your browser, access the Identity System Console:

    http://*server*:*port*/identity/oblix/
13. Rerun the setup program, as described in the following procedure for the Identity System and change any settings that you want to change.

The setup program will display the information that was previously configured for Oracle Access Manager. You can change the configuration information as needed when you rerun setup.

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

14. Restart the Identity Server.

    The information in ois_server_config.xml (the server name, port, administrator DN, password, searchbase, and configuration base) is migrated back to the directory and the information in the config.xml file is deleted.

### To rerun Identity System setup

1. Shut down all but one Identity Server if there is more than one running.
2. Go to the only remaining running Identity Server host and open the setup.xml file:

   *IdentityServer_install_dir*/identity/oblix/config/setup.xml
3. Remove the status parameter (or change the status parameter value from "done" to "incomplete"), as shown below:

   For example:

   ```
   <NameValPair ParamName="status" Value="incomplete"></NameValPair>
   ```
4. Save the file.
5. Restart the Identity Server.
6. From your Web browser, launch the Identity System Console.

   You will see a Setup page similar to the one that appears during the initial Identity System setup.
7. Initiate setup again and specify the new information.
8. After completing the setup, restart the other Identity Servers.

   The other Identity Servers should pick up the new information.

# 6

# Synchronizing System Clocks Across Time Zones

Correct operation of Oracle Access Manager depends on synchronizing the system clocks for all of its main components.

This chapter includes the following topics:

- Synchronization With NTP

> **Note:** This chapter provides a general discussion of NTP. It is provided for informational purposes only. Follow your own company's guidance for installing and configuring NTP.

## About Synchronization

As discussed in *Oracle Access Manager Installation Guide,* if you plan to install Oracle Access Manager components across multiple machines, you must make sure all system clocks are synchronized. This is particularly important if you will be running the software in Cert or Simple mode.

Synchronization is important for normal operations. Extremely accurate synchronization can also be a factor in security. For example, a time-based attack can be performed by changing the time on an expired cookie so that it appears to be earlier than the real time. Closely synchronized computers make it difficult to forge the timestamp on a cookie

## Synchronization With NTP

The Network Time Protocol (NTP) is a commonly-used tool for synchronizing system clocks. The following URL provides information on time synchronization.

`http://www.ntp.org/`

Also, see the comp.protocols.time.ntp news group for information on time synchronization.

NTP can typically synchronize the time on computers to within a few milliseconds. The following example shows the output of an ntp command on a typical workstation in an uncontrolled office environment. The example shows the high degree of synchronization that is achieved with this command:

```
 ntpq -p
   remote           refid           st t when poll reach   delay   offset    disp
==============================================================================
```

```
-qa.mycompany.co    clock.via.net    2 u  228 1024  377     1.33    0.121    5.13
#palantir.mycomp    clock.via.net    2 u  254 1024  377     1.42   -1.518    5.12
-panacea.company    clock.via.net    2 u  244  256  377     0.91    0.551    3.31
+test.mycompany.    nist1.aol-ca.tr  2 u  175  256  376     0.96    3.760    5.41
+test.mycompany.    pra3a.mycompany  3 u  441  256  372     1.12    3.043   65.31
+test.mycompany.    pra3a.mycompany  3 u  232  256  377     0.81    3.736    2.85
+test.mycompany.    pra3a.mycompany  3 u   27  256  377     0.93    3.787    3.34
+test2.mycompany    nist1.aol-ca.tr  2 u  232  256  377     0.74    3.722    2.92
*nist1.abc-ca.tr .ACTS.           1 u  180  256  377    11.53    1.097    2.88
-ntp-cup.externa .GPS.            1 u   96  256  377    38.48   -0.694    4.45
```

The offset field is in milliseconds. Note that all of these computers are within 5 milliseconds of the same time. The nist1 workstation is about 1 millisecond slower (1.097 milliseconds) than the time that the U.S. National Institute of Standards provides. This compares favorably with some radio broadcasts, which can be limited to approximately 10 millisecond accuracy due to varying atmospheric propagation delays.

UNIX operating systems typically ship with a version of NTP. It takes a small amount of configuration to enable these shipped versions:

- **Solaris:** you need to create an ntp.conf file.

  After you create this file using the Solaris conventions, xntp is started automatically when the computer is restarted.

- **HP-UX:** you can use sam to start ntp.

- **AIX:** you need to create an ntp.conf file and enable or create a start script in /etc/init.d (or the equivalent directory on AIX).

For all versions of UNIX, you can also get a current (and more secure) version of the NTP daemon from http://www.ntp.org/.

All UNIX machines use UTC (the pedant's name for GMT) internally and convert to the local time for displaying the time to users.

Windows computers typically perform time synchronization automatically with their domain controller using a Microsoft version of NTP. While NTP can synchronize the times, you also need to synchronize the domain controller with an official time source.

You can obtain a time service from many Internet Service Providers (ISPs). There is a list of open stratum-1 servers available from http://www.ntp.org/. Some of the servers that are listed at this site are open, for example, the servers at NIST. Other servers require an e-mail request before you use them to synchronize your network.

Windows computers keep the clock in local time, but the NTP synchronization programs compensate to convert to the appropriate time in each time zone.

## Synchronization with a GPS-based System

If having the best possible time match is important to your organization, you can purchase GPS-based clocks. The less expensive ones require some assembly. These clocks can be used to set your entire network to the same time. GPS technology requires very accurate times. Each GPS satellite contains 3 atomic clocks with continuous corrections provided from the ground to compensate for relativistic effects. In other words, an accurate estimate of the current time is developed as a side effect of determining where the GPS receiver is.