Oracle® Identity Federation

Administrator's Guide

10*g* (10.1.4.0.1)

B25355-02

July 2007

ORACLE®

Oracle Identity Federation Administrator's Guide, 10*g* (10.1.4.0.1)

B25355-02

Primary Author: Vinaye Misra

Contributors: Nageswara Alladi, Yogesh Bafana, Damien Carru, Smarto Chandra, Marc Chanliau, Wasim Chikhalia, Sumit Jeloka, Andrew Keim, Ari Kermaier, Charles Knouse, Amar Kumar, Naresh Kumar, Peifung Eric Lam, Eric Leach, Karl Miller, Valarie Moore, Joseph Morgan, Vamsi Motukuru, Maya Neelakandhan, Frank Villavicencio, Dai Vu, Aleksandr Yampolskiy

Contents

2 Planning Oracle Identity Federation Deployment

3 Installing Oracle Identity Federation

4 Deploying Oracle Identity Federation

5 Server Administration

6 Configuring Oracle Identity Federation

7 Additional Server Configuration

8 Monitoring Oracle Identity Federation

9 Advanced Topics

A Troubleshooting Oracle Identity Federation

B References

Glossary

Index

List of Tables

List of Figures

Preface

Oracle Identity Federation is a standalone, self-contained federation server that enables single sign-on and authentication in a multiple-domain identity network.

This preface contains these topics:

Audience

This document is intended for administrators of Oracle Identity Federation, who will deploy and manage the operation of the server in a federated network environment.

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:

- *Oracle Access Manager Access Administration Guide*
- *Oracle Application Server Single Sign-On Administrator's Guide*

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

This section describes changes since the 10.1.4.0.1 release.:

New Features in Oracle Identity Federation

With the 10.1.4.2 patch set release, Oracle Identity Federation provides:

- Attribute Name Mapping, Attribute Value Mapping, and Attribute Value Filtering
- a new command-line tool designed to perfom bulk delete operations of federation records from the Federation Data Store
- the ability to configure how Oracle Identity Federation responds when an error occurs in the logout flow
- a mechanism for requiring a client SSL certificate for all SOAP requests
- the no-fail-on-error option during Liberty 1.x/SAML 2.0 logout flow processing
- support for custom authentication engines
- the ability to configure the logout service to return the logout status
- the ability to configure SAML 2.0 Authentication Query Response and Assertion ID Request

Documentation Updates

The following new topics have been added to the *Oracle Identity Federation Administrator's Guide*:

- Sizing Guidelines
- Deploying Oracle Identity Federation with eTrust SiteMinder
- Deploying Oracle Identity Federation with a Sun Java System Web Server
- Configuring Oracle Identity Federation to Use IBM Tivoli Directory Server as the Data Store
- Integrating with Third-Party Identity & Access Management Modules
- Configuring Attribute Mapping
- Interoperating with Microsoft ADFS

- Logout Status
- Configuring SAML 2.0 Authentication Query Response
- Configuring SAML 2.0 Assertion ID Request
- Additional eTrust SiteMinder Configuration
- References

The following instructions have been updated:

- Using SSL with Oracle Identity Federation

Terminology Changes

Several terms that were previously used to describe Oracle Identity Federation or Oracle SHAREid features are no longer in use with release 10.1.4.2:

- SmartMaps - If a SHAREid SP could not map an incoming SSO assertion to a local user, the SmartMaps interface could be used to create a new user.

  This is not a supported feature in Oracle Identity Federation and the term is not used.

- SmartWalls - This was a best practice related to mapping an incoming SSO assertion to a user. SmartWalls was intended to thwart a user from one IdP from impersonating a user from another IdP by falsely asserting attributes for that user.

  As a rule, SAML 2.0 and Liberty do not use attribute mapping; instead they use opaque name identifiers that are not susceptible to this problem. The term SmartWalls has been replaced by "local user mapping".

- SmartMarks - This concept has been superseded in SAML 2.0 with the implementation of the SP-initiated IdP discovery using common domain cookies. While the feature is still applicable in the context of SAML 1.x, from now on it is referred to as SP-initiated IdP discovery.

1

Introduction to Oracle Identity Federation

This chapter provides an introduction to federated identity management and describes key features and benefits of Oracle Identity Federation. It contains the following sections:

Federated Identity Management

Although **single sign-on (SSO)** enjoys wide adoption for its ability to cut down the need for redundant logins, mere SSO is insufficient for companies which must operate in a *federated* environment - that is, an environment where services need to be shared with business partners while protecting those same services from unauthorized access.

A federated environment enables business partners to achieve integration in the identity management realm, by providing a mechanism for companies to share identity information across their respective security domains.

This section provides an introduction to federated identity management. It contains these sections:

Challenges of User Identity Management

Single sign-on for Web-based applications is a business goal that has been approached and solved in various ways over the past several years. Still, enterprises continue to face major challenges in managing their information systems cost-effectively:

- The proprietary nature of many of these solutions means that the protocol and software are particular to a specific vendor, implementer, or deployment scenario, and do not readily lend themselves to easy interoperability with other single sign-on systems.
- The proliferation of content formats, supply chains, customer management systems, and user data stores poses security and maintenance concerns. For example, a financial services company serving healthcare organizations may need to manage hundreds of thousands of employee accounts and incur substantial costs related to provisioning new users, responding to events like forgotten

passwords, and other record maintenance. On the part of the user, disparate authentication systems mean having to remember and perhaps write down multiple IDs and passwords, with the obvious risks inherent in that practice.

- Ever-expanding and increasingly dynamic end-user communities demand that information and applications be accessible not only to employees, but to vendors, partners, and customers as well. Traditional efforts to provide access require maintaining individual user accounts within the organization, leading to duplication of identity data along with administrative and compliance issues.

Federated identity management is the evolution of the SSO paradigm in response to users' growing needs for access to computing resources and services that reside outside their own company's boundaries. In a federated environment, enterprises offering such a service can reliably obtain identity information about an individual or other entity from the user's home organization or security domain. This provides twin benefits:

1. The end user does not need to supply login credentials to access each entity where business is conducted. This also eliminates the need to remember and manage multiple logins/passwords. (Users still need accounts at the sites so that the accounts can be linked.)
2. Enterprises do not need to create additional accounts to manage the identities of users who are already known to a partner organization. In the example cited earlier, the service provider could simply leverage the employee data maintained internally by its client healthcare organizations.

See Also: For a detailed definition, see **federated identity management (FIM)**

Federation Use Cases

Use cases in this section explain how federation can provide a seamless end-user experience by authenticating once for multiple applications, to overcome the real-world business problems of the kind described above.

Use Case 1: Single Sign-On to Partner Site

Figure 1–1 Single Sign-On from Employee Portal to Partner

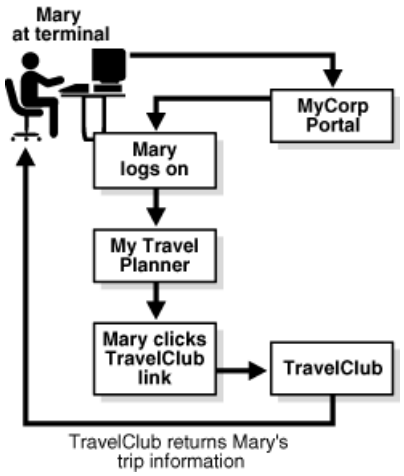


Figure 1–1 describes a situation where Mary, an employee of MyCorp, wishes to plan an upcoming business trip. She is able to achieve this seamlessly, in a single session, by performing the following steps:

1. Mary accesses her company's MyCorp employee portal from her terminal.
2. The portal, which is enabled with **WS-Federation**, presents her with a sign-on dialog.
3. After Mary signs on, the portal returns a page personalized with her information.
4. Mary commences travel planning by clicking on a link inside the portal for TravelClub, which is a partner organization providing access to a range of travel services for MyCorp employees. Mary has already established a federated relationship with TravelClub.
5. TravelClub requires authentication before Mary can access her account, and requests the same from MyCorp, which returns the necessary identity information to the travel site. Mary is then automatically authenticated to the TravelClub site. TravelClub returns a page with Mary's travel account information.
6. When Mary is done, she can log out of both her TravelClub and MyCorp sessions using a single global logout feature at the MyCorp home page.

In this way, Mary is able to authenticate once to her company's web site, connect with another site and perform necessary tasks, without the need for any additional authentication at the second site.

Use Case 2: New Federated Account at Partner Site

Figure 1–2 Creating a Federated Account

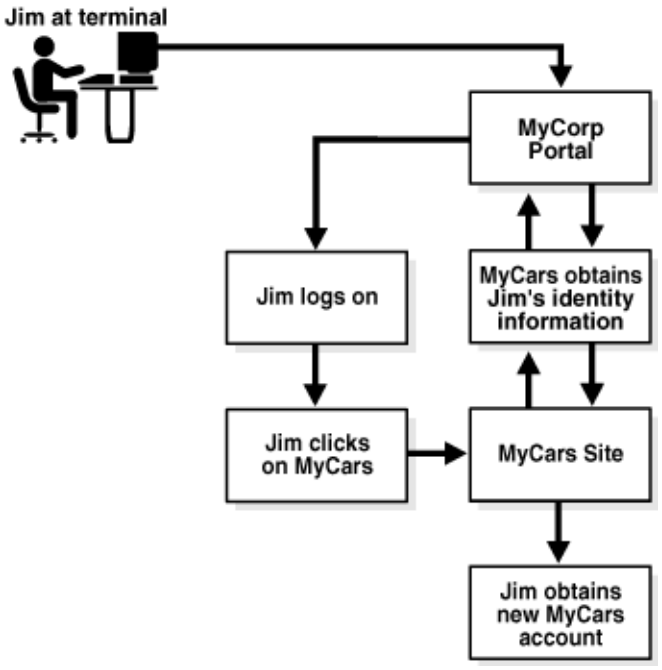


Figure 1–2 illustrates a use case where Jim, another employee at MyCorp, wishes to set up a new account at MyCars, an external site which provides discount auto repair services to MyCorp employees. The steps are as follows:

1. Jim signs on to the MyCorp portal.
2. After doing some work within the portal, Jim elects to move to the "Vendors" page of the portal to look for automotive services, and clicks on the MyCars link.
3. Information is required to set up a new account at MyCars. With Jim's permission, MyCars communicates with MyCorp to obtain information relevant to Jim's identity.
4. Jim now has an account at MyCars, which he can access in a manner similar to that outlined in the previous use case.

These use cases are typical examples of the application of federated single sign-on and federated identity management. In subsequent sections we take a closer look at the key concepts of federation technology, and how they are leveraged in Oracle Identity Federation.

Concepts and Terminology

The following discussion introduces key concepts of federated identity management.

Principal

A **principal** is any entity capable of using a service and capable of acquiring a federated identity.

A Principal is a person (a "user"), a group of users such as a corporation, or a system entity whose identity can be authenticated.

This is one of the three primary roles defined in the identity federation protocols supported by Oracle Identity Federation. The others are Identity Provider (IdP) and Service Provider (SP).

Domains

A **domain** is a web site and applications that enable a principal to utilize resources. A federated site acts as an identity provider (source domain under some specifications), a service provider (destination domain), or both.

Identity Provider

An **identity provider** is responsible for managing, authenticating, and asserting a set of identities within a given circle of trust.

Identity providers are service providers offering business incentives so that other service providers affiliate with them.

This is one of the three primary roles defined in the identity federation protocols supported by Oracle Identity Federation. The others are Principal and Service Provider.

An identity provider is sometimes also referred to as a source domain in the SAML 1.x protocols. From this perspective, it is the point at which requests originate; users from a source domain request permission to access resources on sites residing on destination domains.

Service Provider

A **service provider** provides services or goods to a Principal while relying on an Identity Provider to authenticate the Principal's identity.

A service provider is also referred to as a Relying Party (SAML) or a destination domain. From the domain perspective, a service provider contains the resource that users from source domains wish to access.

Service providers are organizations offering Web-based services to users. This broad category includes practically any organization on the Web today, for example:

- internet portals
- retailers
- financial institutions
- government agencies
- not-for-profit organizations
- entertainment companies
- transportation providers

This is one of the three primary roles defined in the identity federation protocols supported by Oracle Identity Federation. The others are **identity provider** and **principal**.

> **Note:** A single organization may be both an identity provider and a service provider, either generally or in the context of a given interaction.

Circle of Trust

A circle of trust is a trust relationship among a set of **identity provider**s and **service provider**s that allows a Principal to use a single federated identity and single sign-on when conducting business transactions with providers within that set.

Organizations affiliate together into circles of trust based on federation technology and operational agreements that define trust relationships between the parties.

For example, in the enterprise circle of trust, the identity provider is a company leveraging employee network identities across the enterprise.

Another example is the consumer banking circle of trust, where the user's bank has established business relationships with various other service providers, allowing the user to wield his bank-based network identity with those providers.

> **Note:** There can be multiple identity providers within a given circle of trust.

Identity Federation

Identity federation is the linking of two or more accounts a principal may hold with one or more identity providers or **service provider**s within a given circle of trust.

When users federate the otherwise isolated accounts they have with businesses (known as their local identities), they are creating a relationship between two entities, an association comprising any number of service providers and identity providers.

Single Sign-On

Single sign-on enables users to sign on once with a member of a federated group of identity providers and service providers (from a provider's point of view, with a

member of a circle of trust) and subsequently use various resources among the group without the need to sign on again.

Under the Liberty protocol, performing a single sign-on operation between a Principal, an SP and an IdP requires that:

- the SP and IdP reside in the same circle of trust, that is, they have a trusted business relationship.
- the Principal have local accounts at both the SP and IdP and that these two accounts be federated.

Federation Protocols

In building a federated architecture that addresses interoperability, assurance, and trust concerns across security domains, the following protocols have emerged as useful building blocks for identity management integration:

- Security Assertions Markup Language (SAML), a standard developed by OASIS, provides a means for exchanging security information between online business partners. In a typical exchange of SAML messages between two parties, one party acts as a Relying Party while the other acts as an Asserting Party. The Asserting Party asserts information about a given subject, such as whether a subject has been authenticated, whether a subject is authorized to perform a certain action, and so on.

  The Relying Party uses information provided by the Asserting Party to make security-related decisions regarding a subject, such as what types of access to a specific resource the subject should be granted, and so on.
- The Liberty Identity Federation Framework (Liberty ID-FF) is a standard developed by the Liberty Alliance. Its objectives include:
  - enabling simplified sign-on through federated network identification
  - supporting permission-based attribute sharing to enable users to have control over the use and disclosure of their personal identity data

The SAML and Liberty protocols provide a framework for exchanging information between security domains, for provider introduction or "handshake," and for managing identity events such as federated sign-on and global logout. The protocols include:

- SAML 1.0 and 1.1, which define a format for security data exchange known as an assertion, and profiles which provide the means for using the assertions
- Liberty ID-FF 1.0 and 1.1, which extend SAML 1.0 to provide additional profiles for account linking, introductions, and other tasks
- Liberty ID-FF 1.2
- SAML 2.0, which incorporates Liberty ID-FF 1.2
- WS-Federation, which enables different security realms to federate by brokering trust of identities, user attributes, authentication between participating Web services

These standards provide an XML-based framework for communicating security information across domains. Benefits include:

- loose coupling of domains, with no need to synchronize or replicate user data between directories

- a platform- and technology-neutral approach that removes barriers to cross-domain integration
- broad support from application server, web services management, and security products and vendors, and adoption by a growing number of organizations

This section explains fundamental SAML concepts and provides details about federation protocols. It contains these topics:

- SAML Basics
- Evolution of the Federated Identity Standards
- SAML 1.x
- SAML 2.0
- Liberty ID-FF 1.1
- Liberty ID-FF 1.2
- WS-Federation

See "Federation Protocol Profiles" on page 1-15 for a description of the profiles supported in Oracle Identity Federation.

SAML Basics

In a typical exchange of **SAML** messages between two parties, one party acts as a Relying Party while the other acts as an Asserting Party.

The Asserting Party *asserts* information about a given subject, such as whether a subject has been authenticated, whether a subject is authorized to perform a certain action, and so on. The Relying Party uses information provided by the Asserting Party to make security-related decisions regarding a subject, such as what types of access to grant the subject to a specific resource, and so on.

SAML Assertions

SAML associates a principal with additional identity information that can be used to determine the principal's access rights within a specific domain. Every SAML document has an **assertion** element containing such an association.

SAML defines three kinds of assertions, which are declarations of one or more facts about a subject:

- authentication assertions, which state that the user has proven her identity by a particular method at a specific time
- attribute assertions, which contain specific details about the user such as an employee number or an account number
- authorization assertions, which state the resources a user can access and under what conditions they can be accessed

Assertions are coded statements generated about events that have already occurred.

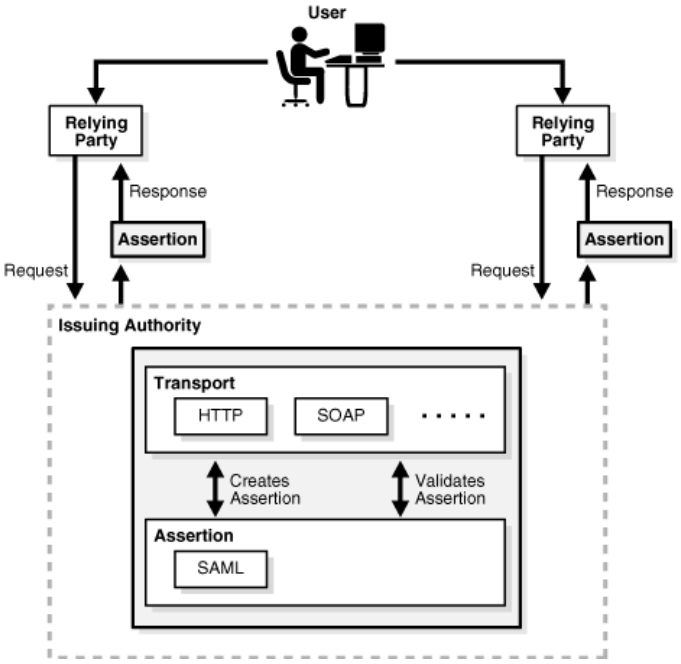
> **Note:** While SAML makes assertions about credentials, it does not actually authenticate or authorize users.

Example 1–1 on page 1-9 shows a typical SAML 1.0 authentication assertion wrapped in a SAMLP response message:

SAML Request and Response Cycle

In a typical SAML cycle, the relying party, which needs to authenticate a specific client request, sends a SAML request to its issuing authority. The issuing authority responds with a SAML assertion, which supplies the relying party with the requested security information. This cycle is illustrated in Figure 1–3.

Figure 1–3 The SAML Request-Response Cycle



For example, when a user signs into a SAML-compliant service of a relying party, the service sends a "request for authentication assertion" to the issuing authority. The issuing authority returns an "authentication assertion" reference stating that the user was authenticated by a particular method at a specific time. See Example 1–1 for a typical authentication assertion.

The service can then pass this assertion reference to other relying party sites to validate the user's credentials. When the user accesses another SAML-compliant site that requires authentication, that site uses the reference to request the "authentication assertion" from the issuing authority, which states that the user has already been authenticated.

At the issuing authority, an assertion layer handles request and response messages using the SAML protocol, which can bind to various communication and transport protocols (HTTP, SOAP, and so on). Note that while the client always consumes assertions, the issuing authority can act as producer and consumer since it can both create and validate assertions.

SAML Protocol Bindings and Profiles

SAML defines a protocol for requesting and obtaining assertions (SAMLP). Bindings define the standard way that SAML request and response messages are transported across the issuing authorities and relying parties by providing mappings between SAML messages and standard communication protocols. For example, one defined transport mechanism for SAML requests and responses over HTTP is Simple Object Access Protocol (SOAP). This enables the exchange of SAML information across several Web services in a standard manner.

A profile describes how SAML assertions are embedded into and extracted out of standard frameworks and protocols. Web browser profiles for single sign-on and SOAP profiles for securing SOAP payloads are some of the available profiles.

Evolution of the Federated Identity Standards

In response to the needs of access control vendors for a standard mechanism to speed development of cross-domain single sign-on applications, efforts by the OASIS standards organization produced SAML 1.0, the first such standard, in 2002. The Liberty Alliance, working on open standards for federated identity, built upon the SAML specifications to produce the Liberty 1 standard. At the same time, another consortium of vendors and companies worked on evolving authentication and authorization standards for Web service-oriented applications. Subsequent efforts led to the development of the WS-Federation standard, and extension and co-evolution of the SAML and Liberty standards in parallel. These different standards are described below.

See Also:

- O'Reilly's xml.com Web site for a broad survey of standards and technology trends at http://webservices.xml.com/security/.
- Project Liberty specifications at https://www.projectliberty.org/resources/specifications.php#box2

SAML 1.x

SAML 1.0 defines two key concepts:

1. a security token format, known as an assertion, which associates a given identity with specific access rights
2. profiles that describe ways to package these assertions to provide single sign-on

SAML 1.1 updates SAML 1.0 with feedback and corrections.

Example 1–1 shows a typical SAML 1.0 authentication assertion wrapped in a SAMLP response message:

Example 1–1 Sample SAMLP Response Containing a SAML 1.0 Authentication Assertion

```
<samlp:Response
       MajorVersion="1" MinorVersion="0"
       ResponseID="128.14.234.20.90123456"
       InResponseTo="123.45.678.90.12345678"
       IssueInstant="2005-12-14T10:00:23Z"
       xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion"
       xmlns:samlp="urn:oasis:names:tc:SAML:1.0:protocol">
       <samlp:Status>
           <samlp:StatusCode Value="samlp:Success" />
       </samlp:Status>
       <saml:Assertion
           MajorVersion="1" MinorVersion="0"
           AssertionID="123.45.678.90.12345678"
           Issuer="IssuingAuthority.com"
           IssueInstant="2005-12-14T10:00:23Z" >
           <saml:Conditions
               NotBefore="2005-12-14T10:00:30Z"
               NotAfter="2005-12-14T10:15:00Z" />
```

```
            </saml:Conditions
            <saml:AuthenticationStatement
                AuthenticationMethod="urn:oasis:names:tc:SAML:1.0:am:password"
                AuthenticationInstant="2005-12-14T10:00:20Z">
                <saml:Subject>
                    <saml:NameIdentifier NameQualifier="RelyingParty.com">
                      john.smith
                    </saml:NameIdentifier>
                    <saml:SubjectConfirmation>
                        <saml:ConfirmationMethod>
                            urn:oasis:names:tc:SAML:1.0:cm:artifact-01
                        </saml:ConfirmationMethod>
                    </saml:SubjectConfirmation>
                </saml:Subject>
            </saml:AuthenticationStatement>
        </saml:Assertion>
</samlp:Response>
```

Liberty ID-FF 1.1

Liberty ID-FF 1.1 is an extension profile of SAML 1.0 intended to support identity federation and single sign-on. The Liberty framework defines three types of actors: **identity provider**s, **service provider**s, and **principal**s.

Within the Liberty framework, a group of Service Providers and Identity Providers may affiliate themselves with one another in what is known as a circle of trust. A Principal can then use a single federated identity and single sign-on when conducting business transactions with providers within the circle of trust.

Liberty ID-FF 1.2

Liberty ID-FF 1.2 is an extension profile of SAML 1.1 intended to support identity federation and single sign-on. The Liberty ID-FF 1.2 framework defines four types of actors: **identity provider**s, **service provider**s, affiliations (or group of service providers), and **principal**s.

SAML 2.0

SAML 2.0 includes support for single sign-on based largely on the framework developed by the Liberty Alliance ID-FF specifications.

Although the concept of identity federation is not present in the specifications, SAML 2.0 promotes the existence of a name identifier for a specific use. SAML 2.0 supports a number of named profiles that largely mirror the functionality of the Liberty ID-FF 1.2 profiles, on top of the name identifiers inherited from SAML 1.x.

Example 1–2 shows a SAML 2.0 authentication assertion:

Example 1–2 SAML 2.0 Authentication Assertion

```
<saml:Assertion
       xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
       Version="2.0"
       ID="id-V01vmFAGUOKmKVJh9-hQ-gsPhX8-"
       IssueInstant="2005-10-06T21:03:17.375Z">
       <saml:Issuer Format="urn:oasis:names:tc:SAML:2.0:nameid-format:entity">
           http://issuingauthority.example.com/
        </saml:Issuer>
        <!-- signature by the issuer over the assertion -->
       <ds:Signature>
```

```
            ...
        </ds:Signature>
        <saml:Subject>
            <saml:NameID
              Format="urn:oasis:names:tc:SAML:2.0:nameid-format:persistent">
                id-V9l9N2S4nA8KlHd0X9Df3KYKm4E-
            </saml:NameID>
            <saml:SubjectConfirmation Method=
              "urn:oasis:names:tc:SAML:2.0:cm:bearer">
                <saml:SubjectConfirmationData
                  NotOnOrAfter="2005-10-06T21:03:32.375Z"
                  Recipient="http://issuingauthority.example.com/fed/sp/art20"
                  InResponseTo="id-G2mgYgtGH9gu8Nwo8KwxPYrpXKE-"/>
              </saml:SubjectConfirmation>
        </saml:Subject>
        <saml:Conditions NotBefore="2006-04-27T16:40:49Z"
          NotOnOrAfter="2006-04-27T17:05:49Z">
            <saml:AudienceRestriction>
                <saml:Audience>
                  http://serviceprovider.example.com:80/fed/sp
                </saml:Audience>
            </saml:AudienceRestriction>
        </saml:Conditions>
        <saml:AuthnStatement
            AuthnInstant="2005-10-06T21:01:03.451Z"
            SessionIndex="1448745">
            <saml:AuthnContext>
                <saml:AuthnContextClassRef>
                    urn:oasis:names:tc:SAML:2.0:ac:classes:
                      PasswordProtectedTransport
                </saml:AuthnContextClassRef>
            </saml:AuthnContext>
        </saml:AuthnStatement>
</saml:Assertion>
```

WS-Federation

The WS-Federation specification is "an integrated model for federating identity, authentication, and authorization across different trust realms and protocols." WS-Federation is a Web services-oriented standard which supports profiles for passive requestors, such as Web browsers, as well as active requestors such as SOAP-enabled applications.

Note: Oracle Identity Federation currently supports passive requestors for WS-Federation.

About Oracle Identity Federation

This section shows how Oracle Identity Federation allows users of Oracle Identity Management products, as well as customers new to the Oracle APS stack, to engage in business associations across heterogeneous environments using various sources of user authentication. It contains the following topics:

- Features and Benefits of Oracle Identity Federation
- Architecture
- High-Level Processing Flow

Features and Benefits of Oracle Identity Federation

Oracle Identity Federation is a standalone, self-contained federation server that enables single sign-on and authentication in a multiple-domain identity network. Oracle Identity Federation supports multiple federated identity protocols including the Liberty ID-FF and SAML protocols. This allows users to federate in heterogeneous environments and business associations, whether or not they have implemented other Oracle Identity Management products in their solution set.

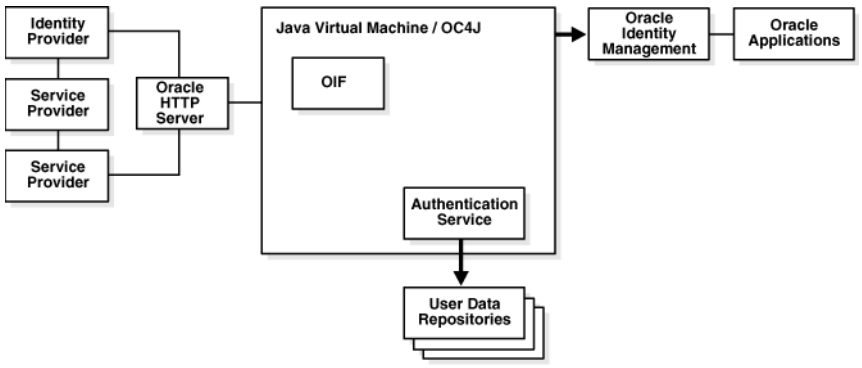
Key features of Oracle Identity Federation include:

- the ability to implement cross-site access and authentication in an environment containing both identity providers and service providers
- the ability to configure, enable, and disable external sites
- the ability to access applications at destination sites using a single sign-on
- support for these leading federation protocols:
  - Liberty ID-FF 1.1
  - Liberty ID-FF 1.2
  - SAML 2.0, including SAML 2.0 attribute responder functionality
  - WS-Federation
- integration with Oracle Access Manager and OracleAS Single Sign-On
- support for cross-protocol single sign-on and sign-out
- support for affiliations, which reduce the number of federations by allowing service providers to share their federation information
- integration with Oracle Internet Directory and support for:
  - a range of authentication engines, including Oracle Access Manager and CA eTrust SiteMinder
  - user data repositories, including LDAP Stores such as Microsoft Active Directory and Sun Java System Directory Server
  - relational databases
- support for X.509 certificate validation

Architecture

Figure 1–4 shows the architecture of Oracle Identity Federation (OIF) and its relationship to other federation components. Here Oracle Identity Federation is a member of a circle of trust containing other identity providers and service providers, which can be additional Oracle Identity Federation instances or third-party providers.

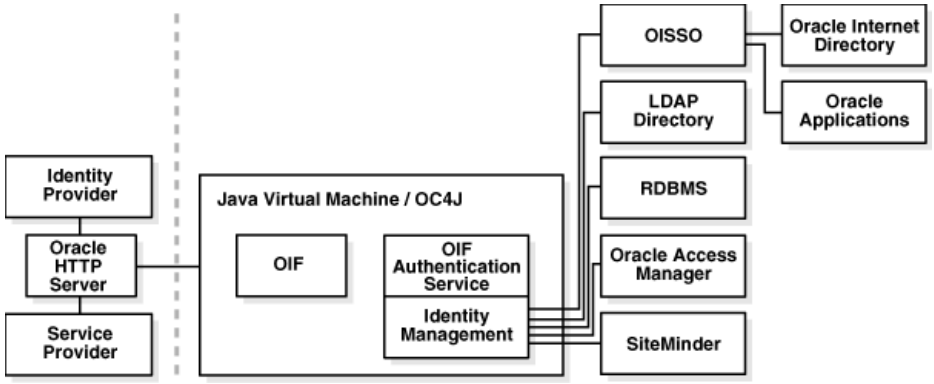
Figure 1–4 Oracle Identity Federation



Note: A single Oracle Identity Federation instance can only support one IdP and one SP. Therefore, the two service providers shown in the figure cannot both belong to the same instance of Oracle Identity Federation. Rather, they are peer providers participating in the circle of trust.

Oracle Identity Federation includes a self-contained, lightweight authentication service. Based on **IdMBridge**, this service - illustrated in Figure 1–5 - is deployed in a WAR (Web Application aRchive) file with Oracle Identity Federation and runs in the same Java Virtual Machine as the server.

Figure 1–5 Oracle Identity Federation 3rd Party Integration



Oracle Identity Federation can communicate with a range of authentication mechanisms and user data repositories:

1. Oracle Identity Management

   You can configure the Oracle Identity Federation authentication service to enable single sign-on access to resources protected by OracleAS Single Sign-On or Oracle Access Manager, including:

   - Oracle Collaboration Suite
   - Oracle E-Business Suite
   - PeopleSoft modules
   - and more

In addition to Oracle Application Server Single Sign-On (with the Oracle Internet Directory user repository) or Oracle Access Manager (with various repositories), this configuration can also leverage third-party access management solutions when OracleAS Single Sign-On is deployed for use with those solutions.

Note: In an environment where Oracle Identity Federation and OracleAS Single Sign-On both protect resources, you can configure either component to serve as the authentication mechanism when a user requests access to a protected resource. For example, Oracle Identity Federation can forward authentication requests to OracleAS Single Sign-On; or, OracleAS Single Sign-On can request Oracle Identity Federation to locate an appropriate identity provider. For details, see *Oracle Application Server Single Sign-On Administrator's Guide*.

Likewise, environments containing both Oracle Identity Federation and Oracle Access Manager provide similar functionality.

---

2. Data Stores

   You can configure Oracle Identity Federation to access:

   - LDAP directories
   - RDBMS databases
   - Oracle Access Manager
   - eTrust SiteMinder

High-Level Processing Flow

Before looking at Oracle Identity Federation features in detail, it will be instructive to consider, at a high level, the processing flow that makes it possible to manage user access in such a federated environment.

Users typically access applications in multiple domains through a corporate portal. For example, Alpha Corp. could have a Portal Server in place, to manage Alpha's user logins, page personalization, and so on. The portal server might consist of homegrown logic running within an application server, or it might be a commercial product. Its partner, Beta Corp., may serve its technical database application with a "MyBeta.com" type of portal. In that case, each company would operate its own portal server.

A user logs into the Alpha portal, with the access being managed by a web access management (WAM) system such as Oracle Access Manager, CA SiteMinder, or some other product. Next the user clicks on a resource that is actually hosted by Beta Corp.

At this point Oracle Identity Federation connects to the WAM system to trigger the creation of a session, which is typically represented by an encrypted session token, that is set on the user's browser as a session cookie. The process by which Oracle Identity Federation triggers this session within the WAM system depends on the specifics of the WAM vendor (see "Authentication Engines" on page 2-12). At a high level, though, the process involves Oracle Identity Federation supplying attributes - obtained from the incoming assertion - to the WAM system through an API, which the WAM system uses to uniquely map to an identity in its local identity store.

Upon successful mapping, the WAM returns a successful response along with the session token (a SAML assertion). Starting with the initial user cookie (generated from the initial single sign-on at the portal), the federation engine thus builds a SAML

assertion that can be presented to Beta Corp., which would then receive the SAML assertion and map it to its local authorization system.

For a more detailed description of Oracle Identity Federation processing flows, see Chapter 2, "Planning Oracle Identity Federation Deployment".

Federation Protocol Profiles

Identity providers and service providers exchange assertions using profiles and services defined in a federation protocol such as SAML or Liberty ID-FF. Assertion functions include:

- establishing secure connections
- conveying authentication data across those connections
- receiving and interpreting assertions from other SAML domains

Profiles describe the types of exchanges required to transfer assertions between IdP and SP. This section takes a closer look at the assertion profiles available in Oracle Identity Federation:

- Browser POST Profile
- Browser Artifact Profile
- SOAP Binding
- Browser HTTP Redirect Profile
- Name Identifier Profiles
- SAML Attribute Sharing Profile
- Federation Termination Profile
- Global Logout Profile
- WS-Federation Passive Requester Profile

Browser POST Profile

The SAML Browser POST profile sends a full assertion from an identity provider to a service provider without the use of an artifact. Oracle Identity Federation sends the assertion to the user's browser as a hidden variable in the HTML form, and the browser then posts the assertion to the destination site.

In SAML 2.0, the HTTP POST Binding provides a framework for the embedding and transport of SAML protocol messages under real-world communication protocols.

Browser Artifact Profile

Some browsers may limit the number of URL characters they can handle. The SAML Browser Artifact profile accommodates this by transmitting data using a compact reference to an assertion, called an artifact, instead of sending the full assertion. The recipient of the artifact then uses an artifact resolution protocol to obtain the full assertion referred to by the artifact.

In SAML 2.0, the HTTP Artifact Binding provides a framework for the embedding and transport of artifacts under real-world communication protocols.

SOAP Binding

A binding is the mapping of abstract message exchanges into real-world messaging or communication protocols. As an example, the SAML SOAP Binding defines how SAML protocol messages can be communicated within SOAP messages.

Browser HTTP Redirect Profile

The Browser HTTP Redirect profile indicates to the requesting party that the requested resource resides under a different URL.

In SAML 2.0, the HTTP Redirect Binding uses the HTTP redirect response to send data in URL query string parameters through a user's browser from one provider to another. The amount of data that can be sent is limited by the maximum URL allowed by the browser, so this is usually employed for shorter messages and not full assertions.

Name Identifier Profiles

Name Identifier Profiles define how providers communicate with each other when one of the providers wishes to update the name identifier assigned to one of their common users. These profiles allow a service provider or identity provider to specify (or register) a *name identifier* for a principal. Peer providers, for their part, must use this name identifier when communicating with other providers about the principal.

Oracle Identity Federation supports these SOAP/HTTP and HTTP-redirect name identifier profiles:

- Liberty ID-FF 1.1 IdP-Initiated Register Name Identifier Profile
- Liberty ID-FF 1.1 SP-Initiated Register Name Identifier Profile
- Liberty ID-FF 1.2 IdP-Initiated Register Name Identifier Profile
- Liberty ID-FF 1.2 SP-Initiated Register Name Identifier Profile
- SAML 2.0 IdP-Initiated Manage NameID Profile for Name Identifier Update
- SAML 2.0 SP-Initiated Manage NameID Profile for Name Identifier Update

SAML Attribute Sharing Profile

SAML 2.0 provides an Attribute Query/Response protocol for retrieving a principal's attributes. The SAML Attribute Sharing Profile defines how to use this protocol with SOAP binding, to enable attribute retrieval when the subject is authenticated using an X.509 certificate.

To see how this is protocol is used, consider a principal who needs to access a web resource maintained at a service provider. Authentication is achieved by presenting the user's federated credential in the form of a trusted X.509v3 certificate, along with proof of possession of the associated private key. One common example of this is the client certificate authentication feature of the SSL (Secure Sockets Layer) protocol used between a user's browser and a web server.

The service provider may require additional information about the principal to determine authorization for some privileged resource. To get this information, the SP utilizes the `SubjectDN` from the principal's X.509v3 certificate to query an identity provider for the required attributes. When the IdP returns these attribute values, the SP can make an authorization decision based on the additional data. Thus, the profile provides additional protection of resources from unauthorized access.

WS-Federation Passive Requester Profile

WS-Federation provides support for integration of identity, authentication, and authorization across security domains and protocols. The WS-Federation passive requestor profile defines the use of this specification when clients for federation services include such passive requestors as Web browsers that support the HTTP protocol.

Federation Termination Profile

Users have the ability to terminate a federation, typically by using a link on the identity provider's or service provider's Web site. If initiated at the IdP, this action tells the SP that the IdP will no longer provide the user's identity information to the SP. If initiated at the SP, this action tells the IdP that the user requests that the IdP no longer provide that user's identity information to the SP.

Note: Federation termination is also referred to as defederation.

The Federation Termination Profile specifies how identity providers and service providers are notified of federation termination.

Oracle Identity Federation supports these federation termination profiles:

- Liberty ID-FF 1.1 IdP Initiated Federation Termination Notification Profile
- Liberty ID-FF 1.1 SP Initiated Federation Termination Notification Profile
- Liberty ID-FF 1.2 IdP Initiated Federation Termination Notification Profile
- Liberty ID-FF 1.2 SP-initiated Federation Termination Notification Profile
- SAML 2.0 IdP Initiated Manage NameID Profile for Name Identifier Deletion
- SAML 2.0 SP Initiated Manage NameID Profile for Name Identifier Deletion

Global Logout Profile

As the name implies, this profile provides support for global logout. The identity provider maintains a list of all the service providers at which a given user has logged in based on assertions provided by the IdP. When the user invokes logout, the IdP sends each SP a logout request for the user, achieving global logout with respect to that IdP.

The steps in the logout process are:

1. Either the user or a peer provider initiates the logout request.
2. The Oracle Identity Federation IdP sends a logout request to the service providers or identity providers where the user was logged in. The type of message sent depends on the type of single sign-on; for example, if single sign-on was based on Liberty 1.2, Oracle Identity Federation will send a Liberty 1.2 `LogoutRequest`.
3. Oracle Identity Federation receives a logout response from the provider to whom it sent the message.
4. Oracle Identity Federation sends the next logout request (step 2).
5. When the user is logged out of all the providers, Oracle Identity Federation logs the user out of the server.
6. For WS-Federation logout, Oracle Identity Federation displays a success page to the user. Liberty 1.x and SAML 2.0 logout profiles send the user back to the requesting peer provider that sent the original logout request.

Oracle Identity Federation supports SOAP/HTTP and HTTP-Redirect global logout profiles for these protocols:

- Liberty ID-FF 1.1 IdP-Initiated Single Logout Profile
- Liberty ID-FF 1.1 SP-Initiated Single Logout Profile
- Liberty ID-FF 1.2 IdP-Initiated Single Logout Profile
- Liberty ID-FF 1.2 SP-Initiated Single Logout Profile
- SAML 2.0 IdP-Initiated Single Logout Profile
- SAML 2.0 SP-Initiated Single Logout Profile
- WS-Federation Passive Requester Logout Profile

Affiliations

A Liberty 1.2/SAML 2.0 affiliation consists of service providers that are part of a logical group.

An affiliation is not a concrete entity or server, but a logical provider; thus, no server can act as an affiliation. Rather, an affiliation will be used by service providers when performing protocol message exchanges - in this case, the affiliation will be viewed as a logical provider, but the sender/receiver of messages for this affiliation will be a concrete service provider. In this way, the service providers participating in the affiliation act as the affiliation, and will have access to all the federation information about that logical provider.

For details on implementing affiliations in Oracle Identity Federation, see "Configuring and Using Affiliations" on page 6-60.

Cryptographic Provider

Oracle Identity Federation expects XML signing/encryption keys to be provided by means of a PKCS#12 wallet. Oracle Identity Federation does not provide configuration knobs for swapping out the underlying crypto provider

Example of Federation Event Flow

This section describes a typical message flow in a federated interaction.

Elaborating on the use case in Figure 1–1 on page 1-3, consider that Mary is already authenticated at mycorp.com, and goes to travelclub.com where she is not logged in. travelclub.com requires Mary to be authenticated before she can access her local account, and redirects Mary with a SAML 2.0 message to mycorp.com requesting a single sign-on for travelclub.com. Since Mary is already logged in at the identity provider, mycorp.com retrieves Mary's account and federation data and redirects her back to travelclub.com. Using the Provider Identifier `mycorp.com` and the User Identifier `xyz123` provided with the redirect, travelclub.com can uniquely retrieve Mary's federation data and her local account.

Supported Standards and Applications

For information about the platforms and product versions supported by Oracle Identity Federation, see the appropriate certification matrix as follows:

1. Log in to MetaLink at `https://metalink.oracle.com`.
2. Click the **Certify** tab.

3. Click **View Certifications by Product**.
4. Select the Application Server option and click **Submit**.
5. Select the Oracle Identity Management option and click **Submit**.
6. Click **Oracle Identity Management Certification Information 10g (10.1.4.0.1)**.
7. Under General Oracle Identity Management Certification Information, click Oracle Identity Federation Certification.

2

Planning Oracle Identity Federation Deployment

This chapter outlines Oracle Identity Federation deployment considerations and helps you understand installation options. It contains these sections:

Architecture Options

In planning to deploy Oracle Identity Federation, you should understand the server architecture, the operating environment, and the role that your server will play in a federated exchange network. This section outlines the architectural aspects of Oracle Identity Federation deployment, including:

Role in Federation

As described earlier, an Oracle Identity Federation instance in a federated network can serve as an identity provider, a service provider, or both.

Identity Provider Role

When a user wishes to access a protected resource in the federated network, the service provider for that resource directs the user to Oracle Identity Federation, which acts as IdP for authentication. Oracle Identity Federation works with an authentication engine to obtain credentials and authenticate the user. Oracle Identity Federation can

now assert the user's identity to the resource (SP), which logs the user in and provides the requested application.

Service Provider Role

A user tries to access a resource protected by an authentication engine such as Oracle Application Server Single Sign-On, which redirects the user to Oracle Identity Federation. In an SP role, Oracle Identity Federation redirects the user to an identity provider such as a portal for global authentication. The IdP portal can now obtain credentials, authenticate the user, and redirect back to Oracle Identity Federation, which then retrieves the asserted identity from the IdP. Oracle Identity Federation redirects the (authenticated) user to the authentication engine, which grants access to the protected resource.

Topology

You can install Oracle Identity Federation in one of two network configurations:

- Hub-and-Spoke
- Peer-to-Peer

Hub-and-Spoke

In a hub-and-spoke network, multiple sources (identity providers) communicate with a single destination (service provider).

Figure 2–1 A Hub-and-Spoke Federation Network

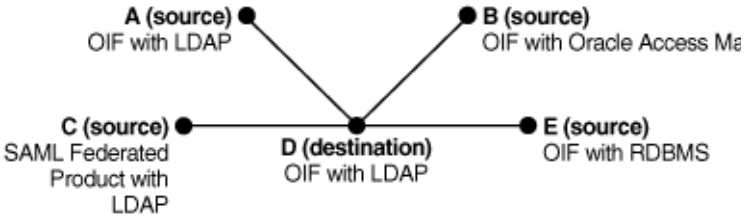


Figure 2–1 shows a hub-and-spoke network in which Company D serves as a destination for companies A, B, C, and E. Company D may offer resources, such as financial services, to which employees of the other companies need access.

Note: The hub-and-spoke is a topology, in which the source (IdP) or destination (SP) can interchangeably serve as either hub or spoke. Thus the roles in Figure 2–1 could just as well be reversed to depict a network consisting of an IdP hub and SP spokes.

Peer-to-Peer

In a peer-to-peer network, multiple domains serve as hubs.

Figure 2–2 A Peer-to-Peer Federation Network

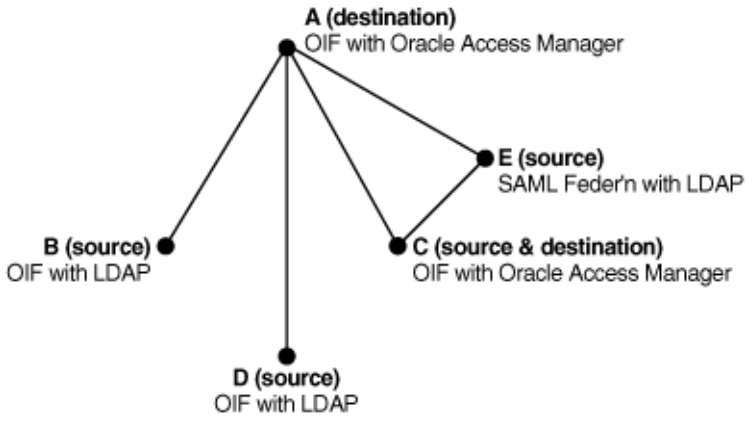


Figure 2–2 shows a peer-to-peer network in which company A is a service provider for companies B and C, while Company C is a service provider for Companies D and E. Thus Company C serves as both an identity provider and a service provider. Company E serves as identity provider for users who consume services at Companies A and C.

Proxy Server

You must decide what components you will put in the DMZ and whether to use a proxy server. If you put Oracle Identity Federation behind the firewall, the proxy must forward requests and responses to the federation server, enabling transparent access to the server from an external network such as the internet.

Oracle Identity Federation configuration varies depending on the type of profile being implemented.

> **See Also:** For more information about setting up a proxy server for Oracle Identity Federation, see "Setting Up a Proxy for Oracle Identity Federation" on page 9-20

POST Profile with Proxy in SP DMZ

The POST profile sends the full assertion to the SP over HTTPS. Both IdP and SP are configured to communicate through their SSL ports. When using the POST profile in production, the SP uses a proxy server in the DMZ.

Artifact Profile with Proxy in IdP and SP DMZ

When using the browser artifact profile, the IdP sends an artifact (an identifier) rather than an actual assertion. The SP receives the artifact and requests the full assertion thereafter.

If you elect to use a proxy, note that proxies must be used for both IdP and SP in order to implement this profile. The proxies serve as receiver and responder services, handling the exchange of artifacts, assertion requests and assertions, and forwarding those objects to their respective providers.

Server Security

Oracle Identity Federation provides secure communication using:

- SSL Encryption
- Certificate-based Authentication

- Certificate Repository and Validation

SSL Encryption

Oracle Identity Federation provides secure SSL communication between partner domains. SSL encryption is an option you can enable or disable for the server instance at installation time.

> **Note:** For more information about SSL and related system security topics, see the *Oracle Application Server Security Guide*.

Certificate-based Authentication

For initial setup and testing, identity providers and service providers can use default self-signed certificates. Before going into production, however, you will want to ensure that your installation is set up to use third-party CA certificates.

Certificate Repository and Validation

Oracle Identity Federation provides a repository where you can store a list of trusted CAs as well as **certificate revocation list (CRL)**s.

If certificate validation is enabled for the server, Oracle Identity Federation will validate every certificate used to verify incoming signatures for the Liberty 1.1, Liberty 1.2, and SAML 2.0 protocols.

To validate a certificate, the server tries to locate the certificate or its issuer as a trusted certificate, and checks that the certificate is not in a CRL.

See Also:

- "Editing Server Properties" on page 6-5 for information about enabling certificate validation
- "Editing the Certificate Validation Store" on page 6-63 for details about the certificate repository

Protocol

When installing Oracle Identity Federation, you need to decide the federation protocols that your server will support. Oracle Identity Federation works with these protocols:

- Liberty ID-FF 1.1
- Liberty ID-FF 1.2
- SAML 1.0
- SAML 1.1
- SAML 2.0
- WS-Federation

As the Oracle Identity Federation administrator, you need to determine which federation protocols you will utilize for your server.

For more information, refer to these resources:

- "Federation Protocols" on page 1-6
- "Liberty ID-FF 1.1" and "Liberty ID-FF 1.2" on page 1-10

Profiles and Bindings

This section discusses profiles and bindings, and contains these topics:

- Supported Protocols
- Choosing a Profile

Supported Protocols

Having selected the protocol(s) your federation server instance will support, you must choose which protocol profiles and security transport bindings you will implement.

Table 2–1 and Table 2–2 show the list of supported protocol profiles and security transport binding combinations that can be enabled for an Oracle Identity Federation instance.

Table 2–1 Oracle Identity Federation Profiles, and Bindings for Liberty 1.x and SAML 2.0

Function	Profiles/ Bindings	Liberty 1.1	Liberty 1.2	SAML 2.0
Single Sign-On	Artifact	x	x	x
Single Sign-On	HTTP Post	x	x	x
Logout	HTTP Redirect	x	x	x
Logout	HTTP Post			x
Name ID Registration	HTTP Redirect	x	x	x
Name ID Registration	HTTP Post			x
Name ID Registration	SOAP	x	x	x
Federation Termination	HTTP Redirect	x	x	x
Federation Termination	HTTP Post			x
Federation Termination	SOAP	x	x	x
Attribute Retrieval	SOAP			x

Table 2–2 Oracle Identity Federation Profiles and Bindings for SAML 1.x and WS-Federation

Function	Profiles/ Bindings	SAML 1.0/1.1	WS-Federation
Single Sign-On	Artifact	x	
Single Sign-On	HTTP Post	x	x
Logout	HTTP Redirect		x

Choosing a Profile

Under the SAML and Liberty protocols, you can specify whether providers should exchange Oracle Identity Federation assertions using the artifact profile or the POST profile. These profiles represent different methods for secure exchange of assertions.

This section discusses:

Using the Artifact Profile

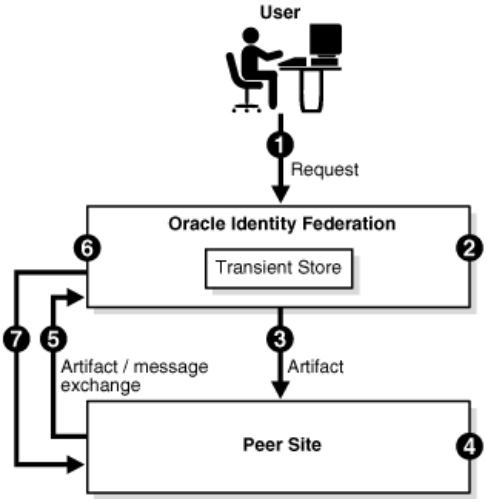
Here are some items to keep in mind when considering the artifact profile:

- The artifact profile is less resource-intensive than the POST profile because the latter uses XML signatures.
- The identity provider's SAML components must reside in the DMZ.

Artifact Profile Request Processing

Figure 2–3 shows the process by which requests are processed under the artifact profile.

Figure 2–3 Artifact Profile Processing Flow



The processing flow takes this path:

1. A user performs a request at Oracle Identity Federation (acting as the IdP).
2. Oracle Identity Federation authenticates the user and creates an artifact which includes an identifier for the IdP that is known to the SP.

   The message to be sent is stored in a repository at the server, with the artifact as a key for retrieval.

   Note: Depending on the installation, the repository may reside either in memory or in a relational database. When using replicated Oracle Identity Federation servers for high availability, the repository must reside in a database.

3. The server redirects the user to the peer site with the artifact. The artifact profile is used to carry the message.

4. The peer site decodes the artifact and deduces that Oracle Identity Federation is the originating site.
5. The peer site contacts the IdP Oracle Identity Federation, sends the artifact and asks the server to dereference it.
6. Oracle Identity Federation retrieves the message from the repository using the artifact.
7. Oracle Identity Federation sends the message to the peer site for processing.

Note: This scenario illustrates IdP-initiated single sign-on. When the request is SP-initiated, the user directly requests the resource at the service provider.

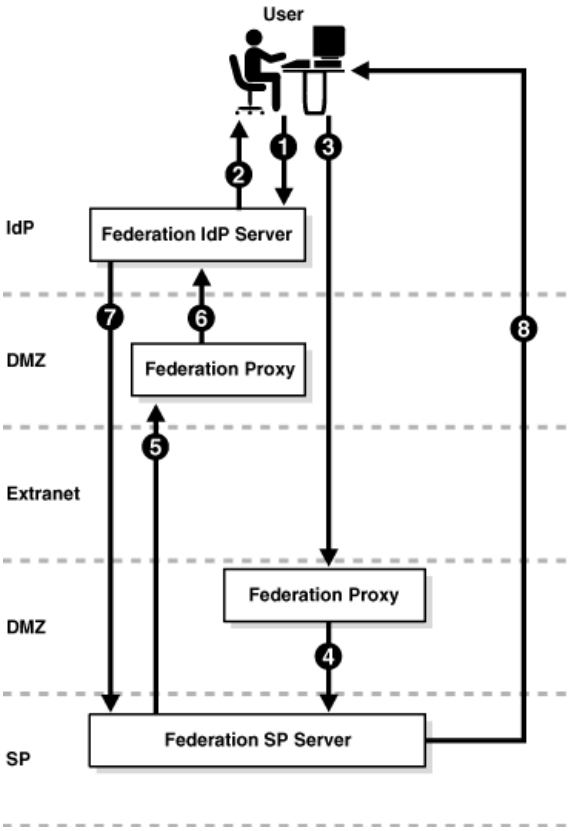
In contrast with user entries and user federation records, artifact objects are considered as transient data. Because of its transient status, the artifact has a limited lifetime and will be removed from the repository after a certain time.

Artifact Profile With Proxy

As shown in Figure 2–4, you can configure Oracle Identity Federation with proxies for IdP and SP servers when using the artifact profile. In this secure environment, the proxies are located within the DMZ.

> **See Also:** For more information about setting up a proxy server for Oracle Identity Federation, see "Setting Up a Proxy for Oracle Identity Federation" on page 9-20

Figure 2–4 Artifact Profile Processing with Proxy



The processing flow is as follows:

1. A user issues a request at the Oracle Identity Federation IdP server.
2. Oracle Identity Federation authenticates the user and creates an artifact which includes a short IdP server identifier. The server redirects the user with the artifact to the receiver service on the SP proxy server.
3. The user's browser sends the request containing the artifact to the URL of the service provider's receiver service, which is located on the proxy in the SP's DMZ.
4. The proxy forwards the request to the Oracle Identity Federation SP server.
5. The SP contacts the IdP's responder service, which is located on the proxy in the IdP's DMZ, sends the artifact, and asks the IdP to dereference it.
6. The proxy forwards the request to the IdP.
7. The IdP retrieves the message from the repository using the artifact, and sends it to the SP.
8. The SP server creates a user session and redirects the user's browser to the desired resource.

For testing purposes, you can configure the peer providers to communicate over open ports. Secure SSL ports are recommended for production, however, and the peer IdP and SP administrators must have exchanged and installed each other's CA certificates. These certificates are used to encrypt and decrypt requests and responses exchanged between the respective federation servers

Using the POST Profile

With the SAML POST profile, the identity provider sends the full assertion to the service provider over HTTPS. While testing, you may wish to configure Oracle Identity Federation without using a proxy.

> **Note:** The assertion can be sent over HTTP as well. However, it is highly recommend that you always use HTTPS in production environments to ensure the security of the interaction.

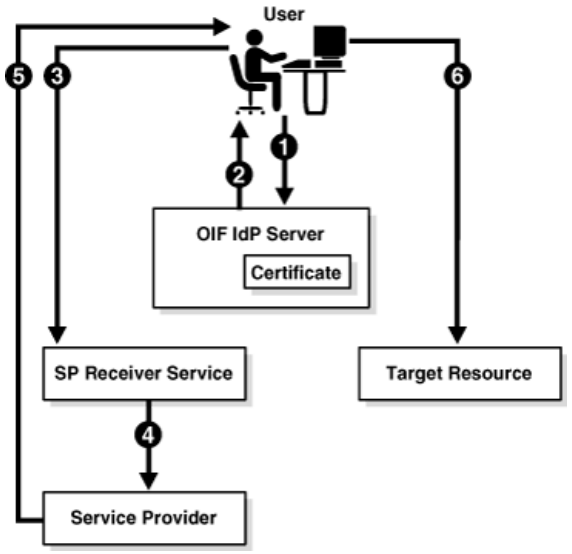
Here are some items to keep in mind when considering the POST profile:

- The POST profile does not require putting your IdP's SAML components in a DMZ.
- The SAML components can be placed behind a firewall.
- The POST profile requires the use of XML signatures, and signing and verifying signatures is resource-intensive.
- If you plan to send or receive large numbers of requests and responses, consider "Sizing Guidelines" on page 2-22 for performance tips.

POST Profile Request Processing

Figure 2–5 shows the process by which requests are processed under the POST profile:

Figure 2–5 POST Profile Processing



The processing flow is as follows:

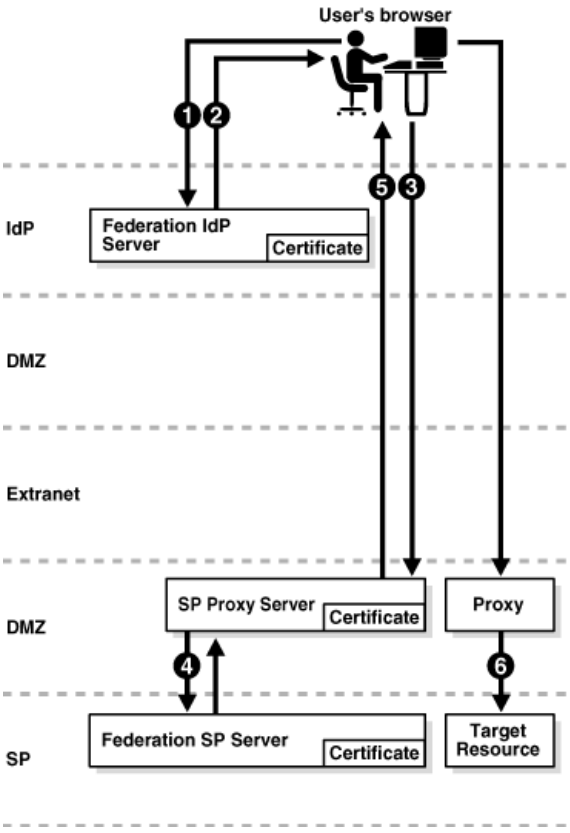
1. The user initiates a request, and must be authenticated before the request can be processed.
2. Oracle Identity Federation - acting as the identity provider - authenticates the user and returns an HTML form containing a response, which consists of an identity assertion and the URL of the service provider. The response is signed using the Oracle Identity Federation IdP's private signing key.

3. The browser posts this form to the URL of the service provider's receiver service. The receiver service verifies the signed response using the IdP's public certificate associated with its signing key.
4. The service provider extracts the assertion, and creates a user session for the assertion.
5. The service provider sends the user's browser a redirect to the requested resource.
6. The user's browser sends a request to the target resource over the user session created by the service provider.

POST Profile With Proxy

In a secure deployment, the POST profile sends the full assertion to the service provider over SSL. The IdP and SP are configured to communicate over HTTPS through their SSL ports. Figure 2–6 illustrates this preferred approach for using the POST profile in production, with Oracle Identity Federation serving as the SP in the DMZ:

Figure 2–6 POST Profile with a Proxy



See Also: For more information about setting up a proxy server for Oracle Identity Federation, see "Setting Up a Proxy for Oracle Identity Federation" on page 9-20

The processing flow is as follows:

1. With Oracle Identity Federation acting as the IdP, the user requests a resource. The SP, an Oracle Identity Federation server, is accessed through a proxy server located in the DMZ.

2. The IdP server authenticates the user and responds with an HTML form that contains an assertion and the URL of the target resource.

   The response is signed using the Oracle Identity Federation IdP's private signing key.

3. The user's browser posts the form to the SP's proxy receiver service URL.
4. The proxy forwards the form to the SP's receiver service.
5. The Oracle Identity Federation SP verifies the signature, extracts the assertion, creates a user session for the assertion, and sends the user's browser a redirect to the resource.
6. The browser conveys the request to the target resource over the new user session. The request may be handled by an additional proxy located in the service provider DMZ.

SAML Security Considerations

SAML provides numerous security features that you can use to ensure privacy, integrity, authenticity, and confidentiality of the SAML messages and the message exchanges.

This section provides a brief summary of message security considerations. For a detailed analysis of the security risks and countermeasures, refer to the OASIS SAML Security Considerations specification, titled *Security and Privacy Considerations for OASIS SAML V2.0*, at:

`http://docs.oasis-open.org/security/saml/v2.0/saml-sec-consider-2.0-os.pdf`

Oracle Identity Federation supports the full set of security technologies and techniques available for use in a SAML deployment. These include

- SSL/TLS for peer authentication and secure communications
- XML-SIG for message-level integrity and authentication
- XML-ENC for message-level confidentiality

Oracle recommends that secure SSL/TLS channels be used for all SAML message flows in addition to message level security. All communications between an Identity Provider and Service Provider must use bilateral authentication (client and server certificates).

SAML profiles provide specific recommendations on how to securely use SAML assertions and request-response messages in communications protocols. Here are the security requirements for the SAML SSO Artifact and POST profiles.

Secure communication using the SAML Artifact Profile

Secure communication using the SAML artifact profile requires the following:

1. SSL is required when the IdP communicates with an SP, for redirection from the IdP to the user's browser, and for redirection from the browser to the SP.
2. The SP's requester service requires a certificate. For SAML 1.0/1.1, the SP requestor may use HTTP basic authentication with a username and password known to the IdP.

Secure communication using the SAML POST Profile

Secure communication using the SAML POST profile requires the following:

1. Secure HTTP (HTTPS) is required to transmit a user request from a browser to the service provider.
2. The identity provider must use an XML signature to sign responses it sends to a service provider.
3. The service provider must verify the XML signature on the response.

Using the SAML Attribute Sharing Profile

The SAML attribute sharing profile is used by service providers to authenticate users by means of SSL client X.509 certificates rather than SAML assertions, when additional user attributes are needed to provide authorization of resource requests.

Oracle Identity Federation provides the attribute sharing profile for use with Oracle Access Manager to enable interoperation with SAML 2.0 implementations at peer sites. For details about components and their respective roles, and how to configure Oracle Identity Federation and Oracle Access Manager, see "Configuring Attribute Sharing" on page 6-98.

Using the WS-Federation Logout Profile

WS-Federation can be used to sign into one or more service providers using an identity provider that performs the actual authentication.

To log out, the user clicks on a link at the IdP site that initiates a WS-Federation signout. Using a session cookie, Oracle Identity Federation has kept track of each SP to which the user signed on. The server returns an HTML signout page to the user's browser. Each SP processes the signout cleanup to sign out the session created for Oracle Identity Federation.

Authentication Engines

Many Oracle Identity Federation features require the user to be authenticated. Such operations include:

- IdP protocol operations such as single sign-on, federation creation, federation termination, and NameID registration
- SP protocol operations such as federation creation, federation termination, and NameID registration

To gain a perspective on how authentication is effected, we can think of the federation server as comprising these distinct modules:

1. Oracle Identity Federation provides support for Liberty 1.x, SAML 1.0/1.1, and SAML 2.0 protocols.
2. An authentication module provides support for user authentication and integration with IdM solutions.

In addition, Oracle Access Manager provides a range of identity administration functions including Web single sign-on, user self-service and registration, policy management, and delegated administration.

In this section we look at the authentication flows these modules enable in different configurations:

- Authentication Methods in Oracle Identity Federation
- Authenticating with a Repository in IdP Mode
- Authenticating with an IdM Solution in IdP Mode

- Authenticating with Oracle Access Manager or CA eTrust SiteMinder in SP Mode
- Authenticating with OracleAS Single Sign-On in SP Mode
- HTTP Basic Authentication

Authentication Methods in Oracle Identity Federation

The Oracle Identity Federation authentication module can perform two distinct roles in user authentication:

- The authentication module acts as a local authentication mechanism. In this mode, the authentication module can authenticate locally with available authentication systems.

  Oracle Identity Federation conveys authentication requests to the authentication module. Depending on the deployment, the authentication module may interact directly with RDBMS or LDAP repositories, or it may delegate authentication to an IdM solution such as Oracle Application Server Single Sign-On.

- The Oracle Identity Federation authentication module acts to propagate the authentication state. In this mode, Oracle Identity Federation, as a service provider, uses federation protocols to have the user authenticated at a peer identity provider. Oracle Identity Federation then forwards the user to the authentication module, which propagates and creates an authenticated user session in the deployed IdM solution at the SP. In turn, this enables access to the requested protected resource.

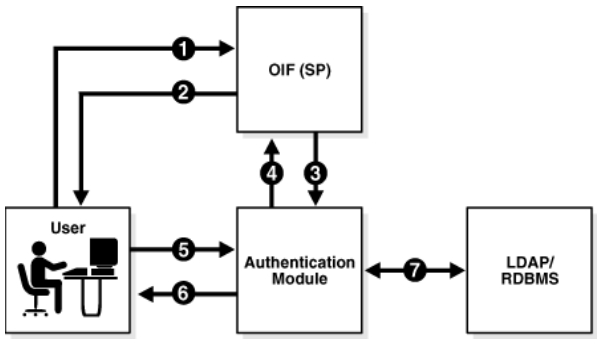
Note: SP mode requires the presence of an IdM solution, while local authentication IdP mode does not require such a solution.

Authenticating with a Repository in IdP Mode

In this deployment, the authentication module interacts directly with a number of repositories and IdM solutions to enable Oracle Identity Federation to authenticate in IdP mode:

- an RDBMS repository
- an LDAP repository
- Oracle Access Manager
- CA eTrust SiteMinder

Figure 2–7 Authenticating with a Repository in IdP Mode



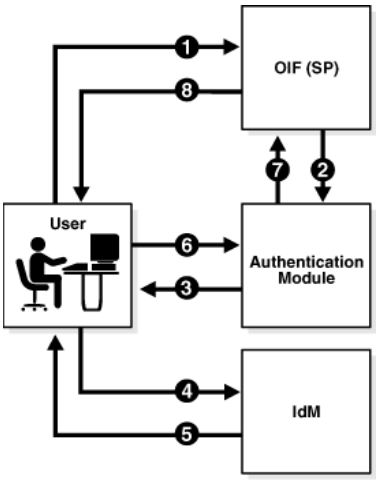
The flow for a local authentication involving such a deployment is as follows:

- The user accesses Oracle Identity Federation (Step 1).
- Oracle Identity Federation forwards the user to the authentication module for local authentication (Step 3).
- The authentication module prompts the user for credentials (Step 6).
- The user enters credentials (Step 5).
- The authentication module interacts with the repository to authenticate the user (Step 7).
- The authentication module forwards the user to Oracle Identity Federation with the user's identification (Steps 6,1).
- Oracle Identity Federation communicates with the authenticated user (Step 2).

Authenticating with an IdM Solution in IdP Mode

In this deployment, the authentication module delegates authentication to the OracleAS Single Sign-On IdM solution to enable Oracle Identity Federation to authenticate in IdP mode.

Figure 2–8 Authenticating with an IdM Solution in IdP Mode



The flow for a local authentication involving an IdM deployment is as follows:

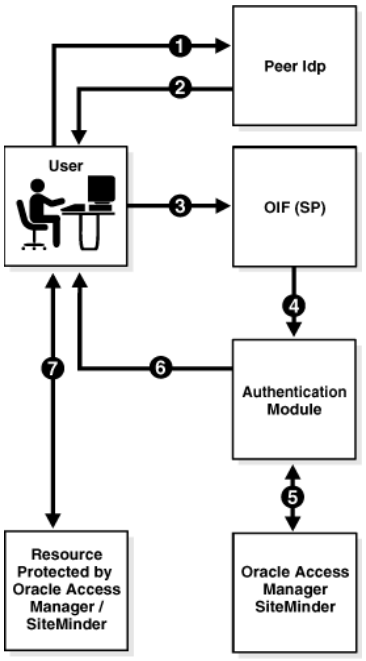
- The user accesses Oracle Identity Federation (Step 1).
- Oracle Identity Federation forwards the user to the authentication module for local authentication (Step 2).
- The authentication module redirects the user to the IdM server for authentication (Steps 3,4).
- The IdM server authenticates the user and redirects the user back to the authentication module (Steps 5,6).
- The authentication module forwards the user to Oracle Identity Federation with the user's identification (Step 7).
- Oracle Identity Federation communicates with the authenticated user (Step 8).

Authenticating with Oracle Access Manager or CA eTrust SiteMinder in SP Mode

In this mode, Oracle Identity Federation uses the federation protocols to identify a user, and requests the authentication module to create an authenticated session at Oracle Access Manager or CA eTrust SiteMinder so that the user can access the requested resource, which is protected by WebGate (for an Oracle Access Manager IdM deployment) or CA eTrust SiteMinder Web Agent (for a CA eTrust SiteMinder deployment).

The request originates at a peer IdP, and Oracle Identity Federation authenticates in SP mode.

Figure 2–9 Authenticating with Oracle Access Manager or CA eTrust SiteMinder in SP Mode



The flow for authenticating a user at a peer provider with Oracle Access Manager or CA eTrust SiteMinder is as follows:

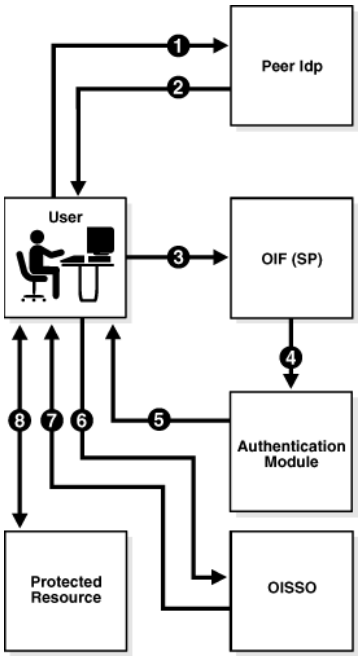
- The user is at the peer IdP (Step 1).
- The IdP redirects the user to Oracle Identity Federation (as SP) with an authentication assertion (Steps 2,3).
- Oracle Identity Federation processes the assertion, creates a local Oracle Identity Federation session, and forwards the user to the authentication module with the identification (Step 4).
- The authentication module interacts with the Oracle Access Manager/CA eTrust SiteMinder server to create an Oracle Access Manager/CA eTrust SiteMinder authenticated session (Step 5).
- The authentication module redirects the user to the protected resource (Step 6).
- WebGate or the CA eTrust SiteMinder Web Agent grant the user access to the protected resource (Step 7).

Authenticating with OracleAS Single Sign-On in SP Mode

In this mode, Oracle Identity Federation uses the federation protocols to identify a user, and requests the authentication module to create an authenticated session at OracleAS Single Sign-On so that the user can access the requested resource, which is protected by `mod_osso`.

The request originates at a peer IdP, and Oracle Identity Federation authenticates in SP mode.

Figure 2–10 Authenticating with OracleAS Single Sign-On in SP Mode



The flow for authenticating a user at a peer provider with OracleAS Single Sign-On is as follows:

- The user is at the peer IdP (Step 1).
- The IdP redirects the user to Oracle Identity Federation (as SP) with an authentication assertion (Steps 2,3).
- Oracle Identity Federation processes the assertion, creates a local Oracle Identity Federation session, and forwards the user to the authentication module with the identification (Step 4).
- The authentication module redirects the user to OracleAS Single Sign-On with the user identification (Steps 5,6).
- OracleAS Single Sign-On creates a local authenticated session and grants access to the resource protected by `mod_osso` (Steps 7,8).

Note: For more information about an environment where Oracle Identity Federation and OracleAS Single Sign-On protect resources and either component can be the authentication mechanism, see "Integrating with Oracle Identity Federation" in Oracle Application Server Single Sign-On Administrator's Guide.

HTTP Basic Authentication

Oracle Identity Federation can be configured to accept HTTP basic credentials without requiring an identity and access management system when using SAML 1.0/1.1 or WS-Federation protocols.

Data Repositories

This section describes installation requirements to enable Oracle Identity Federation to work with data stores. It contains these topics:

- Federation Data Store
- User Data Store
- Transient Data Store

Federation Data Store

You must select a data repository for the persistent federation data store. Oracle Identity Federation works with industry-standard LDAP repositories including Oracle Internet Directory, Sun Java System Directory Server, and Microsoft Active Directory. It also supports a None option (no repository) for SAML 2.0 using non-opaque name identifiers such as e-mail address, X.509 DN, Kerberos, or Windows Name Identifier.

Note: SAML 1.x and WS-Federation do not require a federation data store.

Connection Information

Collect the following information about the repository prior to installing Oracle Identity Federation:

- The Connection URL (space-delimited list of LDAP server URLs - hostname and port)
- The Bind DN

  This is the DN used by the Oracle Identity Federation server to connect to the LDAP server. For example:

  ```
  cn=fedid,dc=mycompany,dc=com
  ```

- Password
- The User Federation Record Context

  This is the node under which all federation records for this Oracle Identity Federation server will be stored.

- The LDAP Container Object Class

  This is the type of User Federation Record Context that Oracle Identity Federation should use when creating the LDAP container, if it does not exist already. If that field is empty, its value will be set to `applicationprocess`. For Microsoft Active Directory this field has to be set, to container for example. The appropriate setting for this field depends on the User Federation Record Context being used. (User Federation Record Context is described later in this section).

  Here are examples of the LDAP Container Object Class for different types of directory servers:

  - Oracle Internet Directory: *empty*
  - Sun Java System Directory Server: *empty*
  - Microsoft Active Directory: `container`
- Unique Federation ID Attribute

  This is the LDAP attribute to be used to uniquely identify a federation record. This attribute should be defined in the LDAP Object Class of the Federation Record type, or in its top parent. If it is empty, the default Federation ID attribute will be used as the DN of the Federation Record.

  Here are examples of the Unique Federation ID attribute for different types of directory servers:

  - Oracle Internet Directory: *empty*
  - Sun Java System Directory Server: *empty*
  - Microsoft Active Directory: *empty*
- Maximum Connections. This is the maximum number of concurrent connections made by Oracle Identity Federation to the LDAP server.
- Connection Wait Timeout. This is the maximum number in seconds to wait until a connection is available, when the maximum number of connections opened by Oracle Identity Federation to the LDAP server has been reached.

Relationship of User Federation Record Context and LDAP Container Object Class

The User Federation Record Context and LDAP Container Object Class need to be compatible. In the User Federation Record Context, the administrator will specify the DN of the container where the federation records will be stored. That DN will contain the parent of the container that must already exist (for example `dc=us,dc=oracle,dc=com`), and an attribute of the Federation Record Context that is part of its object class (for example, `cn=orclfed`). An example of such DN would be `cn=orclfed,dc=us,dc=oracle,dc=com`.

The requirement for that example is that `cn` must be an attribute of the Object Class set in the LDAP Container Object Class field (or the `applicationprocess` object class if not set).

If the administrator chooses to have the DN of the Federation Record Context like `ou=fed,dc=us,dc=oracle,dc=com`, she will need to set the LDAP Container Object Class field to an object class that has `ou` as an attribute, like `organizationalUnit`.

To summarize, the User Federation Record Context references the LDAP container entry under which federation records will be stored, and the LDAP Container's attribute used in the DN must be defined in the LDAP Container Object Class used. For example, if DN is `ou=fed,dc=us,dc=oracle,dc=com`, then the LDAP Container Object Class must define the `ou` attribute; if DN is `cn=fed,dc=us,dc=oracle,dc=com`, then the LDAP Container Object Class must define the `cn` attribute.

A Note About the LDAP Schema

The LDAP Schema needs to be upgraded to include the attributes and object classes defined by Oracle Identity Federation, in order for the Federation server to create records in the LDAP server.

Upgrade the LDAP schema either at installation time (with the Advanced Installation mode), or after installation.

Upgrade Schema at Installation

To perform the upgrade at installation time, take these steps:

1. Choose the Advanced Installation mode.
2. On the "Select Configuration Options" page, check the "Federation Data in LDAP Server" box. This indicates that the federation records will be stored in an LDAP server whose schema must be upgraded.
3. On the "Specify Federation Data Store" page, enter the LDAP connection information. The schema will then be upgraded as part of the installation process.

Post-Installation Schema Upgrade

To perform the upgrade post-installation, note that the Oracle Identity Federation installation includes `LDIF` files that you can execute using the `ldapmodify` tool to upgrade the schema of an LDAP server.

The `LDIF` file to use depends on the type of LDAP server used:

- `$Oracle_Home/fed/setup/ldap/userFedSchemaOid.ldif` if you use Oracle Internet Directory
- `$Oracle_Home/fed/setup/ldap/userFedSchemaIPlanet.ldif` if you use the Sun One Directory Server
- `$Oracle_Home/fed/setup/ldap/userFedSchemaAD.ldif` if you use Microsoft Active Directory Server. In this case, you need to edit the LDIF file to replace the string `%DOMAIN_DN%` with your active directory domain suffix.

  An example suffix is `dc=mydomain,dc=mycompany,dc=com`.
- `$Oracle_Home/fed/setup/ldap/userFedSchemaTivoli.ldif` if you use the IBM Tivoli Directory Server (IBM TDS) 6.0

Using `ldapmodify`, you can upgrade the LDAP schema with the LDIF file. For example:

```
ldapmodify -c -D BIND_DN_USERNAME
   -w PASSWORD
   -f $Oracle_Home/fed/setup/ldap/userFedSchemaOid.ldif
   -h LDAP_HOSTNAME -p LDAP_PORT -x
```

User Data Store

You must select a data repository for the user data store. Oracle Identity Federation works with industry-standard repositories including:

- LDAP (Oracle Internet Directory, Sun Java System Directory Server, and Microsoft Active Directory)
- RDBMS
- Oracle Access Manager
- OracleAS Single Sign-On
- CA eTrust SiteMinder

The role played by the data repository depends on whether Oracle Identity Federation will be configured as an identity provider (IdP) or a service provider (SP):

- As an IdP, Oracle Identity Federation uses the repository to verify user identities and to build protocol assertions.
- As an SP:
  - Oracle Identity Federation uses the repository to map information in received assertions to user identities at the destination, and subsequently to authorize users for access to protected resources.
  - When creating a new federation, Oracle Identity Federation uses the repository to identify the user and link the new federation to that user's account.

Connection Information for LDAP Repositories

Connection information is required for IdM products that use an LDAP directory for their user data, including Oracle Access Manager, OracleAS Single Sign-On, and CA eTrust SiteMinder.

Collect the following information about the repository prior to installing Oracle Identity Federation:

- Connection URL - space delimited list of LDAP URLs
- Bind DN
- Password
- User ID Attribute - the attribute name to use to map users during lookups or authentication procedures

  Here are examples of the User ID Attribute for different types of directory servers:

  - Oracle Internet Directory: `uid`
  - Sun Java System Directory Server: `uid`
  - Microsoft Active Directory: `sAMAccountName`
- User Description Attribute

  This field references the user attribute to use as a human readable federation owner identifier. This information will be stored in the federation record.

  Here are examples of the User Description Attribute for different types of directory servers:

  - Oracle Internet Directory: `uid`
  - Sun Java System Directory Server: `uid`
  - Microsoft Active Directory: `sAMAccountName`
- Person Object Class - the LDAP object class representing a user in the LDAP server

  Here are examples of the Person Object Class for different types of directory servers:

  - Oracle Internet Directory: `inetOrgPerson`
  - Sun Java System Directory Server: `inetOrgPerson`
  - Microsoft Active Directory: `user`
- Base DN - the node under which LDAP user search will be performed. For example:

  ```
  dc=us,dc=oracle,dc=com
  ```

- Maximum Connections - the maximum number of concurrent connections made by Oracle Identity Federation to the LDAP server
- Connection Wait Timeout - the maximum number in seconds to wait until a connection is available, when the maximum number of connections opened by Oracle Identity Federation to the LDAP server has been reached

Connection Information for RDBMS Repositories

Connection information is required for CA eTrust SiteMinder, which uses a database for its user repository.

Collect the following information about the repository prior to installing Oracle Identity Federation:

- JNDI Name - references the data source configured in Oracle Application Server pointing to the RDBMS to use to authenticate/locate users. You will need to define this data source after Oracle Identity Federation installation, prior to authenticating any users.
- User Name
- Password
- Login Table - the RDBMS table containing the user information used for authentication and lookups
- Login ID Column - the RDBMS column in the login table containing the user identifiers
- Login Password Column - the RDBMS column in the login table containing the user passwords
- Password Digest Algorithm - the hashing algorithm to apply to the input password before matching the result against a value stored in the RDBMS table
- User Description Attribute - references the user attribute to use as a human readable federation owner identifier. This information will be stored in the federation record.

Transient Data Store

Oracle Identity Federation also maintains transient data for federation protocol/session state. This data can be stored in either in-memory tables or a relational database.

An RDBMS transient data store is required for high-availability and clustering support. Note that, in addition to transient session data, server configuration data will be persisted to the database for centralized cluster configuration.

See Also: For more information, see:

- "Advanced Installation Procedure" on page 3-7
- "Configuration Assistant Operations" on page 9-2

Installation Requirements

This section explains installation requirements, including:

- Required Components
- Supported platforms

Required Components

Oracle Identity Federation requires the following components:

- Java 2 SDK, Standard Edition (J2SE), Version 1.4.2 (bundled with the installation)
- Oracle Containers for J2EE - OC4J (bundled with the installation)
- A user identity data store. This is typically an LDAP directory, but can optionally be a database store.
- One of these repositories for the user federation data store:
  - Oracle Internet Directory
  - Microsoft Active Directory
  - Sun Java System Directory Server

> **Note:** A user federation data store is not absolutely required for Oracle Identity Federation in all cases: it is required for Liberty 1.x and SAML 2.0 opaque persistent identifiers, but is optional for SAML 1.x, WS-Federation, and SAML 2.0 non-opaque identifiers (such as email address, subject DN, and so on)

- Oracle9*i* Database Server version 10.1.0.3.0 or higher for the RDBMS transient data store
- Oracle HTTP Server for proxy implementation; this is the only proxy server supported by Oracle Identity Federation, and is bundled with the installation.

Supported platforms

Refer to the Oracle Application Server/OC4J certification matrix for information about platforms and components supported by Oracle Identity Federation.

> **See Also:** The Oracle Technology Network product page at `http://www.oracle.com/technology/products/index.html` .

Sizing Guidelines

When planning to deploy a federated identity system that leverages Oracle Identity Federation, it is critical to understand the performance considerations, choices, and trade-offs involved in the architecture.

This section considers various factors that have an impact on performance in a federated environment, and provides some guidelines to help you assess hardware requirements for a production system with a standalone Oracle Identity Federation server. The following topics are included:

- Deployment and Architecture Considerations
- Typical Deployment Scenario
- Reference Server Footprint
- Topology
- Performance Figures

Deployment and Architecture Considerations

Before deploying Oracle Identity Federation, you need to define the architecture and role that Oracle Identity Federation will play in a federated authentication setting. Here are some decisions that you will need to make:

- Which federation specifications will be used with various trusted partners? Choices include:
  - SAML 2.0. With additional flows in comparison to SAML 1.x, performance considerations may play a greater role.
  - Liberty ID-FF 1.1 and 1.2
  - SAML 1.0 and 1.1
  - WS-Federation
- What profiles will you use to federate with your partners? Options include Browser POST or Artifact profile, WS-Federation Passive Requestor profile, attribute sharing, and others.
- Which transport security protocols and certificates will be used? Will the assertions be signed?
- What roles will Oracle Identity Federation be playing? Options are:
  - Identity Provider (IdP), also referred to as a source domain
  - Service Provider (SP), also referred to as a destination domain
  - Both IdP and SP
- What type (and what vendor's) authoritative identity repositories will be installed?
- If deploying Oracle Identity Federation as a Service Provider (SP), consider whether you will deploy an identity and access management (IAM) system to authorize access to protected resources. The overall performance of the solution will be highly dependent on the performance of the IAM system.

  Note: Oracle Identity Federation provides an integration framework that enables you to create lightweight federation endpoints without requiring an access management system.

- Will you install a proxy server with Oracle Identity Federation? If so, take into account where the Oracle Identity Federation and proxy servers will reside - for example, in the DMZ or behind a firewall.
- How will the architecture provide high availability scenarios? Specifically:
  - Whether you want to support cold failover clusters leveraging the Oracle Application Server High Availability topologies
  - Whether you want to set up a common assertion store database to make assertion data available to more than one federation server in a load-balancing and failover configuration

The overall throughput and performance of Oracle Identity Federation can depend on a number of factors, such as:

- Which profiles are supported (for example, Artifact or POST)

- Security features in use (using certificates, digitally signing and/or encrypting assertions)
- Use of individual components involved in processing a transaction, such as fire walls, proxy servers, LDAP directories, databases, and IAM systems

The subsequent subsections provide more detail on these topics.

Profiles

The SAML specification supports a number of profiles, with the two primary deployed profiles being the SAML Browser POST and Artifact profiles. In general, using the SAML Browser POST profile is more performance-friendly than the Artifact profile, as the POST profile requires fewer round trips between the IdP and SP. However, there is a potential security trade-off given that the Artifact profile is, in general, a more secure method of exchanging SAML assertions.

Repositories

When working with LDAP directories, RDBMS, and back end IAM systems, it is important to pay attention to the transaction processing speed of the component in question, since this can affect the performance of your production environment. Note that:

- RDBMS parameters can be tuned to provide options to control database connection pool settings.
- If using Oracle Access Manager as the backed identity and access management system, the AccessGate performance considerations apply, as do the Access Server sizing considerations outlined here:

  `http://www.oracle.com/pls/wocprod/docs/page/ocom/technology/products/id_mgmt/pdf/wp-oracle-idm-sizing-considerations.pdf`

Transient Storage

Place the transient data store in memory for improved performance. (See "Performance Figures" on page 2-27 for an example).

Security for Assertions

Performance can be sensitive to the presence or absence of digital signatures/encryption in the SAML 2.0 assertions. While removing these features can improve performance, it is not recommend if the IdP and SP communication takes place over the internet.

Connection Tuning

Pay attention to the proper adjustment of the maximum number of concurrent connections to:

- LDAP servers,
- the RDBMS (for transient session data and configuration), and
- remote providers (when the Oracle Identity Federation servers interact directly with each other using the SOAP protocol)

Also review the settings in the **IdM Data Stores** -> **User Data Store** and **Federation Data Store**.

Check "Managing Oracle Identity Federation Performance" on page 9-14.

High Availability

For greater performance and high availability, consider scaling and load-balancing multiple Oracle Identity Federation servers. Implementing a load-balancing solution provides backup and failover protection for your site.

For details, see "High Availability" on page 9-16 and "Setting Up a Load Balancer with Oracle Identity Federation" on page 9-18.

Tuning Servers

Take into account the presence of other servers in your production environment. Specifically, consider:

- Tuning Oracle Application Server and setting appropriate connection limits for Oracle Identity Federation. You can:
  - Tune Oracle Application Server using typical configuration parameters such as memory used, number of processes, and so on. For details, see the *Oracle Application Server Performance Guide*.
  - Specify the maximum number of HTTP/JDBC connections that Oracle Identity Federation uses when communicating with remote HTTP servers and RDBMS servers. For details, see "Setting Concurrent Connection Limits" on page 9-14 and "Setting JDBC Connection Limits" on page 9-14.
- Tuning the Oracle HTTP Server, which is leveraged by Oracle Identity Federation. See "Tuning Oracle HTTP Server" on page 9-15 for more information.

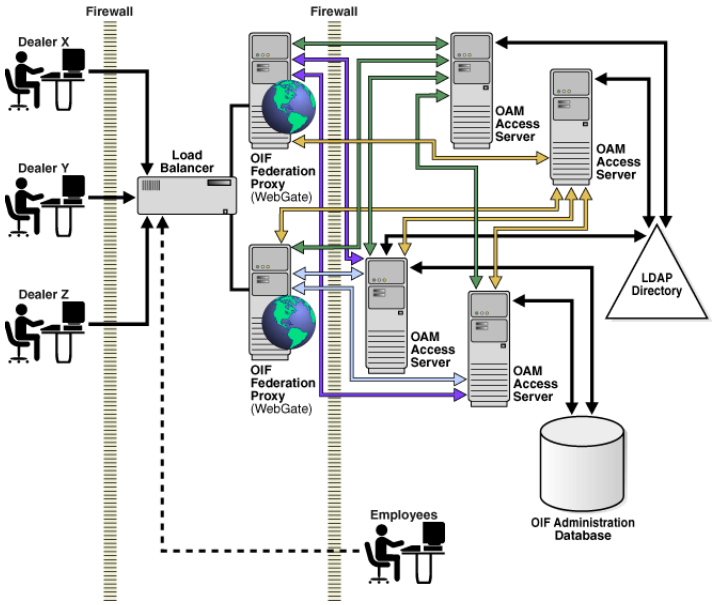
Impact of Additional Security

Introducing additional security measures, such as fire walls, proxy servers, or using SSL authentication, can add extra steps in federated transactions and therefore impact performance.

Typical Deployment Scenario

Figure 2–11 illustrates a typical Oracle Identity Federation deployment architecture for a Service Provider, where Oracle Identity Federation relies on Oracle Access Manager as the back end access management system. The diagram illustrates multiple partners coming in through the DMZ and accessing a load-balanced pair of Oracle Identity Federation Proxy Servers, which are front-ending a pair of Oracle Identity Federation servers.

Figure 2–11 A Typical Federation Deployment Architecture



Reference Server Footprint

The following hardware and equipment is recommended for a baseline Oracle Identity Federation deployment, for an environment supporting up to 2000 concurrent users:

- Any supported server-class machine and operating system for Oracle Identity Federation. The list of certified platforms for Oracle Identity Federation can be found on the Oracle Application Server 10gR3 (10.1.4.0.1) certification matrix at:

  `http://www.oracle.com/technology/software/products/ias/files/idm_certification_101401.html`

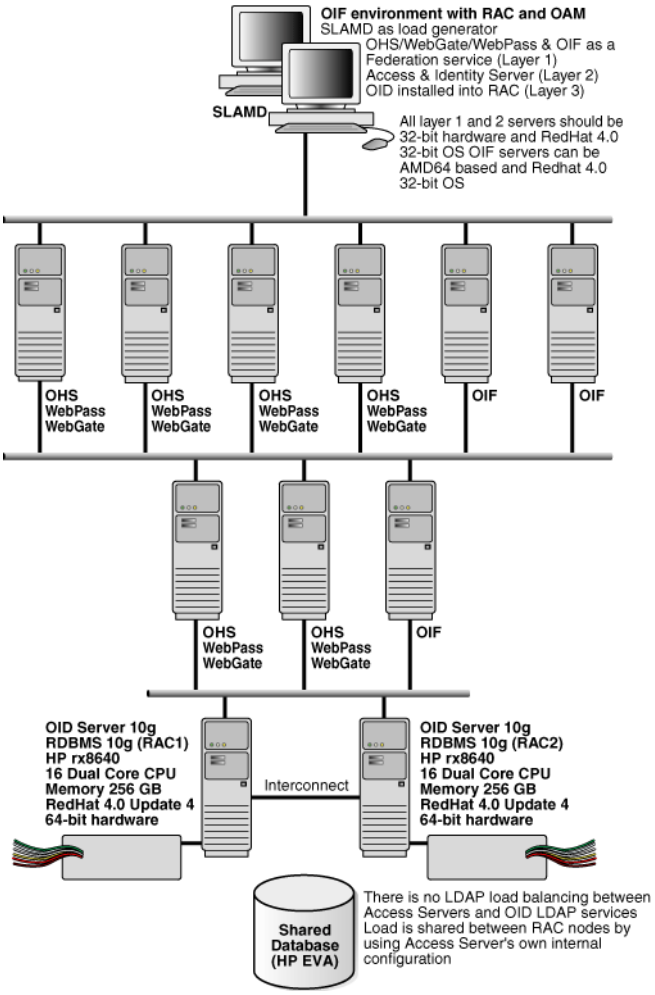
  Failover scenarios would double the number of machines required. Use a minimum of two Oracle Identity Federation servers, on separate machines, for redundancy.

- Server footprint:
  - 2-4 GB memory (4GB recommended)
  - Minimum of 2 CPUs per machine
- If a proxy server is being used, follow the vendor-specific sizing recommendations.

Topology

Figure 2–12 shows the recommended topology for an Oracle Identity Federation deployment in SP mode in which Oracle HTTP Server serves up a provider application that is protected by a webgate.

Figure 2–12 Sample Topology for Oracle Identity Federation



Performance Figures

Using the reference server footprint and topology recommendations as described above, the following performance results have been observed:

Test 1

```
Transient Store: RDBMS
CPUs: 4
Directory: Oracle Internet Directory


7 OC4J_FED instances both SP and IdP
   -Xmx2048 -Dfed.jdbc.min.conn=50 -Dfed.jdbc.min.conn=300
SP OIF -> ko40g OID
IdP OIF -> ko30g OID
SP and IdP User data store ldap connection pool = 300
SP and IdP Federation data store = None (SAML)
SP and IdP OIF transient data in rdbms
SP and IdP HTTP Servers:
   MinSpareServers 50
   MaxSpareServers 250
   StartServers 250
```

```
   MaxClients 500
AccessGate:
   MinSpareServers 5
   MaxSpareServers 10
   StartServers 100
   MaxClients 100
AccessSvr1 -> ko30g OID
AccessSvr2 -> ko40g OID
OID1 and OID2 on port 389:
```

Result: 125 TPS

Test 2

```
Transient Store: Memory
CPUs: 4
Directory: Oracle Internet Directory


5 OC4J_FED instances both SP and IdP
   -Xmx2048 -Dfed.jdbc.min.conn=50 -Dfed.jdbc.min.conn=300
SP OIF -> ko40g OID
IdP OIF -> ko30g OID
SP and IdP User data store ldap connection pool = 300
SP and IdP Federation data store = None (SAML)
SP and IdP OIF transient data in memory
SP and IdP HTTP Server:
   MinSpareServers 50
   MaxSpareServers 250
   StartServers 250
   MaxClients 500
AccessGate:
   MinSpareServers 5
   MaxSpareServers 10
   StartServers 100
   MaxClients 100
AccessSvr1 -> ko30g OID
AccessSvr2 -> ko40g OID
OID1 and OID2 on port 389:
   orclserverprocs 4
   orclmaxcc 4
```

Result: 240 TPS

> **Note:** Switching from RDBMS to an in-memory transient store provided a substantial performance gain in the above tests.

Implementation Checklist

The following checklist summarizes the key items for planning an Oracle Identity Federation installation and provides the essential starting point for deployment.

Table 2–3 Implementation Checklist

Planning Item	Recommended / Proposed Value	Notes
Architecture/Basic Configuration		

Table 2–3 (Cont.) Implementation Checklist

Planning Item	Recommended / Proposed Value	Notes
role played		IdP, SP, or both
topology		hub-and-spoke or peer-to-peer
protocol		Liberty 1.1, Liberty 1.2, SAML 2.0, or any combination of the three. SAML 1.0, SAML 1.1, and WS-Federation.
Repository		Specify repository for the user data and federation persistent data.
LDAP server hostname		for example, `ldap.mydomain.com`
LDAP server port number		for example, `389`
LDAP server access credentials		for example, `Bind DN = {cn=orcladmin}, Password = {mysecret}`
Base DN		for example, `dc=mydomain,dc=com`
federation record context		for example, `cn=fed,dc=mydomain,dc=com`
federation schema update[1]		This information must be provided at the time of installation.
transient data store		Specify repository for transient data: RDBMS or in-memory.
IdP Profiles & Bindings		Use a row for each combination enabled.
SP Profiles & Bindings		Use a row for each combination enabled.
SSL Encryption		
Enabled/Disabled		
Java keystore		Specify Java keystore location if using SSL.
		For information about setting up SSL, see "Using SSL with Oracle Identity Federation" on page 6-133.

Table 2–3 (Cont.) Implementation Checklist

Planning Item	Recommended / Proposed Value	Notes
Certificates		
signing		Specify location of PKCS #12 wallet for signing key pair.
encryption		Specify location of PKCS #12 wallet for encryption key pair.
Performance Planning		
Topology, Reference Server Footprint		For performance tips and recommendations, see "Sizing Guidelines" on page 2-22

[1] For the federation schema update, collect the Connection URL, the Bind DN, password, User Federation Record Context, the LDAP Container Object Class (Microsoft Active Directory), and Unique Federation ID Attribute.

3

Installing Oracle Identity Federation

This chapter details the steps required to install Oracle Identity Federation. As we shall see, there are two installation modes: a basic mode which requires little input and a simpler installation, and an advanced mode which provides more flexibility.

The chapter contains these sections:

Prerequisites

This discussion assumes that you have an understanding of Oracle Identity Federation concepts and features, and have collected the information necessary for installation.

> **See Also:** Chapter 2, "Planning Oracle Identity Federation Deployment" for a checklist of information necessary for deployment.

Overview of Installation Steps

This section explains briefly the steps involved in Oracle Identity Federation installation.

> **Note:** There are two installation modes, **Basic** and **Advanced**. Table 3–1 covers both modes, and each mode is subsequently discussed in its own section.

Table 3–1 Oracle Identity Federation Installation Steps

#	Step	Description
1	Welcome screen	
2	Step for Unix platforms	Run `OrainstRoot.sh`.
3	File locations	Supply source and destination files, paths.
4	Product selection	Choose the product to install.

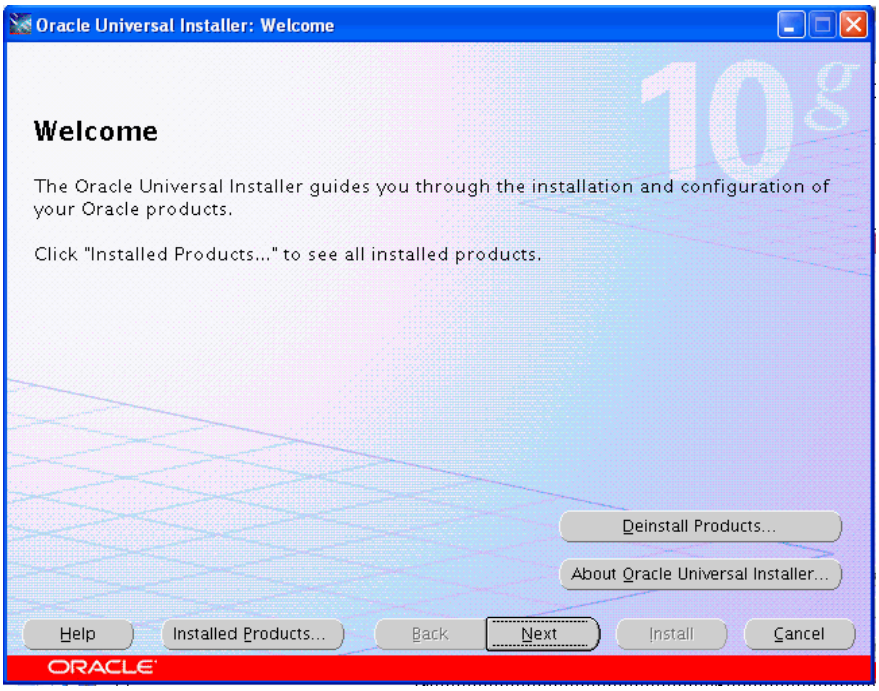
Table 3–1 (Cont.) Oracle Identity Federation Installation Steps

#	Step	Description
5	Type of install	Choose between default and advanced options. If you select the default option, you are directed to Step 11.
6	Pre-install checklist	A screen displays pre-installation requirements for confirmation.
7	Port configuration	Choose between manual and automatic configuration.
8	Virtual host	Select virtual addressing option.
9	Record store	Decide how the record store should be updated.
10	Transient session store	Specify where transient session data will be stored.
11	Server instance creation	Specify a server name and administrator password.
12	Summary screen	Displays install options, settings and requirements.
13	Progress	
14	Run root.sh	This step applies only to Unix/Linux platforms.
15	Post-installation	Run the Configuration Assistant to deploy Oracle Identity Federation.

Basic Installation Procedure

Take the following steps to install Oracle Identity Federation:

1. Run the Oracle Universal Installer. The welcome screen appears.

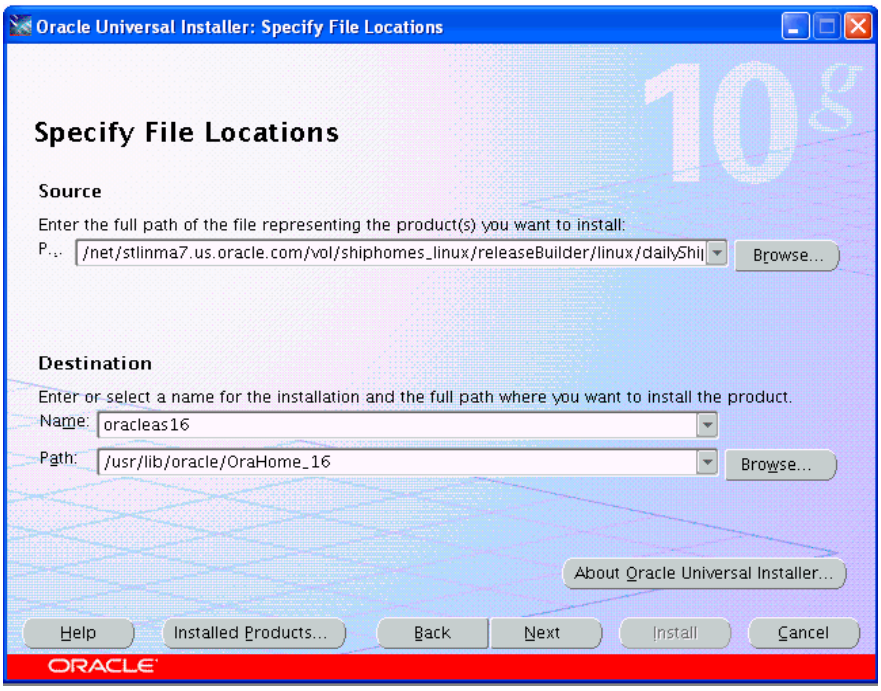


   No input is required on this screen. Click **Next** to continue.

2. If you are installing on a Unix platform, and this is the first install, you must:
   - specify the inventory directory

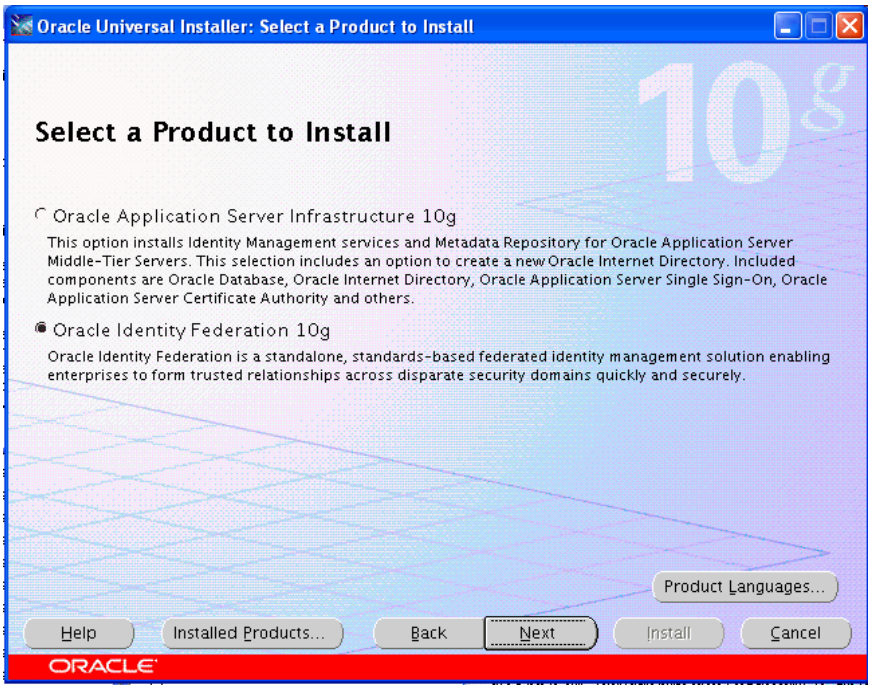
- run the `OrainstRoot.sh` shell script

3. Specify the path and filename for the install file, a name for the installation, and the complete path to the location where you want to install.

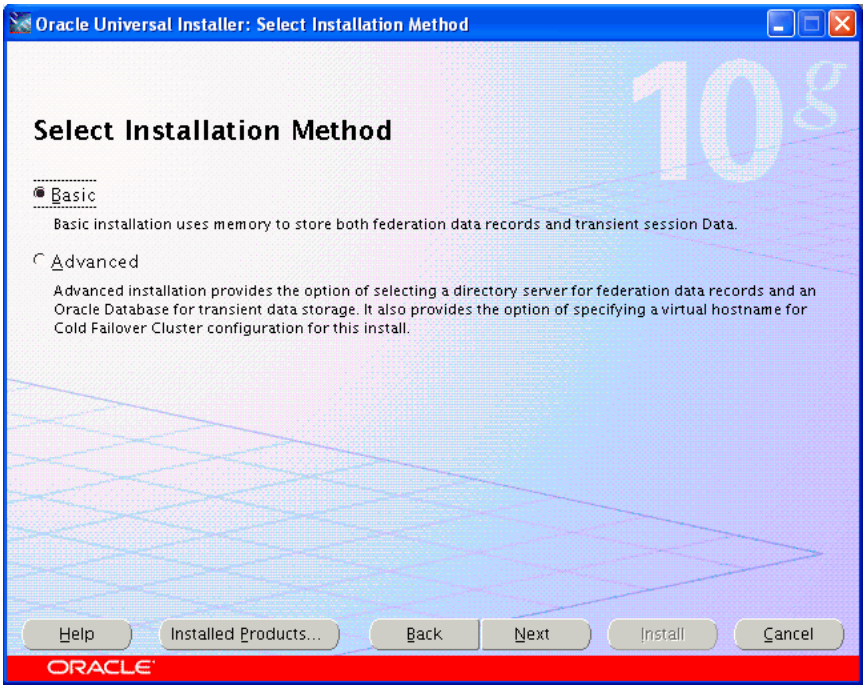


> **Note:** The source file path shown in this screen is for illustration purposes only. The actual path you see will depend on your installation source file.

4. Select Oracle Identity Federation as the product to install.



5. Select the **Basic** installation method.



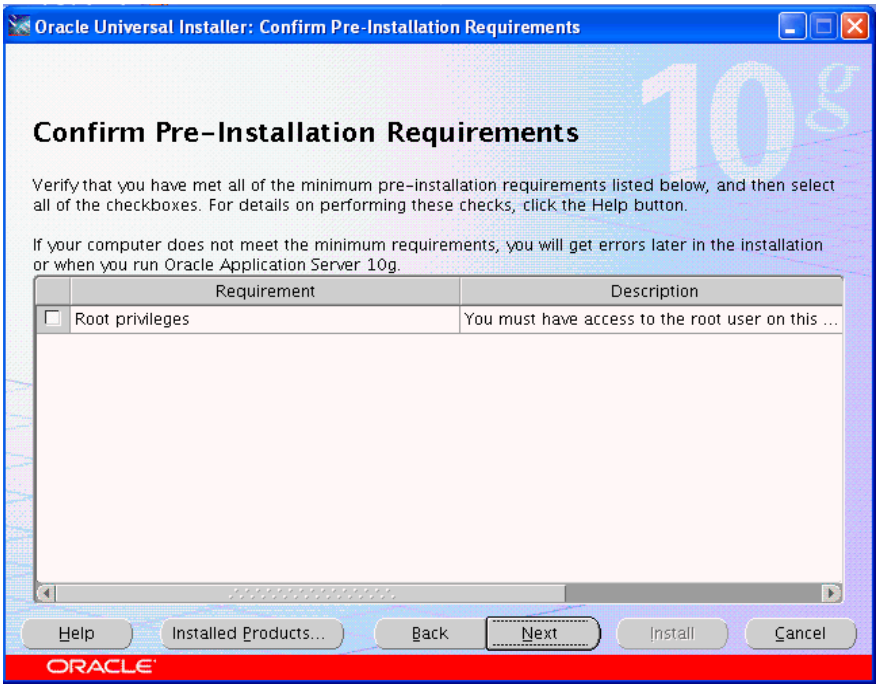
When you choose the basic installation, Oracle Universal Installer makes the following assumptions:

- pre-installation requirements such as root privileges for the host have been met
- ports used by components and services will be configured automatically, using a pre-allotted port range for each component

Note: You can find port information post-install by checking the `$ORACLE_HOME/staticports.ini` file.

- virtual addressing is not required
- your LDAP directory server will not be automatically updated with the federation record schema
- no federation data store information will be collected

6. Confirm pre-installation requirements have been met by checking the box(es).

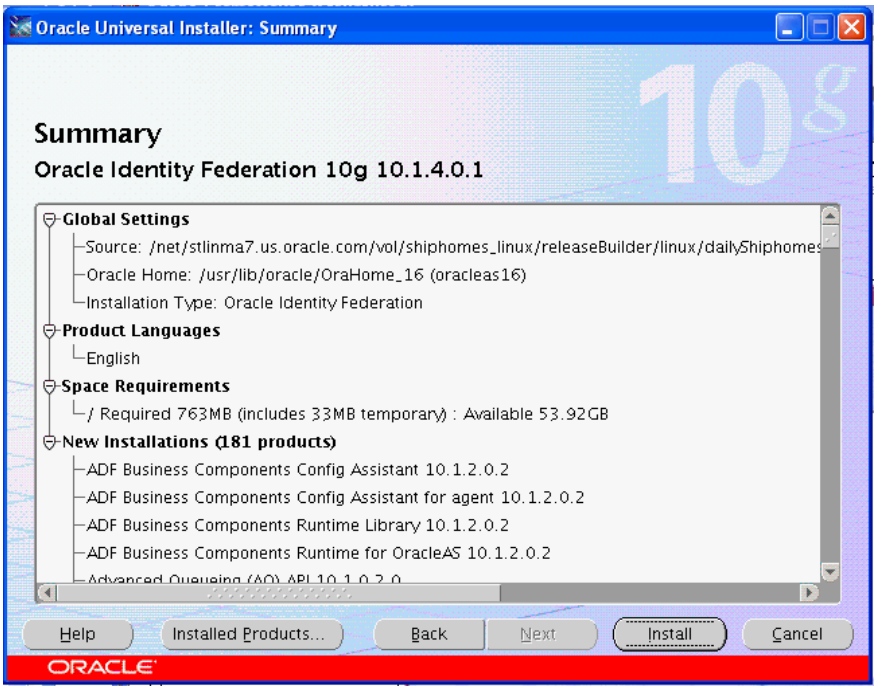


7. Specify Oracle Application Server hostnames and the administrator password for this instance of Oracle Identity Federation.

Note: The Oracle Identity Federation administrator username is `oif_admin`.

Note: This step sets both the `ias_admin` password and the `oif_admin` password. The password field cannot be left blank.

8. Review the summary screen. To revise any information, press the **Back** button. To continue with the installation, press **Install**.



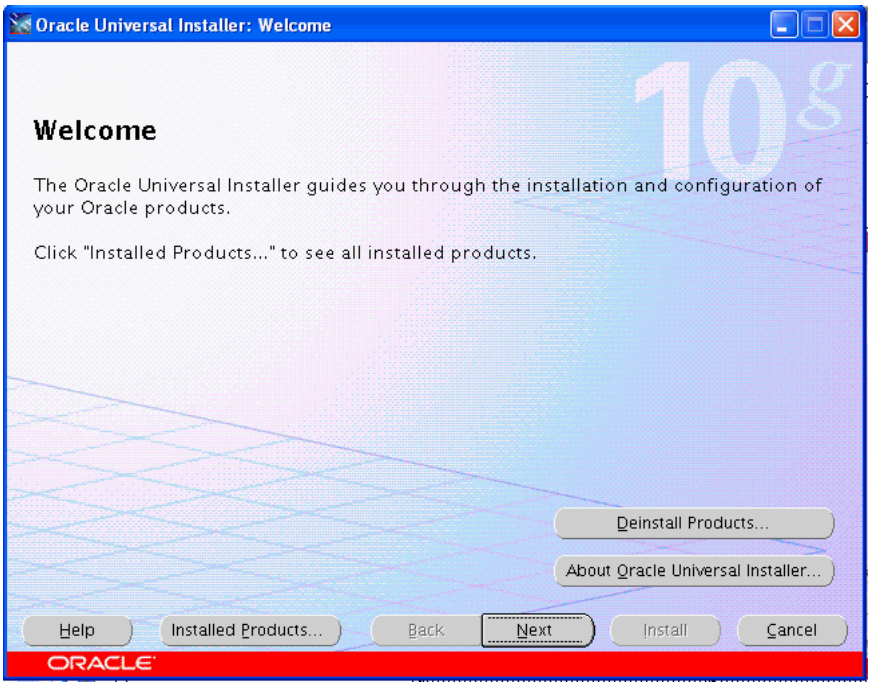
9. Oracle Universal Installer creates an instance of Oracle Containers for J2EE (OC4J) and Oracle Identity Federation.
10. The installer next directs you to the configuration assistant for default settings.
11. The Configuration Assistant configures and deploys the EAR file and modifies configuration files. After configuration is complete, a configuration summary screen appears.
12. The Oracle Universal Installer wizard prompts you to exit the session.

Advanced Installation Procedure

The advanced installation procedure contains several steps that are bypassed in the basic procedure. See Table 3–1 for a description of all the steps.

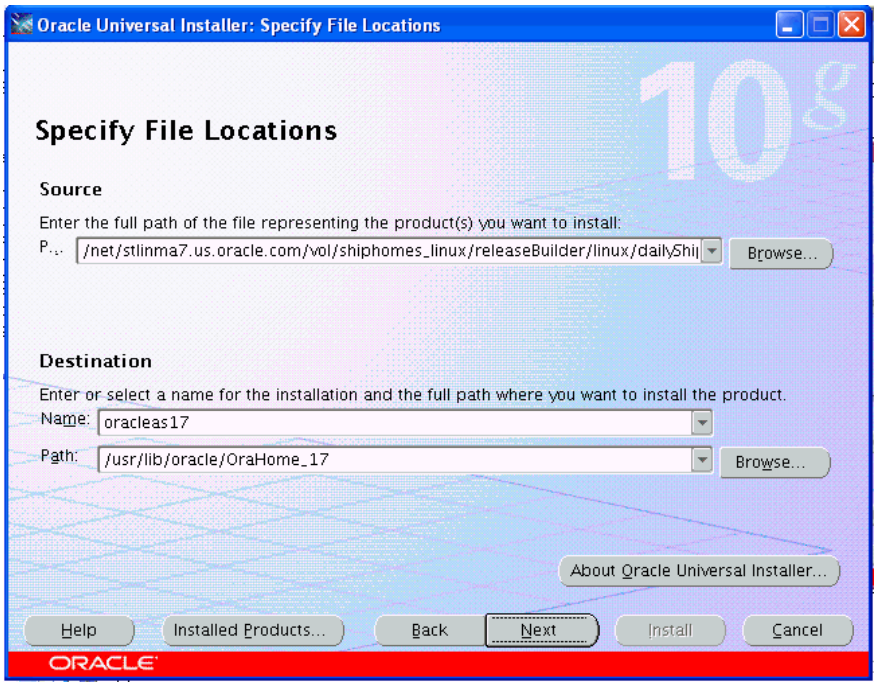
Take the following steps to install Oracle Identity Federation in the advanced mode:

1. Run the Oracle Universal Installer. The welcome screen appears.



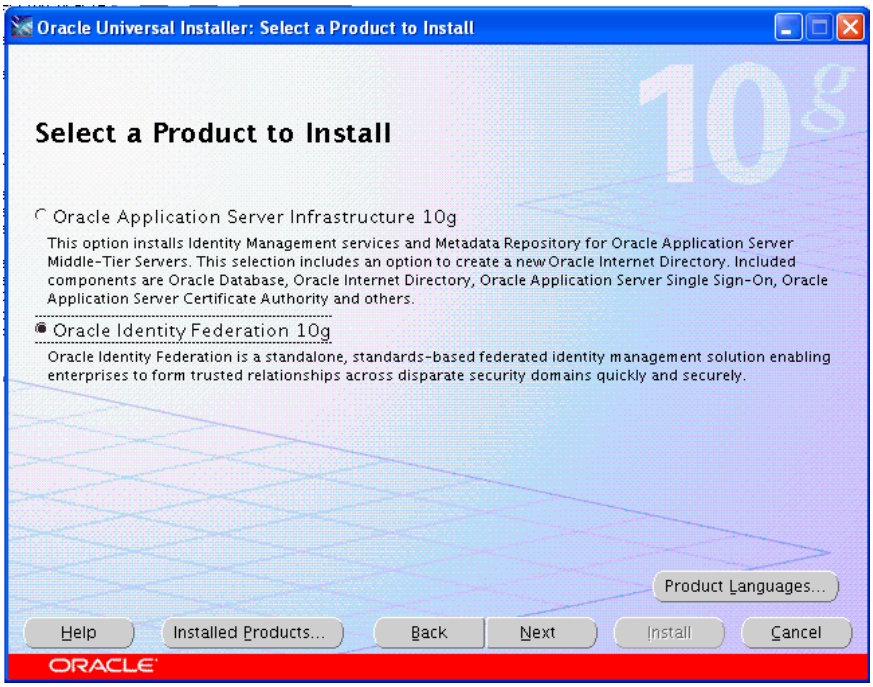
No input is required on this screen. Click **Next** to continue.

2. If you are installing on a Unix platform, and this is the first install, you must:
   - specify the inventory directory
   - run the `OraInstRoot.sh` shell script
3. Specify the path and filename for the install file, a name for the installation, and the complete path to the location where you want to install.

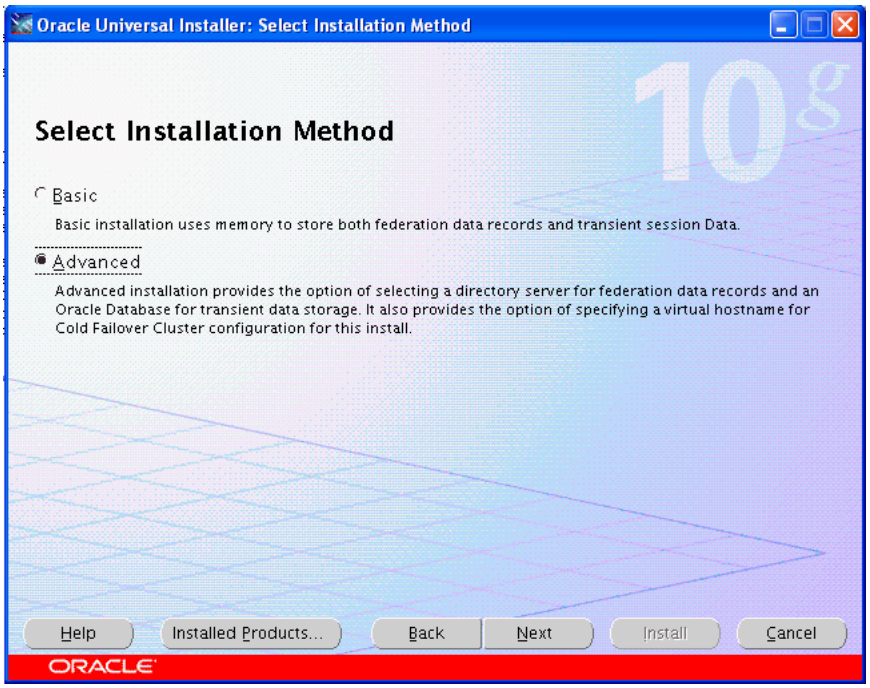


Note: The source file path shown in this screen is for illustration purposes only. The actual path you see will depend on your installation source file.

4. Select Oracle Identity Federation as the product to install.



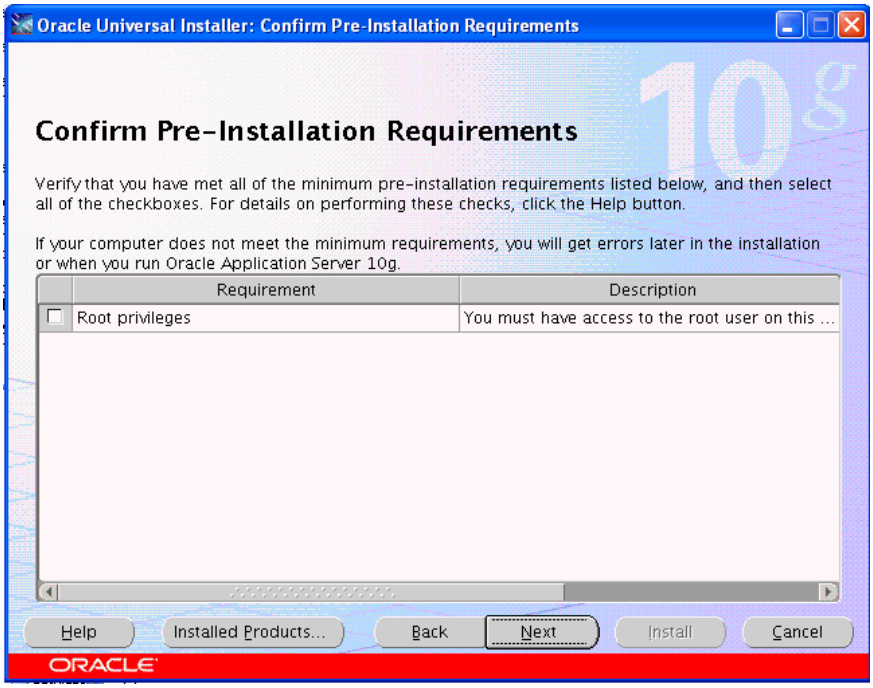
5. Select the **Advanced** installation method.



When you select the **Advanced** option, the installer continues with Step 6 to collect this information:

- confirmation of pre-installation requirements such as root privileges for the host
- port configurations
- virtual addressing
- LDAP directory server information for the federation record schema
- federation data store information

6. Confirm pre-installation requirements have been met by checking the box(es).

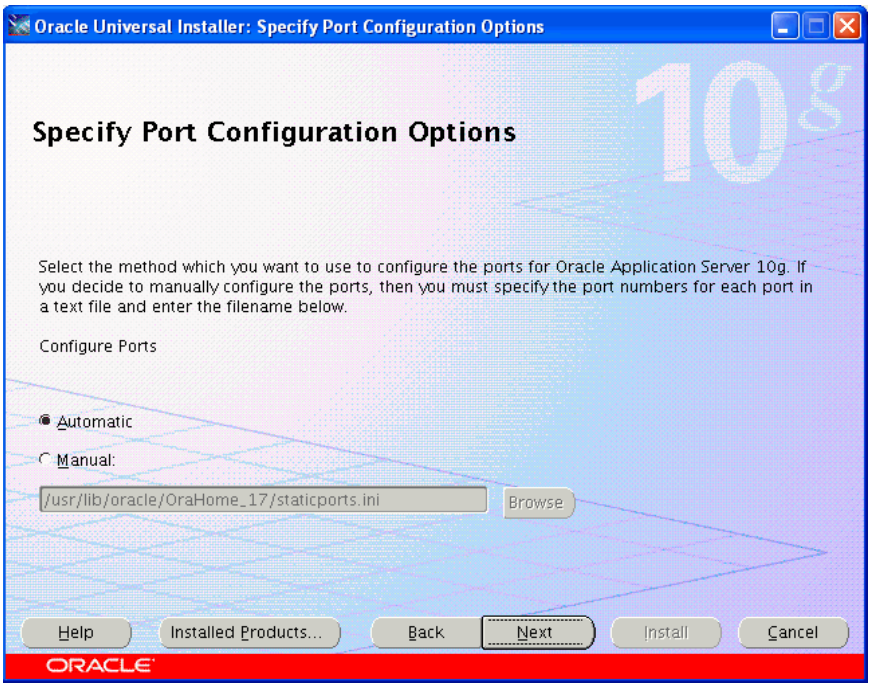


7. Choose how the port configuration will be determined. Oracle Universal Installer can configure the ports automatically, or you can specify a file, called the `staticports.ini` file, listing port numbers for the server.

   This is a sample `staticports.ini` file showing the file format. Replace port numbers with the values that you want to use for the component in question.

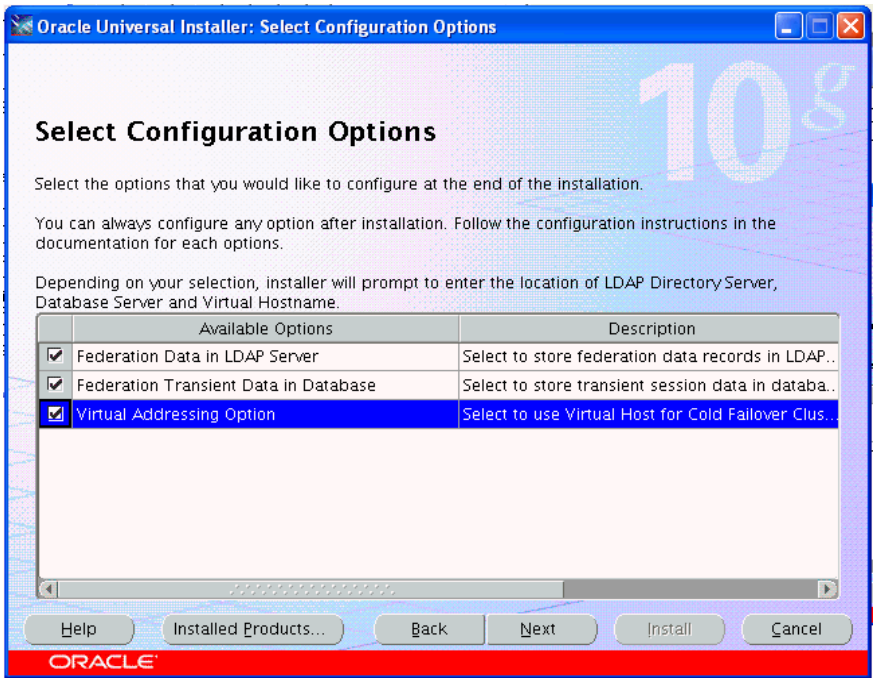
```
[System]
@ Host Name = sys04.my.company.com

[Ports]
Oracle HTTP Server port =  7778
Oracle HTTP Server Listen port = 7778
Oracle HTTP Server SSL port = 4444
Oracle HTTP Server Listen (SSL) port = 4444
Oracle Notification Server Request port = 6004
Oracle Notification Server Local port = 6102
Oracle Notification Server Remote port = 6201
Oracle HTTP Server Diagnostic port = 7201
Java Object Cache port = 7001
Oracle Management Agent Port = 1831
Application Server Control RMI port = 1851
Log Loader port = 44001
DCM Discovery port = 7101
Application Server Control port = 1810
```



Note: The `staticports.ini` file contains Federation, Apache, Opmn, DCM, and EM ports. See Using Custom Port Numbers (the "Static Ports" Feature) in the Oracle Application Server Installation Guide for your platform for additional details about the `staticports.ini` file.

8. Select configuration options to be implemented post-installation:
   - Federation record store - update the LDAP schema of the server where federation records will be stored.
   - Transient data store - transient data can be stored in a relational database; you will be presented with a second screen to provide the database information.
   - Virtual addressing - all components in the installation can be configured to use a virtual hostname; you will be presented with a second screen to specify a virtual hostname.

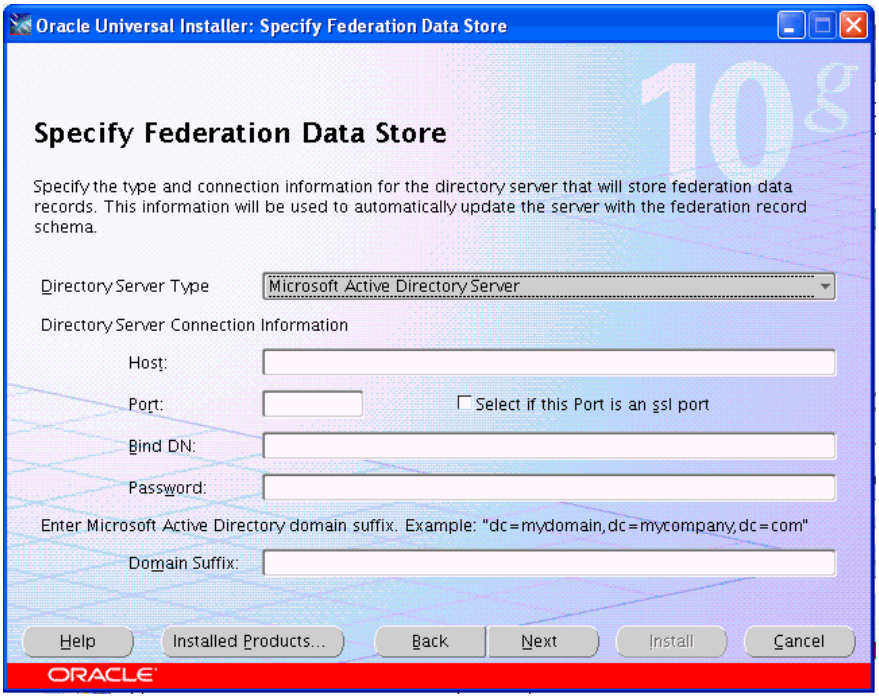


- If you elected to update an LDAP schema for your federation records, the installer now prompts you for details. You can choose between Oracle Internet Directory, Sun Java System Directory, and Microsoft Active Directory:

If the directory server is Oracle Internet Directory or Sun Java System Directory, specify:

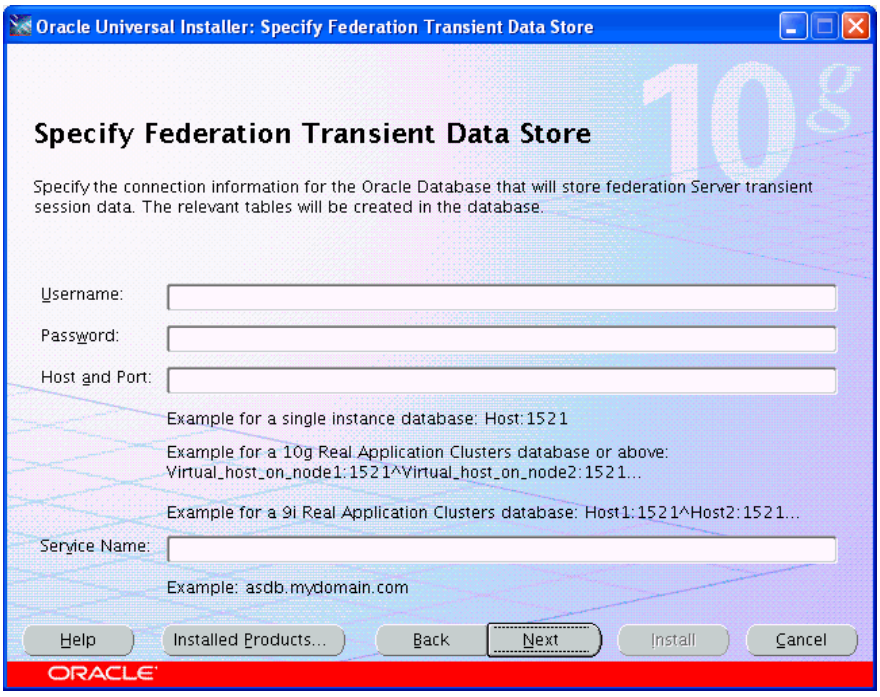
- the server hostname
- the port on which the server listens
- whether SSL is enabled or disabled

- the Oracle Internet Directory superuser name, or a single sign-on username with appropriate install privileges
- the password



If the directory server is Microsoft Active Directory, also specify the Domain Suffix.

- If you elected to store transient data in a relational database, the installer prompts you for details:



If you specified RDBMS storage for one or more types of transient data in Step 8, Oracle Universal Installer requests connection details for the database:

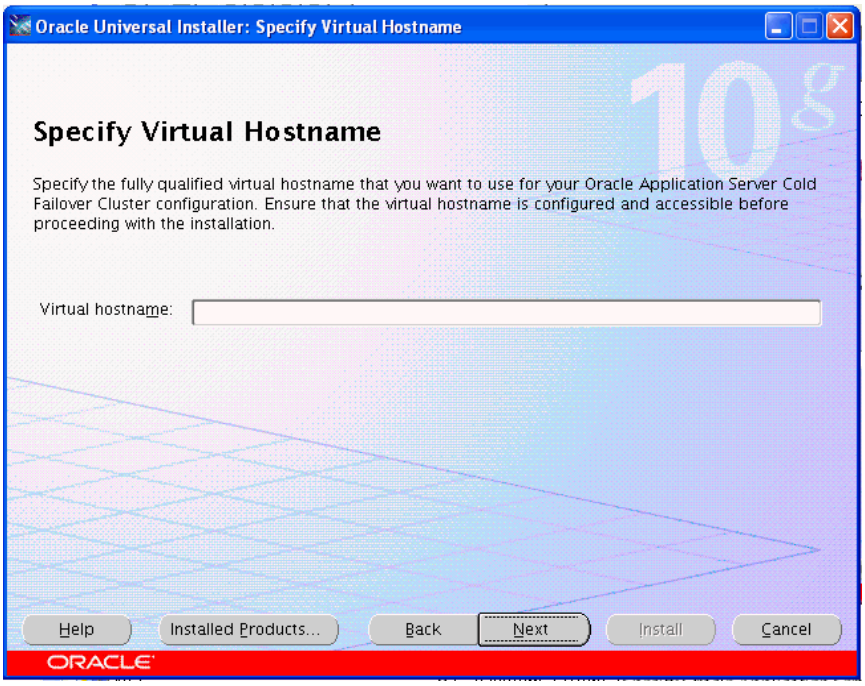
- the username and password of a non-administrator account that has connect and resource roles
- the hostname and the port number at which the server listens
- the Web service name

Note: Whether you can share an RDBMS transient store depends on how your Oracle Identity Federation server is deployed:

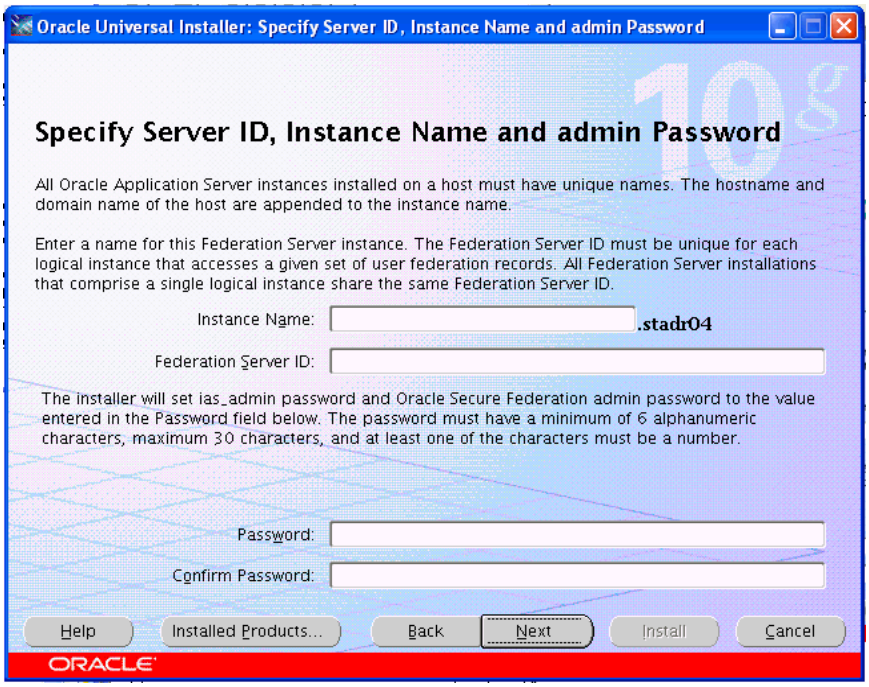
If the Oracle Identity Federation server will function as a standalone server, the database instance/database username combination must only be used by this Oracle Identity Federation instance; attempts to use the same RDBMS server/username to persist data for two Oracle Identity Federation servers will cause runtime conflicts around configuration and user session data.

If the Oracle Identity Federation Server is deployed in a clustered or load balanced environment, the same database instance/database username combination can be used for all Oracle Identity Federation servers that are part of the cluster/load balancing group. In this case all the Oracle Identity Federation instances will use the same configuration and back end user session store.

- If you elected to designate a virtual hostname, enter that information now.



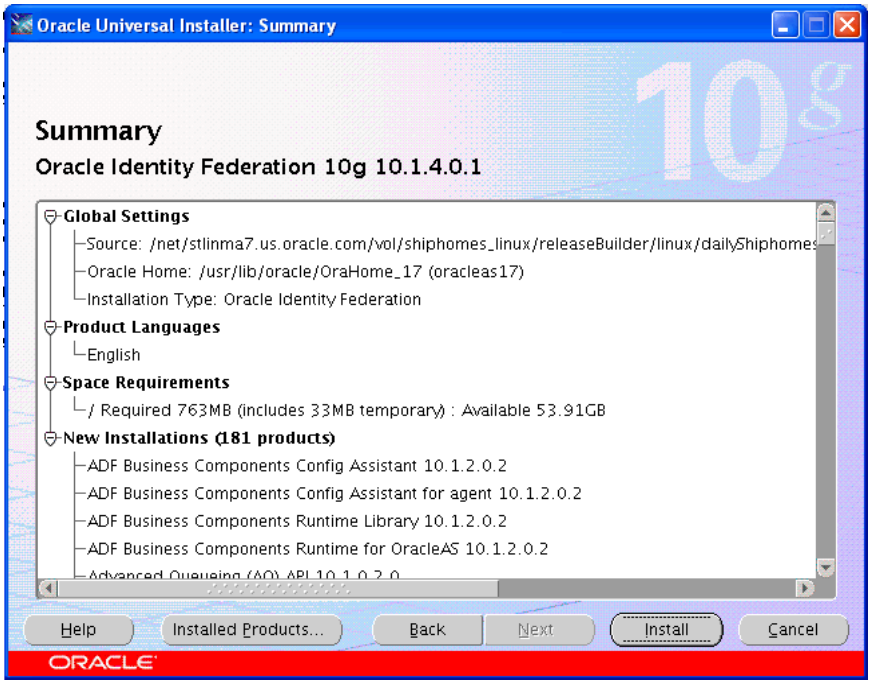
9. Specify Oracle Application Server hostnames, and the administrator password for this instance of Oracle Identity Federation.



Note: The administrator username is `oif_admin`.

Note: This step sets both the `ias_admin` password and the `oif_admin` password. The password field cannot be left blank.

10. Review the summary screen. To revise any information, press the **Back** button. To continue with the installation, press **Install**.



11. Oracle Universal Installer creates an instance of Oracle Containers for J2EE (OC4J) and Oracle Identity Federation.
12. The installer next directs you to the configuration assistant for default settings.
13. The Configuration Assistant configures and deploys the EAR file, modifies configuration files, and creates the federation data LDAP schema if this was requested.
14. The Oracle Universal Installer wizard exits.

Enabling SSL

When you install Oracle Identity Federation, the procedure also installs `SSLConfigTool` in the `$ORACLE_HOME/bin` directory. However, this does not configure SSL for the server. Note that:

- `SSLConfigTool` cannot be used to affect or modify Oracle Identity Federation SSL configuration. You use the Oracle Identity Federation administration console to configure the server to allow it to communicate with other components over SSL. See "Using SSL with Oracle Identity Federation" on page 6-133 for details.
- To enable SSL on the Oracle Application Server instance where Oracle Identity Federation is running, you must use `SSLConfigTool` to configure SSL communications for Oracle HTTP Server. For more information, see the *Oracle Application Server Administrator's Guide*, chapter titled "Enabling SSL in the Infrastructure."

Testing Your Installation

To check that the Oracle Identity Federation server installed correctly, you can access the Oracle Identity Federation administration console at `http://hostname:port/fedadmin`.

What To Do Next

After installation is complete, the Oracle Identity Federation administration console starts up automatically so that you can configure operational details such as:

- user ID repository settings
- authentication source
- overrides for default settings, if desired
- Circle of Trust (COT) metadata (optional)

For detailed information on these and other topics, refer to:

- Chapter 5, "Server Administration" for day-to-day administrative tasks, and for information on managing users and peer providers in the COT
- Chapter 6, "Configuring Oracle Identity Federation" for server configuration details

Reassociating the Server

You may need to change the network configuration to point your Oracle Identity Federation server to a different Infrastructure instance. This process (also referred to as

reassociation) is necessary, for example, when Oracle Identity Federation server is ready to move from a test environment to a production Infrastructure.

For details of the reassociation procedure, see the *Oracle Application Server Administrator's Guide*. In Task 8: Update Oracle Identity Federation, Steps 1 and 2 explain how to perform the Infrastructure change. The remaining steps apply if you reassociate Oracle Identity Federation with a different Oracle Internet Directory or OracleAS Single Sign-On.

4

Deploying Oracle Identity Federation

This chapter describes key deployment scenarios, including integration with identity and access management systems, Web servers, and back-end data stores. It contains these topics:

- Introduction
- Deployment Scenarios

Introduction

Oracle Identity Federation operates in a heterogeneous system environment, and is capable of working with a wide range of applications serving as identity providers, service providers, data stores, and web servers.

To resolve deployment issues and questions, refer to Chapter 2, "Planning Oracle Identity Federation Deployment", which provides extensive background information to help you plan your deployment:

- "Architecture Options" on page 2-1 provides details about supported protocols and profiles, and what to consider when evaluating deployment options.
- "Sizing Guidelines" on page 2-22 explains performance factors and provides topology recommendations.
- "Implementation Checklist" on page 2-28 provides a deployment checklist.

The next section provides step-by-step instructions for configuring Oracle Identity Federation to work with key components of the federated environment.

Deployment Scenarios

This section describes the steps needed to implement common Oracle Identity Federation deployment scenarios. It contains these sections:

- Deploying Oracle Identity Federation with OracleAS Single Sign-On
- Deploying Oracle Identity Federation with Oracle Access Manager
- Deploying Oracle Identity Federation with eTrust SiteMinder
- Deploying Oracle Identity Federation with a Sun Java System Web Server
- Configuring Oracle Identity Federation to Use IBM Tivoli Directory Server as the Data Store
- Integrating with Third-Party Identity & Access Management Modules
- Implementing HTTP Basic Authentication

- Integrating WebGate with Oracle Identity Federation Server

Deploying Oracle Identity Federation with OracleAS Single Sign-On

This section describes the steps needed to install and deploy Oracle Identity Federation so that it is integrated with OracleAS Single Sign-On.

Deployed in this manner, Oracle Identity Federation can leverage the authentication capabilities offered by OracleAS Single Sign-On when local user authentication is required. Oracle Identity Federation can:

- act as an identity provider to authenticate a user and provide the user's authentication information to any third party, or
- act as a service provider that consumes authentication data from an identity provider in order to authenticate a user.

Briefly, the steps to achieve this deployment are:

- Install Oracle Identity Federation using the advanced installation mode, electing to store federation data in Oracle Internet Directory. Optionally, store transient data in a database.
- Integrate Oracle Identity Federation with OracleAS Single Sign-On. This involves, among other things, updating the OracleAS Single Sign-On environment to add Oracle Identity Federation as an authentication mechanism, and associating the server instance with OracleAS Single Sign-On.
- Update Oracle Identity Federation configuration to provide connection details for the OracleAS Single Sign-On and Oracle Internet Directory servers, and exchange metadata with peer providers in the Circle of Trust.

Detailed instructions for these steps follow.

Note: Oracle Identity Federation does not support the ability to force re-challenging the user for credentials when integrated with OracleAS Single Sign-On. This means that Oracle Identity Federation cannot support use cases where reauthentication must be forced.

For example, if an SP sends an AuthnRequest with `ForceAuthn="true"` to an Oracle Identity Federation IdP, and Oracle Identity Federation is integrated with OracleAS Single Sign-On, the `ForceAuthn` flag is ignored.

Install Oracle Identity Federation

Perform these installation steps:

1. Launch Oracle Universal Installer. Select the **Oracle Identity Federation 10g** product, and choose the **Advanced** installation method.
2. On the **Specify Federation Data Store** screen, select `Oracle Internet Directory` as the directory server type, and enter information about the server in the input fields. In this example, the Oracle Internet Directory server hostname and port, respectively, are `infra.example.com` and `389`:

Field	Sample Value
Host	infra.example.com
Port	389

Field	Sample Value
Bind DN	`cn=orcladmin`
Password	password for `orcladmin`

> **Note:** These LDAP connection credentials are used only to update the directory with the federation data schema. Different credentials are typically configured later for runtime directory access.

3. If you selected the **Federation Transient Data in Database** option, a database user must be available with privileges to create tables.

   Rather than using system table space for the transient data, it is recommended that table space be allocated to this user. For example, using SQL*Plus and connecting to the database as user `sysdba`, the following commands create a user named `oifdb` and allocate table space for that user:

```
create tablespace ts_oifdb
   logging
   datafile '/scratch/Oracle/i0120/oradata/i0120/ts_oifdb.dbf'
   size 512m
   autoextend on extent management local;

create user oifdb
   identified by oifdb
   default tablespace ts_oifdb;

grant connect,resource to oifdb;

alter user oifdb account unlock;
```

4. On the **Specify Federation Transient Data Store** screen, enter your database connection information - username, password, host, port, and Web service name.
5. Complete the remainder of the Oracle Identity Federation installation, specifying the federation server ID, instance name, and administrator password.

> **Note:** For installation details, see "Advanced Installation Procedure" on page 3-7.

Integrate Oracle Identity Federation and OracleAS Single Sign-On

These steps 1) make the Oracle Identity Federation server host known to OracleAS Single Sign-On, and 2) associate the Oracle Identity Federation instance with OracleAS Single Sign-On.

1. In the Oracle IdM/Infrastructure home, edit the `sso/conf/policy.properties` file by uncommenting and modifying the following lines, where `oif.example.com:7780` is the host and port of the Oracle Identity Federation server:

```
SASSOAuthnUrl = http\://oif.example.com\:7780/sso/authn

SASSOLogoutUrl =
   http\://oif.example.com\:7780/sso/jsp/sasso_logout_success.jsp
SASSOAuthLevel = MediumHighSecurity
```

2. Add the following lines to the `sso/conf/policy.properties` file, where `content.example.com:8888` is the host and port of the resource server:

```
content.example.com\:8888 = MediumHighSecurity

MediumHighSecurity_AuthPlugin =
    oracle.security.sso.server.auth.SASSOAuth
```

3. Copy the SSO keystore from the Oracle Identity Federation home to the Infrastructure home. For example:

```
cp OIF_HOME/sso/conf/keystore INFRA_HOME/sso/conf/
```

4. Register partner applications with OracleAS Single Sign-On as usual. For example, if you have a resource at `/scratch/protected/index.html`, with a virtual host on 140.87.26.53:8888, make these edits to the `Apache/Apache/conf/httpd.conf` file in the Infrastructure home:

   - Add one of these lines before `<IfDefine SSL>`, at the end of the `LoadModule` section:

     For Linux:

     ```
     LoadModule osso_module libexec/mod_osso.so
     ```

     For Windows:

     ```
     LoadModule osso_module modules/ApacheModuleOSSO.DLL
     ```

   - Also for Windows, at the end of the `AddModule` section, before `<IfDefine SSL>`, add the following line:

     ```
     AddModule mod_osso.c
     ```

   - Add these lines before `"# Include the configuration files needed for mod_oc4j"`:

     ```
     Listen 8888
     NameVirtualHost 140.87.26.53:8888
     <VirtualHost 140.87.26.53:8888>
        ServerName content.example.com
        DocumentRoot "/scratch/protected"
        OssoConfigFile
           "/scratch/Oracle/i0120/Apache/Apache/conf/osso/osso-app.conf"
        OssoIpCheck off
        <Location /index.html>
           AuthType basic
           Require valid-user
        </Location>
     </VirtualHost>
     ```

   - Run the `ssoreg` script, which is `ssoreg.sh` on Linux, and `ssoreg.bat` on Windows. For example:

     ```
     sso/bin/ssoreg.sh -oracle_home_path /scratch/Oracle/i0120
        -site_name content.example.com -config_mod_osso TRUE
        -mod_osso_url http://content.example.com:8888 -virtualhost
        -config_file
           /scratch/Oracle/i0120/Apache/Apache/conf/osso/osso-app.conf
     ```

5. Restart Infrastructure:

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

6. To associate the Oracle Identity Federation instance with OracleAS Single Sign-On, open the Oracle Identity Federation Enterprise Manager console in a web browser. For example:

   `http://infra.example.com:1810/emd/console`

   Perform these steps:

   - Go to **Infrastructure** -> **Identity Management** and click the **Change** button.
   - Enter the Oracle Internet Directory host and port, and click **Next**.
   - Enter the Oracle Internet Directory username (for example `cn=orcladmin`) and password, and click **Next**, then **Finish**.
   - From the Enterprise Manager main page, restart Oracle HTTP Server, Home, and OC4J_FED.

Configure Oracle Identity Federation

These steps 1) provide Oracle Identity Federation with the information needed to connect to data stores, and 2) update and distribute the Oracle Identity Federation metadata to peer providers.

1. Open the Oracle Identity Federation administration console in a web browser:

   `http://oif.example.com:7780/fedadmin`

   Log in as `oif_admin`.

2. On the **IdM Data Stores** -> **User Data Store** screen, select **OracleAS Single Sign-On** and enter the connection information. For example:

Field	Example Value
Connection URL(s):	`ldap://infra.example.com:389` This is the Oracle Internet Directory instance used by OracleAS Single Sign-On.
Bind DN:	`cn=orcladmin`
Password:	the password for `orcladmin`
User ID Attribute:	`uid`
User Description Attribute:	`uid`
Person Object Class:	`inetorgperson`
Base DN:	`dc=example,dc=com`
Other properties' default values	
OSSO Login URL:	`http://infra.example.com:7777/sso/auth`
OSSO Logout URL:	`http://infra.example.com:7777/sso/logout`

3. On the **IdM Data Stores** -> **Federation Data Store** screen, select **LDAP Directory** and enter the connection information. For example:

Field	Sample Value
Connection URL(s):	`ldap://infra.example.com:389` This is the Oracle Internet Directory instance used by OracleAS Single Sign-On.

Field	Sample Value
Bind DN:	`cn=orcladmin`
Password:	the password for `orcladmin`
User Federation Record Context:	`cn=fed,dc=example,dc=com`
LDAP Container Object Class:	`<blank>`
Unique Federation ID Attribute:	`<blank>`

4. Click **Save**.
5. Go to the Oracle Identity Federation Enterprise Manager console and restart OC4J_FED.
6. Go to the Oracle Identity Federation administration console, and navigate to the **Server Configuration** -> **Circle of Trust** screen.

   In the **Add Trusted Provider** section, browse to the file system location of a peer provider's metadata XML document, and enter descriptive text for that provider. Click **Add**, then click **Done**.

   Click **Refresh Server**.
7. If configuring Oracle Identity Federation to be a service provider, go to the **Server Configuration** -> **Service Provider** -> **Global Settings** screen, and select a **Default SSO Identity Provider** from the list box.

   Click **Save**, then **Refresh Server**.
8. Each peer provider in the circle of trust will need a copy of the Oracle Identity Federation metadata XML document. Start by accessing the metadata URL for the particular server role (SP or IdP) and the federation protocol version (Liberty 1.1, 1.2 or SAML 2.0) in question. For example:

   `http://oif.example.com:7780/fed/sp/metadatav20`

   `http://oif.example.com:7780/fed/idp/metadatav20`
9. Save the XML file retrieved from the URL, and distribute it to the other providers in the circle of trust. If setting up another Oracle Identity Federation instance as part of the circle of trust, this is the file you would load using **Add Trusted Provider** on the **Circle of Trust** screen.

Testing Federated Single Sign-On

Take these steps to test your federated single sign-on setup:

1. Use a web browser to access a protected resource. When prompted by the Identity Provider, log in using credentials in the IdP's domain. When prompted by the Service Provider, log in using credentials in the SP's domain. You should now be redirected to the protected resource.
2. Log out, and then try to access the protected resource again. You should be prompted for login only by the Identity Provider.

Deploying Oracle Identity Federation with Oracle Access Manager

This section describes the steps needed to install and deploy Oracle Identity Federation so that it is integrated with Oracle Access Manager. The steps illustrate a deployment scenario consisting of two nodes:

- Node A, referred to as host_a (and with an associated URL of the type `host-a.us.oracle.com`), is a service provider (SP) type server.
- Node B, referred to as host_b (and with an associated URL of the type `host-b.us.oracle.com`), is an identity provider (IdP) type server.

The section is broken out into separate instructions for the different component installation and deployment tasks:

- Install OracleAS Infrastructure
- Install Oracle Access Manager
- Install Oracle Identity Federation
- Integrate Oracle Identity Federation and Oracle Access Manager

Install OracleAS Infrastructure

This section explains how to install OracleAS Infrastructure.

> **Note:** You only need to install the OracleAS Infrastructure with Oracle Access Manager if Oracle Access Manager is going to use Oracle Internet Directory as its directory. Otherwise, the OracleAS Infrastructure does not need to be installed.

1. Launch Oracle Universal Installer, and select the **Oracle Application Server Infrastructure 10g** installation option.
2. Select **Identity Management and Metadata Repository**.
3. Use the default configuration options.
4. After installation is completed, establish database connection.
   - Run the `coraenv` script to set the proper values of the `ORACLE_SID` and `ORACLE_HOME` variables.
   - Connect to the database:

     ```
     sqlplus '/ as sysdba'
     ```
5. Run the following SQL commands:

```
create tablespace ts_fd
   logging
   datafile '/scratch/aswu/Oracle/i0120/oradata/i0120/ts_fd01.dbf'
      size 512m autoextend off,
   '/scratch/aswu/Oracle/i0120/oradata/i0120/ts_fd02.dbf'
      size 512m autoextend off
   extent management local;

create user fd identified by fd default tablespace ts_fd;
grant connect,resource to fd;
alter user fd account unlock;
```

Install Oracle Access Manager

Several Oracle Access Manager components must be installed for use with Oracle Identity Federation:

- Identity Server

- WebPass installed in a Web server
- Access Server
- Oracle Access Manager (administration UI) installed on the same Web server as WebPass
- if the Oracle Identity Federation or Oracle Access Manager site is a service provider (SP), WebGate agents installed on each Web server providing resources or services that are available to federated users

Refer to the *Oracle Access Manager Installation Guide* for details.

Considerations for Oracle Access Manager Installation

When installing and deploying Oracle Access Manager, pay special attention to issues critical for integration with Oracle Identity Federation:

- When configuring the Access Server entry in the Access Manager console, set **Access Management Service** to **On**.

  Note: By default, **Access Management Service** is set to **Off**. Oracle Identity Federation requires that this field be set to **On**.

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

- When enabling default policies, it is highly recommended that you set up the Oracle Access and Identity Basic Over LDAP authentication scheme (previously known as the NetPoint Basic Over LDAP authentication scheme). If this is not done, you will need to configure a basic scheme manually.

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

- As mentioned earlier, WebGate agents must be installed on each Web server providing resources or services available to federated users if the Oracle Identity Federation or Oracle Access Manager site is a service provider (SP). Note the following when configuring Webgates:

  – The Access Management Service setting must match the setting for the Access Server(s) used by the WebGate. So, if the WebGate uses the same Access Server(s) as Oracle Identity Federation will use, then it must be configured with the Access Management Service set to **On**. It is also possible for a WebGate to use a different Access Server instance(s) (in the same domain) with the Access Management Service set to **Off**, in which case the Web setting would be **Off** as well.

  – It is normal practice to set the Primary HTTP Cookie Domain to enable Oracle Access Manager single sign-on across web servers with installed WebGates. At a minimum, the cookie domain must include the Oracle Identity Federation host and at least one WebGate-protected web server. For example, if Oracle Identity Federation is on the host `oif.us.company.com` and the Web server is `www.us.company.com`, the domain setting should be `.us.company.com` or `.company.com`. If the Web server is `www.company.com`, the domain setting should be `.company.com`. *Note*: The default AccessGate setting for the cookie domain is empty (no domain), which will only work in a very atypical deployment where Oracle Identity Federation and all protected resources reside on the same host.

Install Oracle Identity Federation

This section explains how to install Oracle Identity Federation for use with Oracle Access Manager. This is a brief summary of the necessary steps. For details, see "Advanced Installation Procedure" on page 3-7.

1. Launch Oracle Universal Installer, and select the **Oracle Identity Federation 10g** installation option.
2. Select the **Advanced** installation method.
3. In **Select Configuration Options**, select **Federation Data in LDAP Server** and **Federation Transient Data in Database**.
4. In **Specify Federation Data Store**, provide this information:
   - **Server Type** - Oracle Internet Directory
   - **Host/Port** - the LDAP server host and port
   - **Bind DN** - cn=orcladmin
5. In **Specify Federation Transient Data Store**, provide this information:
   - **Username**
   - **Password**
   - **Host, Port and Service Name** - the database for transient data

Integrate Oracle Identity Federation and Oracle Access Manager

This section explains how to integrate Oracle Identity Federation and Oracle Access Manager. This includes certain steps in both environments, such as configuring an AccessGate for Oracle Identity Federation (in Oracle Access Manager) and setting data store and other configuration parameters (in Oracle Identity Federation).

> **See Also:** See the *Oracle Access Manager Access Administration Guide* for details.

1. Use the Access System Console `http://AMhost:AMport/access/oblix` (where *`AMhost:AMport`* is the web server where you installed WebPass and Access Manager) to configure an AccessGate for Oracle Identity Federation.
   a. Select the **Access System Configuration** tab.
   b. Select the **Add New AccessGate** link from the console panel.
   c. Configure the AccessGate as follows, replacing the values in italics with your own values:

> **Note:** The AccessGate can have any name of your own choosing. It is named *OIF* in this example only for illustration.

```
AccessGate Name: OIF
Password: OIF-PASSWORD
Hostname: OIF-HOST
Port: OIF-PORT
Transport Security: Match the Access Servers to be configured in Step d.
Access Management Service: On
Primary HTTP Cookie Domain: .company.com (Note: As noted in the WebGate
configuration, the Primary HTTP Cookie Domain configured for Oracle
Identity Federation must match the Primary HTTP Cookie Domains configured
```

for the WebGates protecting the content to be accessed by federated users.)
Preferred HTTP Host: *OIF-HOST*

Click **Save**.

d. Click **List Access Server**.

Click **Add**.

Select one or more servers from drop-down menu. *Note*: All selected Access Servers must have Access Management Service **On**.

Number of connections: 1
Click **Add**.

2. Use the Access System Console to configure the Fed HostID, if required.

a. Select the **Access System Configuration** tab.

b. Select the **Host Identifiers** link from the console panel.

c. If no host identifiers are defined, you do not need the Fed HostID. Skip to Step 3.

d. If host identifiers are defined, click the **Add** button.

Name: Fed HostID
Note: Enter the same fixed value for all supported languages.

Hostname variables: *OIF-HOST:OIF-PORT*
Note: If OIF-PORT is 80 or 443, also include OIF-HOST.

Click **Save**.

3. Install the Access Server SDK:

a. Run the AccessServerSDK installer (for example, Oracle_Access_Manager10_1_4_0_1_linux_AccessServerSDK) under OIF_HOME/fed/shareid/.

b. If installing on Linux, set the LD_ASSUME_KERNEL environment variable.

Open the Enterprise Manager console for the Oracle Identity Federation installation in a web browser. For example:

http://oif.example.org:1810/emd/console

Perform these steps:

- Under System Components, click the link for OC4J_FED.

- Go to **Administration** - > **Server Properties** and, under Environment Variables, click **Add Environment Variable**.

- In the new entry, enter LD_ASSUME_KERNEL in the Name field, and enter 2.4.19 in the Value field. Leave the Append checkbox unchecked.

- Click **Apply**.

- Click **OK** to restart the OC4J_FED container.

4. Go to the Oracle Identity Federation administration console at http://OIF-HOST:OIF-PORT/oifadmin. Click on the **IdM Data Stores** tab.

Note: Substitute parameter values (bind DN, password, DNs, and so on) as required for your directory.

For the federation data store:

```
Bind DN: cn=orcladmin
Password: your-password
User Federation Record Context: cn=fed,dc=us,dc=oracle,dc=com
```

For the user data store:

```
Active Repository: Oracle Access Manager
Connection URL(s):  ldap://LDAP-Server-Host:Port
Bind DN: cn=orcladmin
Password: your-password
User ID Attribute: uid
User Description Attribute: uid
Person Object Class: inetOrgPerson
Base DN: dc=us,dc=oracle,dc=com
```

For Oracle Access Manager configuration parameters:

```
Master Admin Login ID: orcladmin
Master Admin Password: your-password
Authorization result for unprotected resources: Allow
Oracle Access Manager Cookie Domain: .company.com
Basic Authentication Scheme Name: Oracle Access and Identity authentication
scheme
```

Note:

- **Oracle Access Manager Cookie Domain:** As noted in the WebGate configuration, the Primary HTTP Cookie Domain configured for Oracle Identity Federation must match the Primary HTTP Cookie Domains configured for the WebGates protecting content to be accessed by federated users.
- **Basic Authentication Scheme Name**: Use the Access System Console (`http://AMhost:AMport/access/oblix`) to find a suitable basic authentication scheme. If you enabled the default policies when you installed Oracle Access Manager (by selecting Access Manager in the Oracle Access Manager console, checking the two policies and clicking **Enable**), you can use the basic scheme created for those policies:
  - For Oracle Access Manager 10.1.4: Oracle Access and Identity Basic Over LDAP authentication scheme
  - For COREid 7.0.4: NetPoint Basic Over LDAP

  If no basic schemes are configured, you must set one up following instructions in the *Oracle Access Manager Access Administration Guide*. You can cut and paste the display name for the chosen basic scheme from the Access System Console to the Oracle Identity Federation User Data Store page.

Click **Apply**.

Use these credentials for the Access Server:

- Access Server Host Name: *access-server-host*

Note: This must be one of the servers configured in step 1d.

- Access Server Port: *access-server-port*
- Access Gate ID: OIF
- Access Gate Password: *OIF-password*
- Connection Type: must match the access servers

Restart the Oracle Identity Federation server.

5. To make sure that the integration is complete:
   - Log into the Access System Console `http://AMhost:AMport/access/oblix/`.
   - Click **Access Manager**.
   - Check **Fed Domain** created in **My Policy Domains**.
6. To create a resource protected by Oracle Identity Federation (as a service provider):
   a. Follow the steps for protecting resources in the *Oracle Access Manager Access Administration Guide*, in the chapter Protecting Resources with Policy Domains.
   b. Change the authentication scheme to one of:

      Fed SSO - SAML2.0/Liberty 1.x – to use SAML 2 or Liberty 1.1 or 1.2 SSO profiles

      Fed SSO - WS-Federation – to use the WS-Federation Passive Requester SSO profile

      Fed SSO - SAML 1.x – to use SAML 1.0 or 1.1 SSO profiles

      You may need to ensure that the security levels of any existing authentication schemes and the Fed SSO authentication schemes are configured in Oracle Access Manager such that the Fed SSO scheme is selected when the protected resource is accessed. For example, if your Fed SSO authentication scheme has security level 1, and your basic login authentication scheme has security level 2, then the user will always be challenged for login even if federated single sign-on succeeds.

      Note: The choice is determined by the protocols that the partner identity providers support. If the same resource will be accessed by partners requiring different protocols, you must set up separate policies with different resource URLs to map the requests to the appropriate authentication scheme. For example, to provide access to a resource `/my-resource` using both SAML 2 and WS-Federation, you can set up:

      - a policy for a resource `/my-resource-saml2` that uses the Fed SSO – SAML2.0/Liberty 1.x scheme, and
      - a policy for a resource `/my-resource-wsfed` that uses the Fed SSO – WS-Federation scheme.

      You must then configure the web server to map these pseudonymous URLs to the actual `/my-resource` URL.

Deploying Oracle Identity Federation with eTrust SiteMinder

This section describes how to integrate Oracle Identity Federation with eTrust SiteMinder. Deployed in this manner, Oracle Identity Federation can leverage the identity and access management capabilities of eTrust SiteMinder.

This discussion assumes the following framework:

- eTrust SiteMinder is the Identity and Access Management Server.
- The eTrust SiteMinder User Directory is an LDAP server.
- The User Data Store used by Oracle Identity Federation is an RDBMS.
- The SP functionality will not be exercised, so it is not necessary to install the SiteMinder native library on the eTrust SiteMinder Policy Server.

> **Note:** Oracle Identity Federation requires certain eTrust SiteMinder policy objects to be created on the policy server. Oracle Identity Federation normally creates these automatically at initialization with the eTrust SiteMinder Policy Server, but they may also be created manually. Examples of policy objects created by Oracle Identity Federation are provided in "eTrust SiteMinder Policy Objects" on page 4-17.

This chapter contains the following topics:

> **See Also:** "Additional eTrust SiteMinder Configuration" on page 7-67 for more information about eTrust SiteMinder integration with Oracle Identity Federation.

Requirements for Integrating with eTrust SiteMinder

You must meet the following requirements to integrate with eTrust SiteMinder:

- eTrust SiteMinder SDK 5.5 must be installed on the Oracle Identity Federation machine.
- The eTrust SiteMinder Policy Server must be at version 5.5.

Installing the eTrust SiteMinder SDK

In this deployment, Oracle Identity Federation will act as an eTrust SiteMinder Agent, using the eTrust SiteMinder SDK to communicate with the eTrust SiteMinder Policy Server. Perform these steps to install the SDK:

1. Install the eTrust SiteMinder SDK Version 5.5 to the `$ORACLE_HOME/fed/shareid/SiteMinder` directory.
2. Restart the `OC4J_FED` instance.

Defining the RDBMS DataSource

Define the database that will act as the user data store for Oracle Identity Federation as a DataSource in Oracle Application Server.

The steps to define such a DataSource are as follows:

1. Log in to the EM console of your Oracle Identity Federation instance and navigate to **OC4J_FED** - > **Administration** - > **Data Sources**.
2. Create a new data source using the following example as a guide:

```
Name:                myDS
Data Source Class:   com.evermind.sql.DriverManagerDataSource
JDBC URL:            jdbc:oracle:thin:@stahs08.us.oracle.com:1521:ORCL (enter
                     the correct connection URL)
JDBC Driver:         oracle.jdbc.driver.OracleDriver
Username:            CUSTDATA (replace with the username used to access
                     the RDBMS)
Password:            PASSWORD (replace with the password used to access
                     the RDBMS)
Location:            jdbc/RDBMSUserDataSource
Transactional Loc:   jdbc/xa/RDBMSUserDataSource
EJB Location:        jdbc/RDBMSUserDataSource
```

> **Note:** The database connection information must be updated to reflect your deployment configuration; refer to the Oracle Application Server and Oracle JDBC documentation for more information.

3. Apply the changes.

Configuring the Oracle Identity Federation User Data Store

Take these steps to configure Oracle Identity Federation with eTrust SiteMinder to use an RDBMS User Data Store:

1. Log into the Oracle Identity Federation administration console and navigate to **IdM Data Stores** - > **User Data Store**.
2. Select eTrust SiteMinder from the Active Repository list.
3. Select 'Relational Database' as the back-end data store and enter the appropriate values, as explained in this table:

Field	Example	Comment
JNDI Name	jdbc/RDBMSUserDataSource	Must match the EJB Location entered in the Oracle Enterprise Manager console
User Name	CUSTDATA	Replace with the username used to access the RDBMS
Password	PASSWORD	Replace with the password used to access the RDBMS
Login Table	EMPLOYEES	Replace with the name of the RDBMS table containing the user records
Login ID Column	EMPLOYEE_ID	Replace with the column name containing the user identifiers

Field	Example	Comment
Login Pwd Column	Empty	This attribute is not used when configured with eTrust SiteMinder Server for authentication.
Password Digest Algorithm	None.	A password digest algorithm is not used in this deployment.
User Desc. Attribute	EMPLOYEE_ID	Replace with the value used in the Login ID Column field. This field must be a column containing a user identifier that will be used to create the federation records: this attribute provides a way to have a human-readable attribute set on the user federation record. This field is useful when the main user identifier is not human-readable. In this case, since the main user identifier is the username, setting that field to the Login ID Column value is sufficient.

Click **Apply** to save the changes.

4. Enter the connection information to the eTrust SiteMinder policy server:

   See Also: "Edit User Data Store" on page 6-67

   - Host - This is the host where the policy server is installed.
   - Authorization Port - This is the policy server port used for authentication requests. The default value is 44442.
   - Authentication Port - This is the policy server port used for authentication requests. The default value is 44443.
   - Accounting Port - This is the policy server port used for accounting requests. The default value is 44441.
   - Max Connections - This is the maximum number of agent connections to the policy server.
   - Min Connections - This is the minimum number of agent connections to the policy server.
   - Step Connections - This is the number of connections the agent can open to the policy server at one time.
   - Timeout - This is the time, in seconds, that the agent will wait for a response from the policy server before it returns a failure.

5. Enter the following information in the eTrust SiteMinder Agent Configuration section:

   - Enter the Agent Name and the Agent Secret of an existing eTrust SiteMinder Web Agent. That agent's credentials are used at startup time to verify that the Oracle Identity Federation SMBridgeAgent exists and is correctly registered.
   - Enter the Cookie domain used by the eTrust SiteMinder policy server. Oracle Identity Federation uses this value when locally authenticating a user and setting the SMSESSION Cookie.

- Enter the Oracle Identity Federation SMBridgeAgent password. This password must match the password of the eTrust SiteMinder Agent identified as SMBridgeAgent on the Policy Server.

6. Enter the following information in the Automatic Policy Creation section:
   - Admin ID - This is the username of an eTrust SiteMinder account to access the Policy server.
   - Admin Password - This is the password to access the Policy server.
   - Domain Name - This is the domain in which the policy objects are contained.
   - User Directory - This is the user directory for the domain.

   This information is used at startup time to verify that the necessary eTrust SiteMinder policy objects required by Oracle Identity Federation are present in the eTrust SiteMinder policy server.

   If the policies are already created and correctly configured, the credentials specified in this section can reference an eTrust SiteMinder account with only `read` privileges.
7. Click **Apply**.

Since Oracle Identity Federation was not designed to use a User Data Store different from the eTrust SiteMinder server, the following configuration steps are required to retrieve the correct user identifier during in a Single Sign-On flow:

1. Log in to the Oracle Identity Federation administration console and navigate to **IdM Data Stores** - > **User Data Store**.
2. Select LDAP Directory as the back-end data store for the eTrust SiteMinder server.
3. In the User ID Attribute field, enter the attribute used in the LDAP user record containing the user identifier. For common LDAP servers, it should be set to `uid`.
4. Click **Apply**.
5. Navigate to **IdM Data Stores** - > **User Data Store**.
6. Select Relational Database as the back end data store.
7. Click **Apply**.
8. Restart `OC4J_FED`.

Configuring the eTrust SiteMinder Web Agent

eTrust SiteMinder Web Agents need to be configured to do the following:

- The Web Agent needs to specify the domain on the SMSESSION cookie, so that Oracle Identity Federation will be able to consume the SMSESSION cookie.
- The Web Agent needs to accept the SMSESSION cookies set by Oracle Identity Federation, in the case where the Oracle Identity Federation eTrust SiteMinder Agent will authenticate the user.

Update the Cookie Domain Before changing the Web Agent configuration, take these steps to determine whether the agent is using local configuration files:

1. Identify the eTrust SiteMinder Web Agent responsible for setting the SMSESSION cookie: this is the agent authenticating the user.
2. Go to the eTrust SiteMinder Administration Console.
3. Go to the Web Agent Config Objects section.

4. Select the entry for the eTrust SiteMinder Web Agent you identified in the first step.
5. Find the `AllowLocalConfig` parameter in the list.

Subsequent action depends on the value of this parameter.

If the `AllowLocalConfig` parameter value is "Yes", then you will need to modify the `WebAgent.conf` file of the eTrust SiteMinder Web Agent.

This file is typically located on the machine where the Web Agent is installed, in the directory where the eTrust SiteMinder Web Agent binaries and configuration files were installed: `$HOME\Netegrity\SiteMinder Web Agent\Bin\`*`$HTTP_SERVER_NAME`*`\WebAgent.conf`, with `$HOME` being the parent directory of the eTrust SiteMinder Web Agent installation directory (for example, `c:\Program Files`), and *`$HTTP_SERVER_NAME`* being the directory name that corresponds to the HTTP Web Server integrated with the Web Agent (for example, IIS if eTrust SiteMinder Web Agent is integrated with Microsoft IIS).

In this `WebAgent.conf` file, add the following parameter:

```
CookieDomain=".domain.com"
```

and replace `.domain.com` with the domain on which you are deployed.

Save the changes and restart the HTTP Server to force the Web Agent to reload its configuration.

If the `AllowLocalConfig` parameter value is "No", then you will need to take these steps to modify the Web Agent Configuration object on the eTrust SiteMinder Policy Server:

1. Go to the eTrust SiteMinder Web Agent entry.
2. Go to the Web Agent Config Objects section.
3. Locate the `CookieDomain` parameter, and set its value to the domain on which you are deployed.
4. Save the changes and restart the HTTP Server to force the Web Agent to reload its configuration.

Allow eTrust SiteMinder Web Agent to accept SMSESSION Cookies Following steps similar to those you used to determine whether the Web Agent uses local configuration files (in the section titled "Update the Cookie Domain"), update the `AcceptTPCookie` property to "Yes" to allow the eTrust SiteMinder Web Agent to consume SMSESSION cookies set by Oracle Identity Federation.

This property will be required if the user first authenticates at the Oracle Identity Federation server.

eTrust SiteMinder Policy Objects

This section lists, by way of example, the policy objects from an eTrust SiteMinder Policy Server after post-initialization of the Oracle Identity Federation/eTrust SiteMinder connector.

```
#! SiteMinder Version 5.5
# Export Flags: Encrypted.
objectclass: Scheme
Oid: 0d-9a73cf80-225b-4267-8788-c9d1dd6f49b1
Name: OracleFederation_Login
Desc: Automatically created by SHAREid. Do not modify.
Level: 1
```

```
Lib: smauthdir
Param:
Secret: {RC2}XCoISknMgVRXLjoLd/Fzkg==
IsTemplate: false
IsUsedbyAdmin: false
Type: 1
AllowSaveCreds: false
IsRadius: false
IgnorePwCheck: false

objectclass: Scheme
Oid: 0d-afefd918-eb06-4322-bb54-5095c2a62976
Name: OracleFederation_LoginNoPwd
Desc: Automatically created by SHAREid. Do not modify.
Level: 1
Lib: smanapi
Param:
Secret: {RC2}zlnvNoQWQLsgkpm5lDGtnA==
IsTemplate: false
IsUsedbyAdmin: false
Type: 15
AllowSaveCreds: false
IsRadius: false
IgnorePwCheck: false

objectclass: Agent
Oid: 01-989a3b03-5f9a-4714-9e4c-8ce2e08914af
Name: oif_agent
Desc: oif_agent
AgentType: 10-8d78bb96-ae15-11d1-9cdd-006008aac24b
RealmHintAttrId: 0

objectclass: TrustedHost
Oid: 24-e49e94f7-f914-49be-92ed-a10685759ad2
Name: oif_agent
Desc: oif_agent
IpAddr: 140.87.155.127
Secret: {RC2}4eRzjIbX82w6r2Ke+ph3KA==
Is4xHost: true

objectclass: Agent
Oid: 01-fe46c0bd-2802-4a3e-8dcf-a68a4f3eb40c
Name: smbridgeagent.oif1.staqk10vm4.us.oracle.com
Desc: Automatically created by SHAREid. Do not modify.
AgentType: 10-8d78bb96-ae15-11d1-9cdd-006008aac24b
RealmHintAttrId: 0

objectclass: TrustedHost
Oid: 24-ca92dc34-8ebc-4c83-9bbd-001c33542fb3
Name: smbridgeagent.oif1.staqk10vm4.us.oracle.com
Desc:
IpAddr: 140.87.155.130
Secret: {RC2}z3gdtk5parnuWkqt73fGPA==
Is4xHost: true

objectclass: AgentGroup
Oid: 02-70eb00c8-30f4-4f9e-8e46-722014a42e02
Name: OracleFederation
Desc: Automatically created by SHAREid. Do not modify.
AgentType: 10-8d78bb96-ae15-11d1-9cdd-006008aac24b
```

```
Agents: 01-fe46c0bd-2802-4a3e-8dcf-a68a4f3eb40c

objectclass: Domain
Oid: 03-1c7ac22e-6646-4c61-8f2f-6261a0ef3a92
Name: FederationWebServicesDomain
Desc:
Realms: 06-a0cf82a7-d831-453d-ac9e-8ed814f90369
UserDirectories: 0e-08c6cadb-e30b-4e06-9e2e-b3d7a866fab8
Rules: 0b-ebe4b3ce-5a34-4a64-b67b-d1e64dee414d,
0b-3d48caac-a7d5-4cf7-8722-ea9ed257fa42, 0b-7f5c5171-8af3-48b0-897c-89d02ac15d43
RuleGroups:
Responses:
ResponseGroups:
Policies: 04-3d844415-ee88-46f0-b646-cfbc57e248b8,
04-6d11edc8-b6f4-4522-a956-b3ad263705fb, 04-70d60545-5316-49c3-ac96-ae5c75b7efe8
Admins:
Variables: 1d-dad4d124-7518-4b67-b72c-b4ef76622eb3,
1d-98031368-1b9d-4d40-b572-91cef289f497
ActiveExprs: 1f-ec5691c8-b45c-4ed7-8d69-95e1caa6c8ef,
1f-bcf186ea-cd60-4e9c-96c3-5d3076090ce1
IsAffiliate: false
IMSEnvs:
Mode: 0

objectclass: Realm
Oid: 06-09c376b0-44f4-4aa1-b076-52fe1ca05309
DomainOid: 03-0eda0cd9-58e3-4df4-8eeb-c4cfbded874e
Name: OracleFederation_Login
Desc: Automatically created by SHAREid. Do not modify.
ResourceFilter: /OracleFederation/Login
Agent: 02-70eb00c8-30f4-4f9e-8e46-722014a42e02
AgentType: 10-8d78bb96-ae15-11d1-9cdd-006008aac24b
Scheme: 0d-9a73cf80-225b-4267-8788-c9d1dd6f49b1
ProcessAuthEvents: true
ProcessAzEvents: true
ProtectAll: true
SelfRegOid: 00-
AzUserDirOid: 00-
MaxTimeout: 0
IdleTimeout: 0
ParentRealmOid: 03-0eda0cd9-58e3-4df4-8eeb-c4cfbded874e
SyncAudit: false
Type: 0
SessionType: 0
SessionDrift: -1

objectclass: Realm
Oid: 06-e076704c-2d8e-4c10-83f0-6d67e2c38163
DomainOid: 03-0eda0cd9-58e3-4df4-8eeb-c4cfbded874e
Name: OracleFederation_LoginNoPwd
Desc: Automatically created by SHAREid. Do not modify.
ResourceFilter: /OracleFederation/LoginNoPwd
Agent: 02-70eb00c8-30f4-4f9e-8e46-722014a42e02
AgentType: 10-8d78bb96-ae15-11d1-9cdd-006008aac24b
Scheme: 0d-afefd918-eb06-4322-bb54-5095c2a62976
ProcessAuthEvents: true
ProcessAzEvents: true
ProtectAll: true
SelfRegOid: 00-
AzUserDirOid: 00-
```

```
MaxTimeout: 0
IdleTimeout: 0
ParentRealmOid: 03-0eda0cd9-58e3-4df4-8eeb-c4cfbded874e
SyncAudit: false
Type: 0
SessionType: 0
SessionDrift: -1

objectclass: Rule
Oid: 0b-0bac8120-dd43-45a4-badd-5f56df272c2a
DomainOid: 03-0eda0cd9-58e3-4df4-8eeb-c4cfbded874e
Name: OracleFederation_Login
Desc: Automatically created by SHAREid. Do not modify.
Realm: 06-09c376b0-44f4-4aa1-b076-52fe1ca05309
AgentType: 10-8d78bb96-ae15-11d1-9cdd-006008aac24b
Action: Get
Resource: *
Time: 00000000-00000000-7f7f7f7f7f7f7f7f7f7f7f7f7f7f7f7f7f7f7f7f7f7f7f7f7f7f7f7f
AllowAccess: true
RegularExpression: false
IsEnabled: true
ActiveExpr: 00-
Variables:

objectclass: Response
Oid: 07-7a3e908b-1292-43af-ba0d-52f85b3e2716
DomainOid: 03-0eda0cd9-58e3-4df4-8eeb-c4cfbded874e
Name: OracleFederation_UserAttributes
Desc: Automatically created by SHAREid. Do not modify.
AgentType: 10-8d78bb96-ae15-11d1-9cdd-006008aac24b
Attributes: 08-8d8cf013-e93f-4a06-83b5-7eb44432bfb7,
08-fef77ff7-038d-4a94-840a-c562f137da5d, 08-8cb3d975-56b1-4b0d-a742-a1c044624b28
AccessAccept: false
AccessReject: false
AccessChallenge: false
Type: 0

objectclass: ResponseAttr
Oid: 08-8d8cf013-e93f-4a06-83b5-7eb44432bfb7
DomainOid: 03-0eda0cd9-58e3-4df4-8eeb-c4cfbded874e
Response: 07-7a3e908b-1292-43af-ba0d-52f85b3e2716
AgentTypeAttr: 11-8d78bb90-ae15-11d1-9cdd-006008aac24b
Value: uid=<%userattr="uid"%>
TTL: 0
Flags: 0
ActiveExpr: 00-

objectclass: ResponseAttr
Oid: 08-fef77ff7-038d-4a94-840a-c562f137da5d
DomainOid: 03-0eda0cd9-58e3-4df4-8eeb-c4cfbded874e
Response: 07-7a3e908b-1292-43af-ba0d-52f85b3e2716
AgentTypeAttr: 11-8d78bb90-ae15-11d1-9cdd-006008aac24b
Value: mail=<%userattr="mail"%>
TTL: 0
Flags: 0
ActiveExpr: 00-

objectclass: ResponseAttr
Oid: 08-8cb3d975-56b1-4b0d-a742-a1c044624b28
DomainOid: 03-0eda0cd9-58e3-4df4-8eeb-c4cfbded874e
```

```
Response: 07-7a3e908b-1292-43af-ba0d-52f85b3e2716
AgentTypeAttr: 11-8d78bb90-ae15-11d1-9cdd-006008aac24b
Value: genUserID=<%userattr="genUserID"%>
TTL: 0
Flags: 0
ActiveExpr: 00-

objectclass: PolicyLink
Oid: 05-88999535-dc1e-46c0-8420-b22ae3feb56f
DomainOid: 03-0eda0cd9-58e3-4df4-8eeb-c4cfbded874e
Rule: 0b-75967ff0-bffd-4410-acc8-6cbe3e65c68f
Response: 00-
Policy: 04-98e5b0bc-f051-4447-8b99-08b1d7454115

objectclass: PolicyLink
Oid: 05-8c49d99a-c459-4b48-bd32-e6f755d3cd34
DomainOid: 03-0eda0cd9-58e3-4df4-8eeb-c4cfbded874e
Rule: 0b-2259776d-e839-4eb4-9b0e-97adf95ae19a
Response: 00-
Policy: 04-98e5b0bc-f051-4447-8b99-08b1d7454115

objectclass: UserPolicy
Oid: 0f-d1c1fe0a-526e-4471-93c3-abd40109eb2d
DomainOid: 03-0eda0cd9-58e3-4df4-8eeb-c4cfbded874e
FilterPath: o=company,c=us
FilterClass: organization
PolicyResolution: 5
PolicyFlags: 0
UserDirectory: 0e-0aec4e86-469e-400f-ab82-2d75685ce40b
Policy: 04-98e5b0bc-f051-4447-8b99-08b1d7454115

objectclass: Policy
Oid: 04-aa1d7959-82b9-4dca-9737-ff89f153d10e
DomainOid: 03-0eda0cd9-58e3-4df4-8eeb-c4cfbded874e
Name: OracleFederation_UserAttributes
Desc: Automatically created by SHAREid. Do not modify.
Time: 00000000-00000000-7f7f7f7f7f7f7f7f7f7f7f7f7f7f7f7f7f7f7f7f7f7f7f7f
PolicyLinks: 05-b09b8003-d82c-4586-8132-9c9bf2a40874
UserPolicies: 0f-c1147a2a-a47b-455d-a52f-19f8cf401961
AllowAccess: true
IsEnabled: true
IPAddress:
Type: 0
ActiveExpr: 00-
UserActiveExpr: 00-

objectclass: PolicyLink
Oid: 05-b09b8003-d82c-4586-8132-9c9bf2a40874
DomainOid: 03-0eda0cd9-58e3-4df4-8eeb-c4cfbded874e
Rule: 0b-0bac8120-dd43-45a4-badd-5f56df272c2a
Response: 07-7a3e908b-1292-43af-ba0d-52f85b3e2716
Policy: 04-aa1d7959-82b9-4dca-9737-ff89f153d10e

objectclass: UserPolicy
Oid: 0f-c1147a2a-a47b-455d-a52f-19f8cf401961
DomainOid: 03-0eda0cd9-58e3-4df4-8eeb-c4cfbded874e
FilterPath: (genuserid=*)
FilterClass: User Attribute
PolicyResolution: 3
PolicyFlags: 0
```

```
UserDirectory: 0e-0aec4e86-469e-400f-ab82-2d75685ce40b
Policy: 04-aa1d7959-82b9-4dca-9737-ff89f153d10e
```

Deploying Oracle Identity Federation with a Sun Java System Web Server

This section describes how to integrate the Sun Java System Web Server with Oracle Identity Federation in the following configuration:

- One Oracle Identity Federation instance configured as a Service Provider (SP) with an Oracle Access Manager back end, using any directory server; and
- A second Oracle Identity Federation instance configured as an Identity Provider (IdP), with any directory server or database as the back end.

This section contains the following topics:

Requirements

When using a proxy in front of Oracle Identity Federation, the host name and port number of the proxy server instance are used, respectively, as the host and port of the Oracle Identity Federation `fedadmin` administration console. Since the proxy hides only the `fedadmin` console, it is necessary to create a second instance of Sun Java System Web Server with a reverse proxy plugin to hide the application server console as well.

Configuring Oracle Identity Federation Without a Web Proxy Server

This configuration requires two separate procedures:

- Configuring the Identity Provider
- Configuring the Service Provider

Configuring the Identity Provider

Take the following steps to configure the Identity Provider (IdP):

1. Install an instance of Oracle Identity Federation.

   For installation details, see Chapter 3, Installing Oracle Identity Federation, in the *Oracle Identity Federation Administrator's Guide*.
2. Log in to the administration console at:

   http://`IdPhost:port`/fedadmin
3. Go to **Server Configuration** -> **General** -> **Server properties**
4. Enter the Server Name as the fully qualified hostname.
5. The Server Port and SOAP Port are the same as the `fedadmin` port.

   Do not modify any other information.

6. Go to **Server Properties** -> **Service Provider** -> **Global Settings**.
7. Enter the Provider URL as `http://fully-qualified-host:port/fed/sp`.
8. Go to **Server Properties** -> **Identity Provider** -> **Global Settings**.
9. Enter the Provider URL as `http://fully-qualified-host:port/fed/idp`
10. Enter the Common Domain Name in the format:

    .mycompany.com
11. Go to **IDM Data Store** -> **User Data Store**.
12. Select the active repository.
13. Enter the following repository parameters:

Field	Value
Connection URL	If LDAP, provide the URL as `ldap://host:port`
Bind DN	cn=directory manager and provide password in the password field
User ID Attribute	Login user id
User Description Attribute	Same as User ID Attribute
Person Object Class	As appropriate, for example inetorgperson
Base DN	Same as DN of searchbase

    Accept the default values for the remaining fields.
14. On the same page, go to the Federation Data Store and choose the active repository.
15. Enter the following repository parameters:

Field	Value
Connection URL	If LDAP, provide the URL as `ldap://host:port`
Bind DN	cn=directory manager and provide password in the password field
User Federation Record Context	Same as for searchbase

    Accept the default values for the remaining fields.
16. To collect the metadata, in the browser enter a URL of the form:

    `http://hostname/fed/idp/metadatav11`, or

    `http://hostname/fed/idp/metadatav12`, or

    `http://hostname/fed/idp/metadatav20`
17. Save the metadata to a file.

Configuring the Service Provider

Take the following steps to configure the Service Provider (SP):

1. Install Oracle Access Manager.
2. Install Oracle Identity Federation. Log in to the administration console at:

`http://host:port/fedadmin`

3. Go to **Server Configuration** -> **General** -> **Server properties**.
4. Enter the Server Name as the fully qualified hostname.

   For example:

   `host.mycompany.com:port`
5. The Server Port and SOAP Port are the same as the `fedadmin` port.

   Do not modify any other information.
6. Go to **Server Properties** -> **Service Provider** -> **Global Settings**.
7. Enter the Provider URL as `http://fully-qualified-host:port/fed/sp`.
8. Go to **Server Properties** -> **Identity Provider** -> **Global Settings**.
9. Enter the Provider URL as `http://fully-qualified-IdPhost:port/fed/idp`
10. Enter the Common Domain Name in the format:

    `.mycompany.com`
11. Install the Oracle Access Manager Access Server SDK at *OIF_Install_Dir*`/fed/shareid`. The Access Server SDK is installed at *OIF_Install_Dir*`/fed/shareid/AccessServerSDK`.
12. Use Oracle Access Manager to configure an AccessGate instance for Oracle Identity Federation.

    Note: The Access management Service should be ON for the AccessGate Configuration.
13. At the Access System Console, create a host identifier with name `Fed HostID` in the format:

    `fully-qualified-SPhost:port`
14. Create a policy domain to protect any test resource using the Fed SSO - SAML 2.0/Liberty 1.x authentication Scheme.
15. In the Oracle Identity Federation administration console, go to **IDM Data Store** -> **User Data Store**.
16. Select Oracle Access Manager as the active repository.
17. Enter the following repository parameters:

Field	Value
Connection URL	If LDAP, provide the URL as `ldap://host:port`
Bind DN	cn=directory manager and provide password in the password field
User ID Attribute	Login user id
User Description Attribute	Same as User ID Attribute
Person Object Class	As appropriate, for example inetorgperson
Base DN	Same as DN of searchbase

Accept the default values for the remaining fields.

18. For the Oracle Access Manager configuration parameters, provide following information:

Field	Value
Master Admin Login ID	admin userid
Master Admin Password	admin user password
Authorization result for unprotected resources	Allow
Oracle Access Manager Cookie Domain	Fully qualified host name
Basic Authentication Scheme Name	Oracle Access and Identity Basic Over LDAP (for Oracle Access Manager 10.1.4 and above) or Basic Over LDAP (for 7.0.4).

Click **Apply**.

19. On the same page, go to **Federation Data Store**.
20. Select the active repository.
21. Enter the following repository parameters:

Field	Value
Connection URL	If LDAP, provide the URL as `ldap://host:port`
Bind DN	cn=directory manager and provide password in the password field
User Federation Record Context	cn=fed, *searchbase*

Accept the default values for the remaining fields.

22. Go to **Server Configuration** -> **Circle of Trust**.
23. From the **Add Trusted Provider** page, upload the metadata that you collected for IdP and refresh the server.
24. Once the IdP appears in the list of Identity Providers, go to the main menu, go to Identity Federation, and make sure that the IdP URL shows up in the Identity Providers list.
25. Go to **Server Configuration** -> **Service Provider** -> **Global Settings**.
26. In the drop-down list for the Default SSO Identity Provider, select the URL of the IdP provider.
27. Save the changes and refresh the server.
28. To collect the metadata, in the browser enter a URL of the form:

    `http://`*`hostname`*`/fed/sp/metadatav11`, or

    `http://`*`hostname`*`/fed/sp/metadatav12`, or

    `http://`*`hostname`*`/fed/sp/metadatav20`

29. Save the metadata to a file.
30. Upload this metadata to the IdP.

Configuring Oracle Identity Federation Behind a Web Proxy Server

Follow the procedure defined in "Configuring Oracle Identity Federation Without a Web Proxy Server" on page 4-22, with these modifications:

- Change the configuration URLs to their respective proxy server URLs.
- In the Oracle Identity Federation administration console, go to **SAML 1.x/WS-Fed - > Domains - > MyDomain** and update the SAML 1.x/WS-Fed configuration URLs:
  - Error URL
  - Transfer URL
  - Responder URL
  - SourceID (blank to regenerate for new ResponderURL)
  - Identity Realm Secure Token Service (STS) URL
  - Receiver URL
  - Resource Realm STS URL

  See "Configuring SAML 1.x and WS-Federation Properties" on page 6-81 for information about these URLs.
- Collect the metadata using the proxy URLs, not actual URLs, then upload the metadata.
- At the Access System Console, create a host identifier in the format:

  `proxy-host:port`

  and change the challenge redirect of the authentication scheme to *`proxy-host:port`*.

Integrating Oracle Identity Federation with OracleAS Single Sign-On

To integrate Oracle Identity Federation with an Oracle Single Sign-On back end, take these additional steps:

1. Update the OracleAS Single Sign-On partner application:
   - Go to the Oracle Single Sign-On administration console (`http://osso_host:osso_port/pls/orasso`).
   - Click **Single Sign-On Administration**.
   - Click **Administer Partner Applications**.
   - Choose the partner application referencing the Oracle Identity Federation server, and edit the application configuration.
   - Replace the http(s), hostname and port number by the proxy's values for Home URL, Success URL, and Logout URL.
   - Click **OK**.
2. Update the `policy.properties` file:
   - Open the `ORACLE_HOME/sso/conf/policy.properties` file in the OracleAS Single Sign-On deployment.

- Replace the http(s), hostname and port number by the proxy's value for the `SASSOAuthnUrl` and `SASSOLogoutUrl` entries.
- Save and close the file.

3. Restart the `OC4J_SECURITY` instance of the OracleAS Single Sign-On deployment by using the command:

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

Sample Configuration Files

This section provides samples of the `obj.conf` and `magnus.conf` configuration files.

Sample obj.conf File

```
<Object name="default">
AuthTrans fn="match-browser" browser="*MSIE*" ssl-unclean-shutdown="true"
NameTrans fn="assign-name" from="/*" name="serverexample"
NameTrans fn="ntrans-j2ee" name="j2ee"
NameTrans fn=pfx2dir from=/mc-icons dir="/home/panacea/SunOne6.1/ns-icons"
name="es-internal"
NameTrans fn=document-root root="$docroot"
PathCheck fn=unix-uri-clean
PathCheck fn="check-acl" acl="default"
PathCheck fn=find-pathinfo
PathCheck fn=find-index index-names="index.html,home.html,index.jsp"
ObjectType fn=type-by-extension
ObjectType fn=force-type type=text/plain
Service method=(GET|HEAD) type=magnus-internal/imagemap fn=imagemap
Service method=(GET|HEAD) type=magnus-internal/directory fn=index-common
Service method=(GET|HEAD|POST) type=*~magnus-internal/* fn=send-file
Service method=TRACE fn=service-trace
Error fn="error-j2ee"
AddLog fn=flex-log name="access"
</Object>

<Object name="j2ee">
Service fn="service-j2ee" method="*"
</Object>

<Object name="cgi">
ObjectType fn=force-type type=magnus-internal/cgi
Service fn=send-cgi user="$user" group="$group" chroot="$chroot" dir="$dir"
   nice="$nice"
</Object>

<Object name="es-internal">
PathCheck fn="check-acl" acl="es-internal"
</Object>

<Object name="send-compressed">
PathCheck fn="find-compressed"
</Object>

<Object name="compress-on-demand">
Output fn="insert-filter" filter="http-compression"
</Object>

# Execute these instructions for any resource with the assigned name
# "server.example.com"
<Object name="serverexample">
```

```
# Proxy the requested resource to the URL
# "http://server.example.com:8080"
Service fn="service-passthrough" servers="http://flagstaff.persistent.co.in:7777"
</Object>
```

Sample magnus.conf File

```
#
# The NetsiteRoot, ServerName, and ServerID directives are DEPRECATED.
# They will not be supported in future releases of the Web Server.
NetsiteRoot /home/panacea/SunOne6.1
ServerName calgary
ServerID https-oif_idp_flagstaff
#
RqThrottle 128
DNS off
Security off
PidLog /home/panacea/SunOne6.1/https-oif_idp_flagstaff/logs/pid
User panacea
StackSize 131072
TempDir /tmp/https-oif_idp_flagstaff-65cd125c

Init fn=flex-init access="$accesslog" format.access="%Ses->client.ip%
   - %Req->vars.auth-user% [%SYSDATE%] \"%Req->reqpb.clf-request%\"
   %Req->srvhdrs.clf-status% %Req->srvhdrs.content-length%"
Init fn="load-modules"
shlib="/home/panacea/SunOne6.1/bin/https/lib/libj2eeplugin.so" shlib_
flags="(global|now)"
Init fn="load-modules"
shlib="/home/panacea/SunOne6.1/bin/https/passthrough/plugins/passthrough
   /libpassthrough.so"
```

Configuring Oracle Identity Federation to Use IBM Tivoli Directory Server as the Data Store

This section describes how to configure Oracle Identity Federation to use IBM Tivoli Directory Server (IBM TDS) 6.0 in the back end, as either a federation data store or a user data store. Topics include:

- Prerequisites
- Configuring IBM Tivoli Directory Server as the Federation Data Store for IDP or SP
- Configuring IBM Tivoli Directory Server as the User Data Store for an IdP

Prerequisites

Check that the following prerequisites are met before setting up the IBM TDS data store:

1. Install Oracle Identity Federation.

   See Also: Chapter 3, "Installing Oracle Identity Federation"

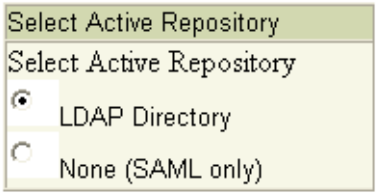
2. Install and configure IBM TDS v6.0.

   Externally load the federation schema into IBM TDS to create federation records; for this LDAP schema update, use the `$ORACLE_HOME/setup/ldap/userFedSchemaTivoli.ldif` file.

Refer to "Federation Data Store" on page 2-17, in the discussion titled "A Note About LDAP Schema" for details on how to configure the LDAP schema.

Configuring IBM Tivoli Directory Server as the Federation Data Store for IDP or SP

In the Oracle Identity Federation Administration Console, navigate to **IdM Data Stores** -> **Edit Federation Data Store**.



See Also: "Edit Federation Data Store" on page 6-64 for details about field configuration.

Select LDAP Directory for the active repository.

Provide these repository parameters:

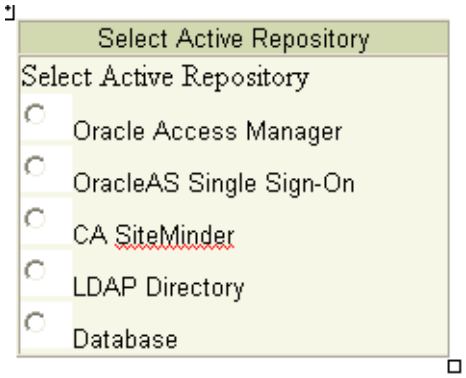
- Connection URL(s) - This is the Connection URL; enter a space-delimited list of IBM Tivoli Directory Server URLs with hostname and port.
- Bind DN - This is the DN used by the Oracle Identity Federation server to connect to the IBM Tivoli Directory Server.
- Password - Enter the administrator account password to use to connect to the LDAP server.
- User Federation Record Context - Enter the node under which all federation records for this Oracle Identity Federation server will be stored.
- LDAP Container Object Class - Enter the type of User Federation Record Context that Oracle Identity Federation should use when creating the LDAP container. For example, for the IBM Tivoli Directory Server this field can be left empty.

Note: The User Federation Record Context must be compatible with the LDAP Container Object Class, as explained in "Edit Federation Data Store" on page 6-64.

- Unique Federation ID Attribute - Enter the LDAP attribute to be used to uniquely identify a federation record. For example, for IBM Tivoli Directory Server this field can be left empty.
- Maximum Connections - Enter the maximum number of concurrent connections that can be made by Oracle Identity Federation to the IBM Tivoli Directory Server.
- Connection Wait Timeout (secs) - Enter the maximum number in seconds to wait until a connection is available, when the maximum number of connections opened by Oracle Identity Federation to the IBM Tivoli Directory Server has been reached.

Configuring IBM Tivoli Directory Server as the User Data Store for an IdP

In the Oracle Identity Federation Administration Console, navigate to **IdM Data Stores** -> **Edit User Data Store**.



See Also: "Edit User Data Store" on page 6-67 for details about field configuration.

Select LDAP Directory from the active repository.

Provide these repository parameters:

- Connection URL(s): Enter the Connection URL as a space-delimited list of IBM Tivoli Directory Server URLs with hostname and port.
- Bind DN: This is the DN used by the Oracle Identity Federation server to connect to IBM Tivoli Directory Server.
- Password: Enter the administrator account password to use to connect to the data store.
- User ID Attribute: Enter the attribute name to use to map users during lookups or authentication procedures. For example, for the
- IBM Tivoli Directory Server this field can be set to `uid`.
- User Description Attribute: This field references the user attribute to use as a human readable federation owner identifier. This information will be stored in the federation record. For example, for the IBM Tivoli Directory Server this field can be set to `uid`.
- Person Object Class: Enter the LDAP object class representing a user in the IBM Tivoli Directory Server. For example, for the IBM Tivoli Directory Server this field can be set to `inetOrgPerson`.
- Base DN: Enter the node under which the IBM Tivoli Directory Server user search will be performed. For example: `dc=us,dc=oracle,dc=com`.
- Connection Wait Timeout (secs): Enter the maximum number in seconds to wait until a connection is available, when the maximum number of concurrent connections made by Oracle Identity Federation to the IBM Tivoli Directory Server has been reached.

Integrating with Third-Party Identity & Access Management Modules

Oracle Identity Federation provides cross-domain Single Sign-On using standard protocols such as SAML, Liberty, and WS-Federation.

Out of the box, Oracle Identity Federation integrates with several Identity and Access Management (IAM) products, including:

- Oracle AS Single Sign-On
- Oracle Access Manager

- CA eTrust SiteMinder

Oracle Identity Federation also provides a framework for developers to integrate custom or third-party Identity and Access Management solutions. This section explains the components of this framework, and shows you how to configure and integrate a custom IAM solution into the framework.

> **Note:** Oracle strongly discourages users from deploying any applications on the OC4J_FED J2EE container instance other than the custom integration and authentication described in this section, because doing so introduces potential security risks. Extraneous applications deployed in the OC4J_FED container can potentially affect the security of the federation server by allowing rogue software to change the behavior of the server flows.

This section includes the following topics related to custom engine implementation:

- Architecture and Flows
- Creating a Custom Authentication Engine
- Creating a Custom SP Integration Engine
- Logout
- The GenericSPCookieProvider Example

Architecture and Flows

At runtime, Oracle Identity Federation interacts with two types of external modules: a user data store, and an Identity and Access Management system.

Oracle Identity Federation works with the user data store to:

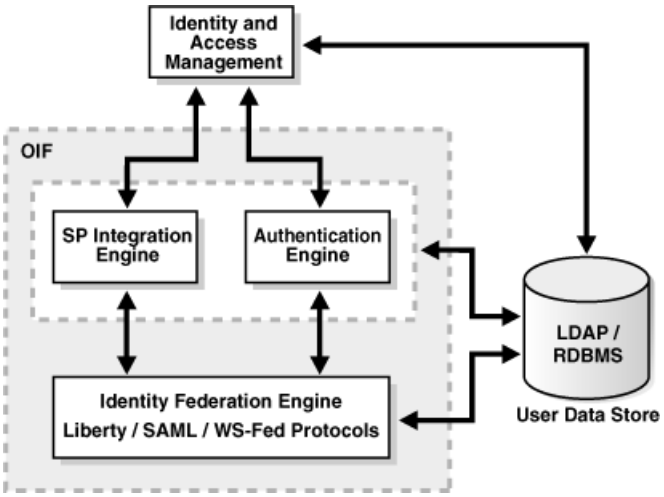
- locate a user after local authentication
- locate a user after processing an incoming SAML Assertion
- retrieve attributes for a specific user

The Identity and Access Management (IAM) system provides access control for protected resources. Oracle Identity Federation, as the federation server, interacts with IAM to:

- authenticate a user when the server needs to obtain a user's local identity. This operation might occur when the server acts as an IdP, or when the server, as an SP, needs to authenticate the user during initial account linking/federation creation.
- create an authenticated session for a specific user when the server processes an incoming SAML Assertion and asserts the user's identity to the IAM system
- process logout flows; in this case, the federation server invokes IAM capabilities to log the user out of the system

Architecture Figure 4–1 depicts the different external and internal modules of an Oracle Identity Federation deployment, and how they interact at runtime:

Figure 4–1 Oracle Identity Federation Module Interactions



For the sake of this discussion, the Oracle Identity Federation server is depicted as three internal modules:

- the Identity Federation Engine, which is responsible for creating and processing SAML messages such as AuthnRequest, Assertion, and Logout messages.

  This module:

  - works with the user data store when processing SAML messages;
  - interacts with the Authentication Engine when it is necessary for a user to be locally identified; and
  - interacts with the SP Integration Engine when the user is redirected to the IAM component after processing an Assertion.

- the Authentication Engine, which is responsible for processing requests from the Federation Engine to authenticate users. This module interacts with the IAM component and the user data store to authenticate the user and retrieve the unique identifier Oracle Identity Federation uses to reference the user.

  This module can be invoked in either IdP or SP mode when local authentication is required.

  After authenticating the user, the Authentication Engine sends the authentication information, such as the user's identifier, the time of authentication, and other data to the Federation Engine.

- After successfully processing an incoming SAML assertion and locating the user referenced by the assertion, the Federation Engine instructs the SP Integration Engine to create an authenticated session for that user in the IAM domain. It passes the necessary information (user's identifier, authentication time, and so on) to the SP Integration Engine, which interacts with the IAM server to create the session.

Authentication Engine Processing Flow The Authentication Engine included with Oracle Identity Federation contains a servlet to process requests from server to authenticate a user, and several internal plug-ins that allow it to interact with the various IAM servers supported out of the box (see the introduction to this discussion on page 4-35).

Here is a step-by-step description of how the Authentication Engine interacts with the other federation server components in a typical user flow:

1. The user accesses Oracle Identity Federation for an SSO operation (in either SP or IdP mode).
2. An internal process in the server determines that the user needs to be identified.
3. The federation server redirects the user to itself in order to drop the existing query `String` parameters and replace them with Oracle Identity Federation query `String`s. A typical new query `String` includes these parameters:
   - `doneURL` - The relative URL where the user needs to be forwarded back once the Authentication Engine identifies the user, when the `getUsrSess` parameter value is not true.
   - `authnMech` - The authentication mechanism requested by the federation server to use to authenticate the user.
   - `refID` - An identifier referencing data in the federation server. When present, this parameter must be passed back to Oracle Identity Federation when the Authentication Engine forwards the user.
   - `getUsrSess` - A boolean parameter indicating whether the user should be redirected to the WS-Fed/SAML 1.x protocol engines instead of to the `doneURL` parameter.
4. The Federation Engine internally forwards the user's request to the `/sso/authn` URL (the context is `/sso`, and the forward URL is `/authn`). On the HttpServletRequest, a `String` attribute called `oracle.security.sso.sasso.authn.dyna_mode` is set to the value `idp` to indicate that it is requesting a user authentication.
5. The Authentication Engine processes the incoming request. It has access to the following information:
   - `doneURL`, `authnMech`, `refID` , and `getUsrSess` as query parameters
   - `oracle.security.sso.sasso.authn.dyna_mode` as an HttpServletRequest attribute
   -
6. The Authentication Engine interacts with the IAM component and may challenge the user for credentials. After successful authentication, it may set a cookie (for example, to maintain the authenticated session with the IAM server and/or the target application).
7. If the `getUsrSess` parameter is missing or set to `false`, the Authentication Engine forwards the user back to the Liberty 1.x / SAML 2.0 Federation Engine at the `doneURL` in the federation context (the context is `/fed` and the forward URL is the `doneURL` query parameter received earlier).

   If the `getUsrSess` parameter value is `true`, the Authentication Engine forwards the user back to the WS-Fed / SAML 1.x Federation Engine into the `/shareid` context, at the `/saml/ObSAMLLoginService` relative URL.

   The Authentication Engine sets these HttpServletRequest attributes:
   - A `String` attribute, `oracle.security.sso.sasso.uid`, containing the user's identifier. The Federation Engine uses this identifier to locate the user based on the user record attribute set in the Oracle Identity Federation Administration Console (specifically, the User ID field on the User Data Store page).
   - A `String` attribute, `oracle.security.sso.sasso.refID`, containing the `refID` query parameter value sent earlier.

- A `String` attribute, `oracle.security.sso.sasso.authnMech`, containing the oracle authentication mechanism identifier that references the method used to authenticate the user. At this time, the value can only be set to `oracle:fed:authentication:password-protected`.
- A `date` attribute, `oracle.security.sso.sasso.authnInst`, containing the instant when the user was authenticated.

Note: This attribute contains an object of class `java.util.Date`.

8. Oracle Identity Federation:
   - processes the incoming request
   - retrieves the data embedded as attributes in the HttpServletRequest
   - locates the user in the user data store
   - creates a session for the user
   - sets a cookie, and
   - resumes the SSO operation.

SP Integration Engine Processing Flow The SP Integration Engine included with Oracle Identity Federation consists of a servlet that processes requests from the server to create a user authenticated session at the IAM server. The engine includes several internal plug-ins that allow it to interact with different IAM Servers.

Here is a step-by-step description of how the SP Integration Engine interacts with the other federation server components in a typical user flow:

1. The user triggers a Federation SSO operation with a remote IdP. The SSO is triggered either at the IdP (IdP-initiated SSO), or at the Oracle Identity Federation SP, by requesting the `/fed/sp/initiatesso` URL with these query parameters:
   - `providerid` - the IdP's provider ID [REQUIRED]
   - `returnurl` - the URL to redirect the user to after successful Federation SSO [OPTIONAL]

   Note: Refer to "SP-initiated SSO with Liberty 1.x/SAML 2.0" on page 7-10 for a complete description of these parameters.

2. The IdP redirects the user, with an assertion, to the federation server acting as an SP.
3. The server processes the assertion and locates the user in the user data store. The user is now authenticated at the federation server.
4. The federation engine internally forwards the user to the `/sso/authn` URL and passes the following data as HttpServletRequest attributes:
   - the user's Identifier
   - the time of authentication
   - the expiration time of the authenticated session. This is the lesser of the Oracle Identity Federation Session Timeout and the session expiration sent by the IdP in the assertion

- the URL to which to redirect the user

5. The SP Integration Engine interacts with the IAM server to create an authenticated session for the user. The session is based on the data received from Oracle Identity Federation.
6. The SP Integration Engine redirects the user to the final target URL.

Requirements Oracle Identity Federation's design is consistent with certain requirements regarding authentication operations and SP integration (where a user session is created at the IAM server). Consequently, you must meet the following requirements when implementing a custom authentication engine or an SP Integration Engine:

- The Authentication Engine, the SP Integration Engine, the Identity Federation Engine and the IAM server must use the same user data store as the user repository. This store contains the user data used to look up and authenticate users.
- The Authentication Engine and the SP Integration Engine must include a Java Servlet /JSP.
- The data exchanges between OIF and the Authentication/SP Integration Engines are done via internal HTTP request forwarding. This is actually an internal API call between the modules that relies on the J2EE servlet framework via the HTTP protocol.
- A logout service needs to be implemented and made available to the Authentication Engine and/or the SP Integration Engine. This logout service must be published as Servlet/JSP.

If both a customized Authentication Engine and an SP Integration Engine are implemented with Oracle Identity Federation, you must configure the user data store for LDAP or RDBMS in Standalone mode. This is necessary so that Oracle Identity Federation does not attempt to interact with an IAM Server (Oracle SSO, Oracle Access Manager or CA SiteMinder), for example at initialization time.

Creating a Custom Authentication Engine

This section explains how to plan, develop, and implement a custom authentication engine.

Planning a Custom Authentication Engine Creating a customized authentication engine involves:

- creating a service that will process incoming requests from Oracle Identity Federation
- implementing a module to authenticate a user
- creating a service that forwards the user to the federation server with the required information
- deciding whether or not the Authentication Engine will set a cookie after authenticating a user. If yes, the authentication module must be integrated into the logout process (see "Logout" on page 4-49)
- packaging the services and module into a web application, and deploying the application to the OC4J_FED instance of the Oracle Application Server where Oracle Identity Federation is running

- modifying Oracle Identity Federation configuration files to reference the new Authentication Engine.

> **Note:** This step requires an OC4J_FED restart.

- ensuring that the user repository used by the Authentication Engine is the same as the user data store configured in the Oracle Identity Federation Administration Console and used by the federation server

Developing and Implementing the Authentication Module Several aspects of module development are explained here.

URLs

Communication between the Federation Engine and the Authentication Engine occurs through internal servlet forwards that are equivalent to API calls. These forwards use the following J2EE API:

```
ServletContext.getContext(String contextPath)
   .getRequestDispatcher(String relativePath)
   .forward(HttpServletRequest request,
      HttpServletResponse response)
```

where:

- `contextPath` is the root context path of the web application. For example, the `contextPath` of Oracle Identity Federation is either `/fed` or `/shareid`.
- `relativePath` is the service URL to which to forward the user; it is relative to the `contextPath`. For example, after authenticating a user, the Authentication Engine uses `/user/loginsso` or `/saml/ObSAMLLoginService` as the `relativePath` when forwarding the user.

The Authentication Engine needs to be aware that the `contextPath` of the Oracle Identity Federation server is `/fed` or `/shareid`.

Additionally, Oracle Identity Federation needs to be aware of the `contextPath` and the `relativePath` of the new Authentication Engine. This is the URL that will process authentication requests issued by the federation server.

Modify the following files to configure these URLs:

```
$ORACLE_HOME/fed/conf/idpmanager/authn-mappings.xml
$ORACLE_HOME/fed/conf/spmanager/authn-mappings.xml
$ORACLE_HOME/fed/conf/usermanager/authn-mappings.xml
```

Each file needs the same set of changes:

- In the `<authn-mapping>` element containing the `<authn-method>` whose value equals `sasso_login_idp_context`, modify the value of the corresponding `<authn-screen>` to the `contextPath` of the new Authentication Engine; by default, that value is set to `/sso`.
- In the `<authn-mapping>` element containing the `<authn-method>` whose value equals `sasso_login_idp_url`, modify the value of the corresponding `<authn-screen>` to the `relativePath` of the new Authentication Engine; by default, that value is set to `/authn`.

The SAML 1.x and WS-Fed protocols also need to be aware of the `contextPath` and the `relativePath` of the new authentication engine. To configure these URLs,

modify the $ORACLE_HOME/fed/shareid/oblix/config/shareid-config.xml file like this:

- In the <LoginConfig> XML element, locate the <AuthnMethod> element whose Name attribute equals Sasso, and set the LoginURL attribute to the concatenation of the contextPath and the relativePath.

> **Note:** Due to a limitation for the SAML 1.x/WS-Fed protocols in Oracle Identity Federation, the relativePath can contain only one "/" character; there are no restrictions on the contextPath. For example, contextPath can be set to /path1/path2/path3, while relativePath can only be set to /path4; a value of /path5/path6 for the relativePath is invalid.

Restart the OC4J_FED instance after changing the authn-mappings.xml files.

Implementing the Service

This section describes the roles that are played by the authentication engine, and the processing tasks that the service must be able to handle for a successful implementation.

The Authentication Engine needs to:

- process requests from the Federation Engine
- forward the user to the federation server after a successful authentication

When processing authentication requests from the server, the engine must process the following incoming data:

- doneURL query parameter - The relative URL to which the user must be forwarded, after being identified by the Authentication Engine, when the getUsrSess parameter value is not true.
- authnMech query parameter - The authentication mechanism requested by the federation server to be used to authenticate the user. In this release, it will always be oracle:fed:authentication:password-protected.
- refID query parameter - An identifier referencing data in the federation server. When present, this parameter must be passed back to Oracle Identity Federation when the Authentication Engine forwards the user.
- getUsrSess - A boolean parameter indicating whether the user should be redirected to the WS-Fed/SAML 1.x protocol engines instead of to the doneURL parameter.
- oracle.security.sso.sasso.authn.dyna_mode HttpServletRequest attribute. This value is always set to idp when local authentication is invoked.

After successful authentication, the engine must forward the user to the federation server. If the getUsrSess parameter was missing or its value is not true, the rootContext of the federation engine is /fed, and the relativePath for this release is /user/loginsso; this relativePath parameter is also stored in the doneURL query parameter. If the getUsrSess parameter is present and its value is true, the rootContext of the federation engine is /shareid, and the relativePath for this release is /saml/ObSAMLLoginService.

Oracle Identity Federation expects this data when processing the internal forward:

- `oracle.security.sso.sasso.uid` - HttpServletRequest attribute of type `String` containing the user's identifier.
- `oracle.security.sso.sasso.refID` - HttpServletRequest attribute of type `String` containing the `refID` query parameter value sent earlier by Oracle Identity Federation.
- `oracle.security.sso.sasso.authnMech` - HttpServletRequest attribute of type `String` containing the Oracle authentication mechanism identifier that references the method used to authenticate the user. In this release, this value must be set to `oracle:fed:authentication:password-protected`.
- `oracle.security.sso.sasso.authnInst` - HttpServletRequest attribute of type `java.util.Date` containing the instant when the user was authenticated.

Here are some additional implementation requirements:

- If the service needs to set any cookies, perform this operation before forwarding the user to the federation server.
- Set the cookie path value to "/". This is required because of the internal forwards between the Oracle Identity Federation web application and the Authentication Engine web application; the user's browser needs to send the cookies related to the Authentication Engine, even when it is accessing only the federation server. This way, at an internal forward from the federation server to the Authentication Engine, the cookies set by the engine are available in the HTTP Request.

Sample Authentication Module for Oracle AS Single Sign-On Integration This section describes how to integrate a custom Authentication Engine with OracleAS Single Sign-On.

Setup

In this example, the Application Server where Oracle Identity Federation is running has been integrated with the OracleAS Single Sign-On server, and the SSO module statically protects the `/engine/forward.jsp` URL.

Additionally, the user data store configured in the Oracle Identity Federation Administration Console references the Oracle Internet Directory server used by OracleAS Single Sign-On.

> **See Also:** "Deploying Oracle Identity Federation with OracleAS Single Sign-On" on page 4-2 for more information on SSO integration.

Packaging

The Authentication Engine consists of a Web application with a root context set to `/engine`, and contains two JSP pages:

- `authentication.jsp` which processes the incoming request from the Oracle Identity Federation server
- `forward.jsp` which is protected by OracleAS Single Sign-On, and which forwards the user back to the federation server with the required data.

Implementation of authentication.jsp

```
<%@page buffer="5" autoFlush="true" session="false"%>
<%@page language="java" import="java.net.*"%>
<%
response.setHeader("Cache-Control", "no-cache");
response.setHeader("Pragma", "no-cache");
response.setHeader("Expires", "Thu, 29 Oct 1969 17:04:19 GMT");
```

```
String doneURL = request.getParameter("doneURL");
String authnMech = request.getParameter("authnMech");
String refID = request.getParameter("refID");
String getUserSession = request.getParameter("getUsrSess");
String authnMode =
(String)request.getAttribute("oracle.security.sso.sasso.authn.dyna_mode");

if (authnMode != null && !"idp".equals(authnMode))
   throw new ServletException("Incorrect authn mode");

String redirectURL = "/engine/forward.jsp?doneURL=" +
   (doneURL != null ? URLEncoder.encode(doneURL) : "")  + "&refID=" +
   (refID != null ? URLEncoder.encode(refID) : "")  +
   (getUserSession != null && getUserSession.length() > 0 ? "&getUsrSess="  +
URLEncoder.encode(getUserSession) : "");

response.sendRedirect(redirectURL);
%>
```

Implementation of forward.jsp

```
<%@page buffer="5" autoFlush="true" session="false"%>
<%@page language="java" import="java.util.*"%>
<%
response.setHeader("Cache-Control", "no-cache");
response.setHeader("Pragma", "no-cache");
response.setHeader("Expires", "Thu, 29 Oct 1969 17:04:19 GMT");

String doneURL = request.getParameter("doneURL");
String refID = request.getParameter("refID");
String getUserSession = request.getParameter("getUsrSess");
String userID = request.getRemoteUser();
String authnMethod = "oracle:fed:authentication:password-protected";
Date now = new Date();

request.setAttribute("oracle.security.sso.sasso.uid", userID);
request.setAttribute("oracle.security.sso.sasso.refID", refID);
request.setAttribute("oracle.security.sso.sasso.authnMech", authnMethod);
request.setAttribute("oracle.security.sso.sasso.authnInst", now);

if ("true".equals(getUserSession))
   request.getSession().getServletContext().getContext("/shareid").getRequestDispa
tcher("/saml/ObSAMLLoginService").forward(request, response);
else
   request.getSession().getServletContext().getContext("/fed").getRequestDispatche
r(doneURL).forward(request, response);
%>
```

Oracle Identity Federation Configuration Files

Modify these files to define the Authentication Engine `contextPath` and `relativePath`:

```
$ORACLE_HOME/fed/conf/idpmanager/authn-mappings.xml
$ORACLE_HOME/fed/conf/spmanager/authn-mappings.xml
$ORACLE_HOME/fed/conf/usermanager/authn-mappings.xml
```

Each file needs the same set of changes:

```
<authn-mapping>
        <authn-method>sasso_login_idp_context</authn-method>
        <authn-screen>/engine</authn-screen>
    </authn-mapping>
    <authn-mapping>
        <authn-method>sasso_login_idp_url</authn-method>
        <authn-screen>/authentication.jsp</authn-screen>
    </authn-mapping>
```

The `$ORACLE_HOME/fed/shareid/oblix/config/shareid-config.xml` file is modified to define the Authentication Engine URL, and to set the `useLocalConfig` attribute to `true` to force changes to be persisted at restart:

```
<SHAREidConfiguration … useLocalConfig="true">
…
   <AuthMethod Name="Sasso"
      LoginURL="/engine/authentication.jsp" LogoutURL="/sso/logout" />
</SHAREidConfiguration>"
```

Logout

Since the OracleAS Single Sign-On framework sets cookies in the user's browser, the authentication engine should be integrated into the logout flow (see "Logout" on page 4-49).

Sample Authentication Module for LDAP Integration This section shows how to integrate a customized Authentication Engine with a standalone LDAP server.

Setup

The user data store configured in the Oracle Identity Federation Administration Console references the LDAP server used by the Authentication Engine.

Packaging

The Authentication Engine consists of a Web application with a root context set to `/engine`, and contains two JSP pages:

- `loginpage.jsp`, which processes the incoming request from the federation server, and displays the login page.
- `ldapforward.jsp`, which authenticates the user's credentials against the LDAP server; upon success it forwards the user to the federation server.

Implementation of loginpage.jsp

```
<%@page buffer="5" autoFlush="true" session="false"%>
<%@page language="java" import="java.net.*"%>
<%
response.setHeader("Cache-Control", "no-cache");
response.setHeader("Pragma", "no-cache");
response.setHeader("Expires", "Thu, 29 Oct 1969 17:04:19 GMT");

String doneURL = request.getParameter("doneURL");
String authnMech = request.getParameter("authnMech");
String refID = request.getParameter("refID");
String getUserSession = request.getParameter("getUsrSess");
String authnMode =
(String)request.getAttribute("oracle.security.sso.sasso.authn.dyna_mode");

if (authnMode != null && !"idp".equals(authnMode))
```

```
   throw new ServletException("Incorrect authn mode");

String postURL = "/engine/ldapforward.jsp?doneURL=" +
   (doneURL != null ? URLEncoder.encode(doneURL) : "")  + "&refID=" +
   (refID != null ? URLEncoder.encode(refID) : "")  +
   (getUserSession != null && getUserSession.length() > 0 ? "&getUsrSess="  +
URLEncoder.encode(getUserSession) : "");

String msg = request.getParameter("message");
%>
<HTML>
<BODY>
<FORM action="<%=postURL%>" method="POST">
<% if(msg != null && msg.length() > 0) { %> <%=msg%><BR/> <%}%>
Username: <INPUT type="text" name="username"/><BR/>
Password: <INPUT type="password" name="password"/><BR/>
<INPUT type="submit" value="Submit"/>
</FORM>
</BODY>
</HTML>
```

Implementation of forward.jsp

```
<%@page buffer="5" autoFlush="true" session="false"%>
<%@page language="java" import="java.util.*, javax.naming.*,
javax.naming.directory.*, java.net.*"%>
<%
response.setHeader("Cache-Control", "no-cache");
response.setHeader("Pragma", "no-cache");
response.setHeader("Expires", "Thu, 29 Oct 1969 17:04:19 GMT");

String doneURL = request.getParameter("doneURL");
String refID = request.getParameter("refID");
String getUserSession = request.getParameter("getUsrSess");
String userID = request.getParameter("username");
String password = request.getParameter("password");
String authnMethod = "oracle:fed:authentication:password-protected";
Date now = new Date();

Hashtable env = new Hashtable();
env.put(Context.INITIAL_CONTEXT_FACTORY, "com.sun.jndi.ldap.LdapCtxFactory");
env.put(Context.PROVIDER_URL, "ldap://stadm14.us.oracle.com:389");
env.put(Context.SECURITY_AUTHENTICATION, "simple");
env.put(Context.SECURITY_PRINCIPAL, "cn=" + userID +
",cn=users,dc=us,dc=oracle,dc=com");
env.put(Context.SECURITY_CREDENTIALS, password);

try {
   DirContext ctx = new InitialDirContext(env);
} catch (NamingException ex) {
   String redirectURL = "/engine/loginpage.jsp?doneURL=" +
      (doneURL != null ? URLEncoder.encode(doneURL) : "") + "&refID=" +
      (refID != null ? URLEncoder.encode(refID) : "") + "&authnMech=" +
      URLEncoder.encode(authnMethod) + "&message=" +
      URLEncoder.encode(ex.toString() + " for " + userID + " / " + password) +
         (getUserSession != null && getUserSession.length() > 0 ? "&getUsrSess=" +
URLEncoder.encode(getUserSession) : "");
   response.sendRedirect(redirectURL);
   return;
}
```

```
request.setAttribute("oracle.security.sso.sasso.uid", userID);
request.setAttribute("oracle.security.sso.sasso.refID", refID);
request.setAttribute("oracle.security.sso.sasso.authnMech", authnMethod);
request.setAttribute("oracle.security.sso.sasso.authnInst", now);

if ("true".equals(getUserSession))
   request.getSession().getServletContext().getContext("
    /shareid").getRequestDispatcher("/saml/ObSAMLLoginService").forward(request,
response);
else
   request.getSession().getServletContext().getContext
    ("/fed").getRequestDispatcher(doneURL).forward(request, response);
%>
```

Oracle Identity Federation Configuration Files

Modify these files to define the Authentication Engine `contextPath` and `relativePath`:

```
$ORACLE_HOME/fed/conf/idpmanager/authn-mappings.xml
$ORACLE_HOME/fed/conf/spmanager/authn-mappings.xml
$ORACLE_HOME/fed/conf/usermanager/authn-mappings.xml
```

Each file needs the same set of changes:

```
<authn-mapping>
   <authn-method>sasso_login_idp_context</authn-method>
   <authn-screen>/engine</authn-screen>
</authn-mapping>
<authn-mapping>
   <authn-method>sasso_login_idp_url</authn-method>
   <authn-screen>/loginpage.jsp</authn-screen>
</authn-mapping>
```

The `$ORACLE_HOME/fed/shareid/oblix/config/shareid-config.xml` file is modified to define the Authentication Engine URL, and to set the `useLocalConfig` attribute to `true` to force changes to be persisted at restart:

```
<SHAREidConfiguration … useLocalConfig="true">
   …
   <AuthMethod Name="Sasso" LoginURL="/engine/authentication.jsp"
    LogoutURL="/sso/logout" />
</SHAREidConfiguration>"
```

Logout

Since no cookies are set in this flow, the authentication engine is not required to integrate with the logout flow (described in "Logout" on page 4-49).

Creating a Custom SP Integration Engine

This section explains how to plan, develop, and implement a custom SP integration engine.

See Also:

- "Architecture" on page 4-31 for a description of the SP Integration Engine and how it fits into Oracle Identity Federation architecture.
- "The GenericSPCookieProvider Example" on page 4-54 for a description of a working SP Integration Engine.

Planning a Custom SP Integration Engine The steps for developing a custom SP Integration Engine involve:

- creating a service to process requests from Oracle Identity Federation in SP mode
- implementing a module to create an authenticated session for a user at the IAM server
- redirecting the user to the final target URL
- deciding whether the SP Integration Engine will set a cookie after it creates an authenticated session at the IAM Server. If so, the engine needs to be integrated into the logout process (see "Logout" on page 4-49).
- packaging these services and module into a web application, and deploying it to the OC4J_FED instance of the Application Server where the federation server is running
- modifying Oracle Identity Federation configuration files to reference the new SP Integration Engine; this requires that OC4J_FED be restarted.
- if the SP Integration Engine accesses a user repository, ensuring that it is the same user data store configured in the Oracle Identity Federation Administration Console for use by Oracle Identity Federation

Developing and Implementing the Integration Module This section describes how to develop the integration module and how to implement it in the federation environment.

URLs

Communication between the Identity Federation Engine and the SP Integration Engine requires internal servlet forwards that are equivalent to API calls. These forwards are achieved with the following J2EE API:

```
ServletContext.getContext(String contextPath)
   .getRequestDispatcher(String relativePath)
   .forward(HttpServletRequest request,
      HttpServletResponse response)
```

where

- `contextPath` is the root context path of the web application. For example, the `contextPath` of the Oracle Identity Federation server is `/fed`.
- `relativePath` is the service URL to which the uses if forwarded, and is relative to the `contextPath`. For example, after authentication, the Authentication Engine uses `/user/loginsso` as the `relativePath` when forwarding the user.

Oracle Identity Federation needs to be aware of the `contextPath` and the `relativePath` of the new SP Integration Engine; these are the URLs publishing the service that interacts with the IAM Server to create an authenticated session. You must modify the `$ORACLE_HOME/fed/conf/spmanager/authn-mappings.xml` file to configure these URLs:

- In the <authn-mapping> element containing the <authn-method> whose value equals sasso_login_sp_context, modify the value of the corresponding <authn-screen> to the contextPath of the new SP Integration Engine (by default, this is set to /sso).
- In the <authn-mapping> element containing the <authn-method> whose value equals sasso_login_sp_url, modify the value of the corresponding <authn-screen> to the relativePath of the new SP Integration Engine (by default, this is set to /authn).

The SAML 1.x and WS-Fed protocols also need to be aware of the contextPath and the relativePath of the new SP Integration Engine. To configure these URLs, modify the $ORACLE_HOME/fed/shareid/oblix/config/shareid-config.xml file like this: in the <LoginConfig> XML element, locate the <AuthnMethod> element whose Name attribute equals SassoSP, and set the LoginURL attribute to the concatenation of the contextPath and the relativePath.

> **Note:** Due to a limitation in Oracle Identity Federation for the SAML 1.x/WS-Fed protocols, the relativePath can only contain one "/" character; there are no restrictions on the contextPath. For example, contextPath can be set to /path1/path2/path3, while relativePath can only be set to /path4; a value of /path5/path6 for the relativePath is invalid.

To persist these changes, set the useLocalConfig attribute of the top <SHAREidConfiguration> XML element to true.

After changing the authn-mappings.xml and the shareid-config.xml files, restart the OC4J_FED instance.

Implementing the Service

Upon receiving a request from Oracle Identity Federation, the SP Integration Engine needs to:

- create an authenticated session for the user
- redirect the user to the final URL

The engine must process the following incoming data:

- oracle.security.sso.sasso.uid – HttpServletRequest attribute of type String containing the unique user identifier
- oracle.security.sso.sasso.authnInst – HttpServletRequest attribute of type java.util.Date containing the user authentication time
- oracle.security.sso.sasso.expiryInst – HttpServletRequest attribute of type java.util.Date containing the instant at which the authenticated session should expire
- oracle.security.sso.sasso.targetURL – HttpServletRequest attribute of type String containing the final target URL
- oracle.security.sso.sasso.authn.dyna_mode – HttpServletRequest attribute of type String. In SP mode, this value is always set to sp.

Using this data, the SP Integration Engine creates an authenticated session and redirects the user to the final target URL.

If the service needs to set cookies, the cookie path must be set to "/". This is necessary because of the internal forwards between the Oracle Identity Federation and SP Integration Engine web applications; the user's browser needs to send the cookies related to the SP Integration Engine, even when accessing only the federation server. This way, when an internal forward occurs from the federation server to the SP Integration Engine, the cookie set by the latter will be available in the HTTP Request.

Sample Integration Modules The next two sections provide examples of implementing a custom authentication engine:

- Sample Integration Module 1: OC4J_FED Integration
- Sample Integration Module 2: Customized Single Sign-On Integration

> **Note:** Oracle strongly discourages users from deploying any applications on the OC4J_FED J2EE container instance other than the ones for custom integration and authentication described as sample integration modules 1 and 2 below, because doing so introduces potential security risks. Extraneous applications deployed in the OC4J_FED container can potentially affect the security of the federation server by allowing rogue software to change the behavior of the server flows.

Sample Integration Module 1: OC4J_FED Integration This section shows a simple SP Integration Engine that uses the `javax.servlet.http.HttpSession` to set an attribute. The presence of this attribute shows whether or not a user is authenticated.

> **Note:** The example in this section is intended for illustration only and should not be used in a production environment. Indeed, it supposes that other applications deployed on the OC4J_FED instance will consume data set by the SP Integration Engine, which is an approach strongly discouraged by Oracle. Furthermore, this example might not function properly in certain deployments, especially when propagating HttpSession across J2EE applications.

Setup

The SP Integration Engine will not interact with the user data store used by Oracle Identity Federation. Several J2EE applications are deployed on the OC4J_FED instance.

Packaging

The SP Integration Engine consists of a Web application with a root context set to `/engine`, and contains two JSP pages:

- `oc4jintegration.jsp`, which processes the request from the federation server and creates an HttpSession with a `feduserid` attribute containing the user's identifier
- `application.jsp`, which serves as an application. It looks for the HttpSession's `feduserid` attribute, and triggers a Federation SSO if the attribute is not found

Implementation of application.jsp

```
<%@page buffer="5" autoFlush="true" session="false"%>
<%@page language="java" import="java.net.*"%>
<%
```

```
response.setHeader("Cache-Control", "no-cache");
response.setHeader("Pragma", "no-cache");
response.setHeader("Expires", "Thu, 29 Oct 1969 17:04:19 GMT");

String userid = (String)request.getSession().getAttribute("feduserid");
if (userid == null || userid.length() == 0) {
   String redirectURL = "/fed/sp/initiatesso?providerid=" +
      URLEncoder.encode("http://stadm14.us.oracle.com:7778/fed/idp") +
      "&returnurl=/engine/application.jsp";
   response.sendRedirect(redirectURL);
   return;
}
%>
Welcome <%=userid%>
```

Implementation of oc4jintegration.jsp

```
<%@page buffer="5" autoFlush="true" session="false"%>
<%@page language="java" import="java.util.*"%>
<%
response.setHeader("Cache-Control", "no-cache");
response.setHeader("Pragma", "no-cache");
response.setHeader("Expires", "Thu, 29 Oct 1969 17:04:19 GMT");

String userid = (String)request.getAttribute("oracle.security.sso.sasso.uid");
Date authnInst =
(Date)request.getAttribute("oracle.security.sso.sasso.authnInst");
Date expirationInst =
(Date)request.getAttribute("oracle.security.sso.sasso.expiryInst");
String targetURL =
(String)request.getAttribute("oracle.security.sso.sasso.targetURL");

request.getSession().setAttribute("feduserid", userid);
response.sendRedirect(targetURL);
%>
```

Oracle Identity Federation Configuration Files

The `$ORACLE_HOME/fed/conf/spmanager/authn-mappings.xml` file is modified to define the SP Integration Engine `contextPath` and `relativePath`:

```
<authn-mapping>
   <authn-method>sasso_login_sp_context</authn-method>
   <authn-screen>/engine</authn-screen>
</authn-mapping>
<authn-mapping>
   <authn-method>sasso_login_sp_url</authn-method>
   <authn-screen>/oc4jintegration.jsp</authn-screen>
</authn-mapping>
```

The `$ORACLE_HOME/fed/shareid/oblix/config/shareid-config.xml` file is modified to define the SP Integration Engine URL, and to set the `useLocalConfig` attribute to `true` to force the changes to be persisted at restart:

```
<SHAREidConfiguration … useLocalConfig="true">
   ..
   <AuthMethod Name="SassoSP" LoginURL="/engine/oc4jintegration.jsp"
LogoutURL="/sso/logout" />
</SHAREidConfiguration>
```

Logout

Since this application sets up an HttpSession in the OC4J_FED instance, the SP Integration engine must be integrated in the logout flow (see "Logout" on page 4-49).

Sample Integration Module 2: Customized Single Sign-On Integration This section shows an SP Integration Engine that uses a simple Single Sign-On framework based on a cookie containing the username and the expiration time of the authenticated session.

> **Note:** This example is intended for illustration only and should not be used in a production environment. For example, the cookies set in this example are not encrypted, allowing an attacker to impersonate a user by manually constructing such cookies.

Setup

The SP Integration Engine will not interact with the user data store used by Oracle Identity Federation. The Engine will set up a cookie, for the entire domain, containing the user's identifier as a `String` variable and the session timeout as a long.

Packaging

The SP Integration Engine consists of a Web application with a root context set to `/engine`, and contains two JSP pages:

- `domainintegration.jsp`, which processes the request from the Oracle Identity Federation server and creates a cookie with the user ID and session timeout
- `domainapplication.jsp`, which serves as an application. It looks for the cookie and triggers a federation SSO if the cookie is not found.

Implementation of domainapplication.jsp

```
<%@page buffer="5" autoFlush="true" session="false"%>
<%@page language="java" import="java.net.*, java.util.*"%>
<%
response.setHeader("Cache-Control", "no-cache");
response.setHeader("Pragma", "no-cache");
response.setHeader("Expires", "Thu, 29 Oct 1969 17:04:19 GMT");

Cookie[] cookies = request.getCookies();
String userid = null;
Date timeout = null;
for(int i = 0, size = (cookies != null ? cookies.length : 0); i < size; i++) {
   String name = cookies[i].getName();
   if ("spintegrationcookie".equals(name)){
      String value = cookies[i].getValue();
      StringTokenizer st = new StringTokenizer(value, "*");
      userid = st.nextToken();
      timeout = new Date(Long.parseLong(st.nextToken()));
      break;
   }
}

if (userid == null || userid.length() == 0) {
   String redirectURL = "http://stadm04.us.oracle.com:7778" +
      "/fed/sp/initiatesso?providerid=" +
      URLEncoder.encode("http://stadm14.us.oracle.com:7778/fed/idp") +
      "&returnurl=" + URLEncoder.encode(
         "http://stadm04.us.oracle.com:7778/engine/domainapplication.jsp");
```

```
   response.sendRedirect(redirectURL);
   return;
}
%>
Welcome <%=userid%>. You are logged until <%=timeout%>
```

Implementation of domainintegration.jsp

```
<%@page buffer="5" autoFlush="true" session="false"%>
<%@page language="java" import="java.util.*"%>
<%
response.setHeader("Cache-Control", "no-cache");
response.setHeader("Pragma", "no-cache");
response.setHeader("Expires", "Thu, 29 Oct 1969 17:04:19 GMT");

String userid = (String)request.getAttribute("oracle.security.sso.sasso.uid");
Date authnInst =
(Date)request.getAttribute("oracle.security.sso.sasso.authnInst");
Date expirationInst =
(Date)request.getAttribute("oracle.security.sso.sasso.expiryInst");
String targetURL =
(String)request.getAttribute("oracle.security.sso.sasso.targetURL");

String cookieValue = userid + "*" + expirationInst.getTime();
Cookie cookie = new Cookie("spintegrationcookie", cookieValue);
cookie.setDomain(".us.oracle.com");
cookie.setPath("/");
response.addCookie(cookie);
response.sendRedirect(targetURL);
%>
```

Oracle Identity Federation Configuration Files

The `$ORACLE_HOME/fed/conf/spmanager/authn-mappings.xml` file is modified to define the SP Integration Engine `contextPath` and `relativePath`:

```
<authn-mapping>
   <authn-method>sasso_login_sp_context</authn-method>
   <authn-screen>/engine</authn-screen>
</authn-mapping>
<authn-mapping>
   <authn-method>sasso_login_sp_url</authn-method>
   <authn-screen>/domainintegration.jsp</authn-screen>
</authn-mapping>
```

The `$ORACLE_HOME/fed/shareid/oblix/config/shareid-config.xml` file is modified to define the SP Integration Engine URL, and to set the `useLocalConfig` attribute to `true` to force changes to be persisted at restart:

```
<SHAREidConfiguration … useLocalConfig="true">
…
   <AuthMethod Name="SassoSP" LoginURL="/engine/domainintegration.jsp"
LogoutURL="/sso/logout" />
</SHAREidConfiguration>
```

Logout

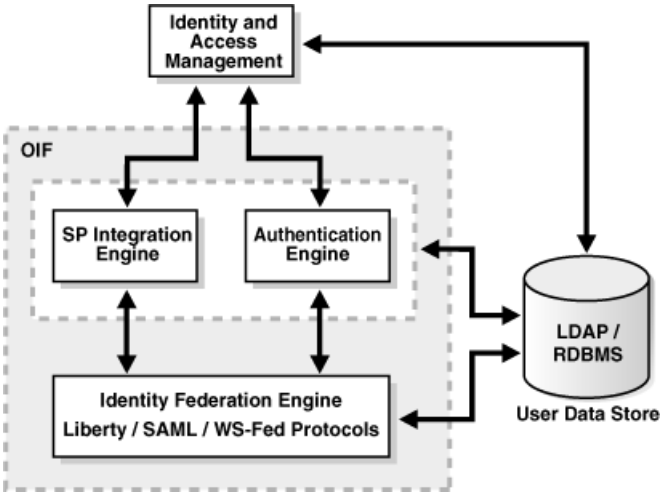
Since this sample application sets up a domain cookie, the SP Integration engine must be integrated into the logout flow (see Logout below).

Logout

This section explains how to configure logout flows.

Current Integration During the logout flows, the Identity Federation Engine considers the SP Integration and Authentication engines as a single entity, as illustrated by Figure 4–2:

Figure 4–2 Oracle Identity Federation Modules



Here is a description of the action performed by the different Oracle Identity Federation modules when a logout process is being invoked:

1. The user previously accessed Oracle Identity Federation to perform an SSO operation (in either SP or IdP mode).
2. The user accesses the Oracle Identity Federation Logout service URL `/fed/user/logout` with the following query parameter:
   - `returnurl` - the URL to which to redirect the user after completion of the logout process [optional]. See "Configuring the Logout Service" on page 6-132 for details.
3. The Oracle Identity Federation user session will be marked as logging out.
4. Oracle Identity Federation will redirect the user to the servlet responsible for the session termination of the Authentication and SP Integration engines located at the `/sso/logout` URL. The servlet may interact with the IAM framework to perform the logout operation. The redirect from Oracle Identity Federation to the logout servlet could contain the following query parameters:
   - `invokeOSFSLogout` – a boolean value indicating whether the Logout Authentication and SP Integration servlet should redirect the user to the `/fed/user/logoutsso` URL after having logged the user out from the IAM and the Authentication/SP Integration engines. If this parameter is missing or false, the `doneURL` parameter should be used to redirect the user after local logout is complete.
   - `doneURL` – a URL-encoded value containing the URL to which to redirect the user after having logged out the user from the IAM and the Authentication/SP Integration engines. This parameter is to be used when `invokeOSFSLogout` is `false` or missing.

5. The user is redirected to Oracle Identity Federation to perform the Federation Logout operations with the remote providers.
6. After the federation logout steps are performed, the Oracle Identity Federation user session is destroyed.
7. The user is redirected to the `returnurl` location.

Changing Logout Flow This section contains topics relevant to redirection during logout.

URLs

During the logout operations, the user is being redirected between the Federation Engine and the logout service of the Authentication and SP Integration Engines.

Oracle Identity Federation needs to be aware of the location of the logout service in order to redirect the user to the servlet/jsp page for logout. This URL is defined in the `authn-mappings.xml` file. The URL can be defined as the union of a `contextPath` and a `relativePath`:

- `contextPath` is the root context path of the web application. For example, the `contextPath` of Oracle Identity Federation is `/fed`.
- `relativePath` is the service URL to which to forward the user; it is relative to the `contextPath`. For example, the default logout location of the Authentication and SP Integration Engine is `/logout` as the `relativePath`.

Modify the following files to configure these URLs:

```
$ORACLE_HOME/fed/conf/idpmanager/authn-mappings.xml
$ORACLE_HOME/fed/conf/spmanager/authn-mappings.xml
$ORACLE_HOME/fed/conf/usermanager/authn-mappings.xml
```

Each file needs the same set of changes:

- In the `<authn-mapping>` element containing the `<authn-method>` whose value equals `sasso_logout_context`, modify the value of the corresponding `<authn-screen>` to the `contextPath` of the new Authentication Engine; by default, that value is set to `/sso`.
- In the `<authn-mapping>` element containing the `<authn-method>` whose value equals `sasso_logout_url`, modify the value of the corresponding `<authn-screen>` to the `relativePath` of the logout service; by default, that value is set to `/logout`.

The SAML 1.x and WS-Fed protocols also need to be aware of the `contextPath` and the `relativePath` of the new logout service. To configure these URLs, modify the `$ORACLE_HOME/fed/shareid/oblix/config/shareid-config.xml` file like this:

- In the `LoginConfig` XML element, locate the `AuthnMethod` element whose `Name` attribute equals `Sasso`, and set the `LogoutURL` attribute to the concatenation of the `contextPath` and the `relativePath`.
- In the `LoginConfig` XML element, locate the `AuthnMethod` element whose `Name` attribute equals `SassoSP`, and set the `LogoutURL` attribute to the concatenation of the `contextPath` and the `relativePath`.
- In the `<LogoutConfig>` XML element, set the `SassoURL` attribute to the concatenation of the `contextPath` and the `relativePath`.

Note: Due to a limitation in Oracle Identity Federation for the SAML 1.x/WS-Fed protocols, the `relativePath` can only contain one "/" character; there are no restrictions on the `contextPath`. For example, `contextPath` can be set to `/path1/path2/path3`, while `relativePath` can only be set to `/path4`; a value of `/path5/path6` for the `relativePath` is invalid.

To persist these changes, set the `useLocalConfig` attribute of the top `<SHAREidConfiguration>` XML element to `true`.

Restart the OC4J_FED instance after changing the `authn-mappings.xml` files.

Implementing the Logout Service

The operations that need to be performed by the logout service include:

- processing requests from the Federation Engine
- interacting with the IAM framework with which it is integrated to log the user out
- redirecting the user to the Federation Engine or to the `doneURL` parameter, if both Authentication and SP Integration engines bundled with Oracle Identity Federation have been replaced
- redirecting the `/sso/logout` URL with the query parameters passed from Oracle Identity Federation (`invokeOSFSLogout` and/or `doneURL`), if only one of the engines, authentication engine or SP Integration engine has been replaced. This is required since one of the default Oracle Identity Federation engines has been used and the user session needs to be logged out from this module.

Sample Logout Services In the next two sections, these scenarios of logout services are outlined:

- Logout Service Example #1 describes a custom logout service when both the Authentication and SP Integration engines are customized
- Logout Service Example #2 describes a custom logout service when only the SP Integration engine is customized

Logout Service Example #1 This section describes how to integrate a custom Logout Service, assuming that both the Authentication and SP Integration engines have been customized, that is, the default engines are not used anymore.

Setup

In this example, the Authentication Engine is the LDAP engine described in section 3.3.2, and the SP Integration Engine is the OC4J_FED Integration engine described in "Sample Integration Module 1: OC4J_FED Integration" on page 4-45.

Packaging

The Logout service consists of a JSP page bundled with the Authentication and SP Integration engines:

- `logout.jsp`, which will process the request from the Oracle Identity Federation server, remove the `feduserid` attribute from the HttpSession object, set in the `oc4jintegration.jsp` page, and redirect the user to either Oracle Identity Federation or the `doneURL` parameter.

Implementation of logout.jsp

```
<%@page buffer="5" autoFlush="true" session="false"%>
<%@page language="java" import="java.net.*"%>
<%
response.setHeader("Cache-Control", "no-cache");
response.setHeader("Pragma", "no-cache");
response.setHeader("Expires", "Thu, 29 Oct 1969 17:04:19 GMT");

String oifContext = "/fed";
String oifLogoutPath = "/user/logoutsso";
String invokeOSFSLogout = request.getParameter("invokeOSFSLogout");
String doneURL;
if ("true".equals(invokeOSFSLogout))
doneURL = oifContext + oifLogoutPath;
else
doneURL = request.getParameter("doneURL");

request.getSession().removeAttribute("feduserid");
response.sendRedirect(doneURL);
%>
```

Oracle Identity Federation Configuration Files

Modify these files to define the Authentication Engine `contextPath` and `relativePath`:

```
$ORACLE_HOME/fed/conf/idpmanager/authn-mappings.xml
$ORACLE_HOME/fed/conf/spmanager/authn-mappings.xml
$ORACLE_HOME/fed/conf/usermanager/authn-mappings.xml
```

Each file needs the same set of changes:

```
<authn-mapping>
   <authn-method>sasso_logout_context</authn-method>
   <authn-screen>/engine</authn-screen>
</authn-mapping>
<authn-mapping>
   <authn-method>sasso_logout_url</authn-method>
   <authn-screen>/logout.jsp</authn-screen>
</authn-mapping>
```

The `$ORACLE_HOME/fed/shareid/oblix/config/shareid-config.xml` file is modified to define the Logout Service URL in three places, and to set the `useLocalConfig` attribute to `true` to force the changes to be persisted at restart:

```
<SHAREidConfiguration … useLocalConfig="true">
<LoginConfig>
    …
   <AuthMethod Name="Sasso" LoginURL="/engine/loginpage.jsp"
    LogoutURL="/engine/logout.jsp" />
   <AuthMethod Name="SassoSP"
    LoginURL="/engine/oc4jintegration.jsp" LogoutURL="/engine/logout.jsp" />
</LoginConfig>
<LogoutConfig Protocol="http" HostName="…"
 Port="…" SassoURL="/engine/logout.jsp" OsfsURL="/fed/user/logoutwsfed" />
</SHAREidConfiguration>
```

Logout Service Example #2 This section describes how to integrate a custom Logout Service, assuming that only the SP Integration engine has been customized, that is, the default authentication engine is still in use.

Setup

In this example, the Authentication Engine is LDAP standalone, configured through the Oracle Identity Federation Administration Console, and the SP Integration Engine is the OC4J_FED Integration engine described in "Sample Integration Module 1: OC4J_FED Integration" on page 4-45.

Packaging

The Logout service consists of a JSP page bundled with the Authentication and SP Integration engines:

- `logout.jsp`, which will process the request from the Oracle Identity Federation server, remove the `feduserid` attribute from the HttpSession object, set in the `oc4jintegration.jsp` page, and redirect the user to the `/sso/logout` URL with the `invokeOSFSLogout` and `doneURL` parameters.

Implementation of logout.jsp

```
<%@page buffer="5" autoFlush="true" session="false"%>
<%@page language="java" import="java.net.*"%>
<%
response.setHeader("Cache-Control", "no-cache");
response.setHeader("Pragma", "no-cache");
response.setHeader("Expires", "Thu, 29 Oct 1969 17:04:19 GMT");

String ssoLogout = "/sso/logout";
String invokeOSFSLogout = request.getParameter("invokeOSFSLogout");
String doneURL = request.getParameter("doneURL");
String queryString = "";

if (invokeOSFSLogout != null && invokeOSFSLogout.length() > 0)
queryString = queryString + "invokeOSFSLogout=" +
URLEncoder.encode(invokeOSFSLogout);
if (doneURL!= null && doneURL.length() > 0)
{
        if (queryString.length() > 0)
                queryString = queryString + "&";
        queryString = queryString + "doneURL=" + URLEncoder.encode(doneURL);
}

if (queryString.length() > 0)
        ssoLogout = ssoLogout + "?" + queryString;

request.getSession().removeAttribute("feduserid");
response.sendRedirect(ssoLogout);
%>
```

Oracle Identity Federation Configuration Files

Modify these files to define the Authentication Engine `contextPath` and `relativePath`:

```
$ORACLE_HOME/fed/conf/idpmanager/authn-mappings.xml
$ORACLE_HOME/fed/conf/spmanager/authn-mappings.xml
$ORACLE_HOME/fed/conf/usermanager/authn-mappings.xml
```

Each file needs the same set of changes:

```
<authn-mapping>
   <authn-method>sasso_logout_context</authn-method>
   <authn-screen>/engine</authn-screen>
</authn-mapping>
<authn-mapping>
   <authn-method>sasso_logout_url</authn-method>
   <authn-screen>/logout.jsp</authn-screen>
</authn-mapping>
```

The `$ORACLE_HOME/fed/shareid/oblix/config/shareid-config.xml` file is modified to define the Logout Service URL in three places, and to set the `useLocalConfig` attribute to `true` to force the changes to be persisted at restart:

```
<SHAREidConfiguration … useLocalConfig="true">
<LoginConfig>
    …
   <AuthMethod Name="Sasso" LoginURL="/engine/loginpage.jsp"
    LogoutURL="/engine/logout.jsp" />
   <AuthMethod Name="SassoSP"
    LoginURL="/engine/oc4jintegration.jsp" LogoutURL="/engine/logout.jsp" />
</LoginConfig>
<LogoutConfig Protocol="http" HostName="…"
 Port="…" SassoURL="/engine/logout.jsp" OsfsURL="/fed/user/logoutwsfed" />
</SHAREidConfiguration>
```

The GenericSPCookieProvider Example

Metalink note 427374.1 contains the source code for a working SP Integration Engine. It includes these files:

- `cookieprovider.jsp`, which generates a user cookie
- `testapplication.jsp`, which tests the deployment by looking for the cookie
- `GenericSPCookieProvider.EAR`

The note provides complete details about this working example, including:

- a description of what the EAR does;
- how to deploy the EAR file to the application server; and
- how to test EAR deployment.

This Metalink note is available at http://webiv.oraclecorp.com/cgi-bin/webiv//do.pl/Get?WwwID=note:427374.1.

Implementing HTTP Basic Authentication

Oracle Identity Federation can be configured to accept HTTP basic credentials, with or without an identity and access management system, when using SAML 1.0/1.1 or WS-Federation protocols.

> **Note:** The techniques described in this section are applicable only if Oracle Identity Federation is acting as an IdP. They do not apply if Oracle Identity Federation is the SP.

Basic Authentication with an Identity Store

You can configure basic authentication in an environment where Oracle Identity Federation as IdP authenticates users against an identity store such as Oracle Access Manager. A typical flow using the Browser Artifact (or Browser POST) profile is as follows:

1. A non-browser client wants to federate with an application at an SP, using an Oracle Identity Federation server installed at an IdP. The client sends an HTTP GET request to the IdP.
2. The Oracle Identity Federation IdP sends a `401 Not Authorized` response back to the client.
3. The client re-sends the `GET` with an Authorization header containing HTTP basic or NTML credentials that the client has obtained through unspecified means.
4. The IdP uses the credentials in the Authorization header to authenticate the user against its identity store.
5. The Oracle Identity Federation IdP then creates an assertion for the user, and (for the artifact profile), associates the assertion with an artifact, and sends a 302 Redirect response to the client for the SP's Assertion Receiver Server with the artifact and the target URL.
6. The client responds to the redirect by sending a GET request to the SP's Assertion Receiver Service.
7. The SP's Assertion Receiver Service processes the GET request, determines the IdP from the artifact and sends an Artifact retrieval request to the IdP's SOAP responder service. The IdP returns the assertion associated with the artifact and the SP performs the appropriate processing on the artifact to allow access to the target URL.
8. The client responds to the redirect by sending a GET request to the application, including any SSO cookie set in step 6.
9. If the application is protected by a web access manager, the access manager accepts the SSO cookie and allows access to the application. Alternately, the application might directly retrieve the assertion from the SP SAML component and use that to authorize access.

To use HTTP basic authentication for this flow when using Oracle Access Manager (OAM):

1. Using the OAM Policy Manager, modify the Transfer Service policy in the Fed Domain policy domain (which is automatically set up when you configure Oracle Identity Federation to use OAM) to use a basic authentication scheme. Depending on what OAM setup options were chosen, some pre-configured basic schemes can be used.
2. Install an OAM OHS WebGate agent into the HTTP_Server component of Oracle Identity Federation. The Transfer Service policy set up in Step 1 will cause WebGate to send the 401 response back to the client, and to use the Authorization header from the client to create an `ObSSOCookie`, which is then used by Oracle Identity Federation.

> **Note:** The fix for Oracle Access Manager bug 5736326 is required for this step.

Basic Authentication without an Identity Store

You can configure basic authentication in an environment where Oracle Identity Federation does not authenticate users against an identity store. The flow is similar to that described in "Basic Authentication with an Identity Store" earlier.

To configure basic authentication in this scenario, start with the `$ORACLE_HOME/fed/shareid/oblix/config/shareid-config.xml` file. The file contains a `LoginConfig` element, which looks like this by default:

```
<LoginConfig AuthMethod="Sasso" LogoutPage="/shareid/logout.jsp"
      LoggedInPage="/shareid/loggedIn.jsp" SSOCookieDomain=".us.oracle.com">
   <AuthMethod Name="Basic" RealmName="SHAREid Login" />
   <AuthMethod Name="Form" LoginPage="/shareid/login.jsp" />
   <AuthMethod Name="External" Header="userid" />
   <AuthMethod Name="Sasso" LoginURL="/sso/authn" LogoutURL="/sso/logout" />
   <AuthMethod Name="SassoSP" LoginURL="/sso/authn" LogoutURL="/sso/logout" />
</LoginConfig>
```

The default AuthMethod is the Sasso component that was mentioned in "Basic Authentication with an Identity Store", which can only perform a form-based login. If you change the `AuthMethod` attribute in the `LoginConfig` element to `Basic`, it will activate the HTTP basic method for SAML 1.x. If needed you can also configure the realm used in the 401 response in the `RealmName` attribute.

Integrating WebGate with Oracle Identity Federation Server

This section describes how to install and integrate a WebGate component with the HTTP Server bundled with the Oracle Identity Federation server.

This scenario assumes that Oracle Identity Federation is already integrated with Oracle Access Manager, as shown in "Deploying Oracle Identity Federation with Oracle Access Manager" on page 4-6 and "Edit User Data Store" on page 6-67.

Take these steps for WebGate integration:

1. Create a new AccessGate using the Access System Console, and install the corresponding WebGate on the Oracle HTTP Server of the Application Server where Oracle Identity Federation is running.

   Refer to Oracle Access Manager documentation on how to install WebGate on Oracle HTTP Server.

2. Modify the login JSP page in Oracle Identity Federation as follows:

   Note: This is needed for certain types of authentication schemes to work properly.

   - Back up the `$ORACLE_HOME/j2ee/OC4J_FED/applications/sso/web/jsp/salogin.jsp` file.
   - Edit the $ORACLE_HOME/j2ee/OC4J_FED/applications/sso/web/jsp/salogin.jsp and replace its content with:

     ```
     <%
     response.sendRedirect("/fed/user/sassoredirectlogin?" +
     request.getQueryString());
     %>
     ```

3. Configure Oracle Access Manager policies:

- From the Access System Console, go to the policy manager section and select the Fed Domain policy.
- Click on the **Resources** tab.
- Check the `/shareid` resource URL and click **Delete**.
- Click **Add** to add new resources, with the URLs:

```
/shareid/saml/userAttributes
/shareid/saml/mapping
/shareid/sasso
```

4. Create an Oracle Access Manager policy for use by WebGate/Oracle Identity Federation:
   - From the Access System Console, go to the policy manager section.
   - Create a new policy domain, enter a name and click **Save**.

> **Note:** This policy will be used to protect Oracle Identity Federation login process URLs.

   - From the Resources tab, add the URLs:

```
/fed/user/sassoredirectlogin
/shareid/sassologin.jsp
```

   - Go to the **Default Rule** tab and click on the **Authentication Rule** sub-tab.
   - Select an authentication scheme to use, for example FORM Based or Basic.
   - Go to the **Authorization Rule** tab.
   - Add a new authorization rule and enable it. Click **Save**.
   - Go to the **Allow Access** sub-tab.
   - Configure the users who can access Oracle Identity Federation (for example, `Anyone` for all users) and click **Save**.
   - Go to **Default Rule** tab and choose the **Authorization Expression** sub-tab.
   - Add the authorization rule created in the previous step and click **Save**.
   - Go to the Policy Domains list.
   - Select the newly created policy and enable it.

5

Server Administration

This chapter describes tasks related to day-to-day administration of Oracle Identity Federation, and additional tasks that the administrator may need to perform on occasion. It contains these topics:

- Basic Administration
- Managing Identity Federations
- Reassociation

> **See Also:** For details about initial server configuration, see Chapter 6, "Configuring Oracle Identity Federation"

Basic Administration

This section describes basic administration of Oracle Identity Federation. It contains these topics:

- Role of the Federation Server Administrator
- Logging into Oracle Identity Federation
- Starting and Stopping the Server
- Changing your Administrator Password
- Oracle Identity Federation Log Files

Role of the Federation Server Administrator

The Oracle Identity Federation administrator performs two roles, which can be characterized as:

- Basic runtime administration of the server, including starting, stopping, and monitoring the server
- Federated identity administration, which involves user administration (user creation, deletion, and federation), and maintaining information about trusted providers and the users affiliated with those providers

The remainder of this section provides information to help you plan your Oracle Identity Federation deployment.

Subsequent sections in this chapter focus on basic runtime tasks and identity administration.

Deployment Planning

When deploying Oracle Identity Federation in a network of trusted sources and destinations, you will need to exchange information with other site administrators, and configure identity providers and service providers accordingly.

> **See Also:** "Architecture Options" in Chapter 2, "Planning Oracle Identity Federation Deployment" for details about managing server properties, federation protocols, and circle of trust provider information

Exchange User Identities

In a federated environment, at the simplest level the service provider acts as a consumer of identity information, while the identity provider (where the user request originated) acts as the supplier of identity information. The identity provider may, in turn, adopt a consumer role as it communicates with an authentication and authorization mechanism (a AAA system) to obtain the necessary credentials. Service providers may also want to map users to identities at the destination, although this is not a requirement. Identity suppliers and consumers must be able to achieve a runtime exchange of data, which results in the source asserting some identity information about the principal which the destination can trust as a means of uniquely identifying the principal.

As an identity provider, you may wish to work with partner site administrators to provide the relevant lists of users from your domain. This is an optional information exchange.

Establish Cross-Domain Trust

Oracle Identity Federation can produce and consume provider metadata that conforms to the Liberty metadata specifications for ID-FF 1.1 and 1.2, as well as the SAML 2.0 metadata specification. Additionally, Oracle Identity Federation supports the ability to import provider metadata that uses the metadata extensions for SAML 2.0 query requesters.

You will need to establish cross-domain trust by setting up authentication and exchanging keys or certificates among the network of trusted sources and destinations.

For initial setup and testing, identity providers and service providers can both use default self-signed certificates. When going into production, however, consider the usage type when deciding whether self-signed certificates are sufficient: CA-issued certificates are most useful when there is no prior trust relation between entities, for example, when you use SSL to access a web site over the Internet. But given that the trust relationship between federation peers requires the exchange of metadata or the equivalent, which can and usually does include the peer certificates, self-signed certificates should be sufficient for production deployment so long as you can trust how you obtained the peer certificates. Note that CA-issued certificates might be used in the metadata exchange, for example signed e-mail or a download from a web server over SSL.

The process of setting up cross-domain trust can be simplified by the use of metadata. Oracle Identity Federation enables you to store provider-specific metadata which overrides global IdP and SP settings with data specific to communication with each peer provider.

Note: The SAML 1.x and WS-Federation protocols do not specify metadata formats, so peers using those protocols have to exchange the equivalent information, including certificates, on their own.

PKI and SSL Encryption

Oracle Identity Federation provides secure communication using **X.509** client certificate authentication.

Oracle Identity Federation provides encryption for data integrity using **public key cryptography**, a technique that uses a public and private key pair. Data is signed with a sending party's private key and the signature is verified by the recipient using the sender's public key.

Oracle Identity Federation uses documents known as **certificate**s to enable peer providers to establish trust. A **Certificate Authority (CA)** issues a certificate to vouch for a user's identity, including the party's public key in the certificate for use by the receiving party.

You configure key pairs and certificates using a local keystore. The identity provider configures a public and private key pair and a certificate - providing validation of the public key from a **Certificate Authority (CA)** - when using the POST profile. The presentation of the public key by the IdP, and certificate importation by the SP, are critical aspects in managing the trust relationship between partners.

You can also implement **SSL** connections. For details on how to configure SSL connections and client certificates, see "Using SSL with Oracle Identity Federation" on page 6-133.

Note: SSL functionality is external to Oracle Identity Federation.

Other Planning Tasks

Besides exchanging identities and securing communications involving those identities, parties that plan to engage in a federated network must agree on a range of additional topics, such as:

- federation protocols
- services
- profiles

You will need to work with others in your network to ensure that the various IdPs and SPs understand their business partners' setups in order for federation to work properly.

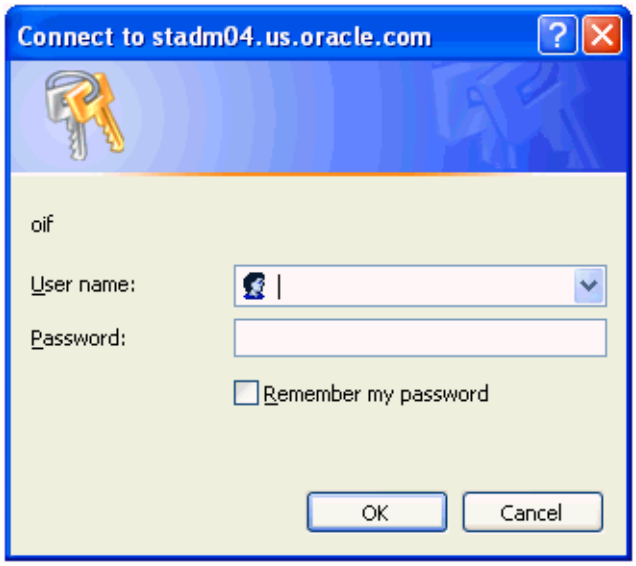
Logging into Oracle Identity Federation

To log in to the Oracle Identity Federation administration console:

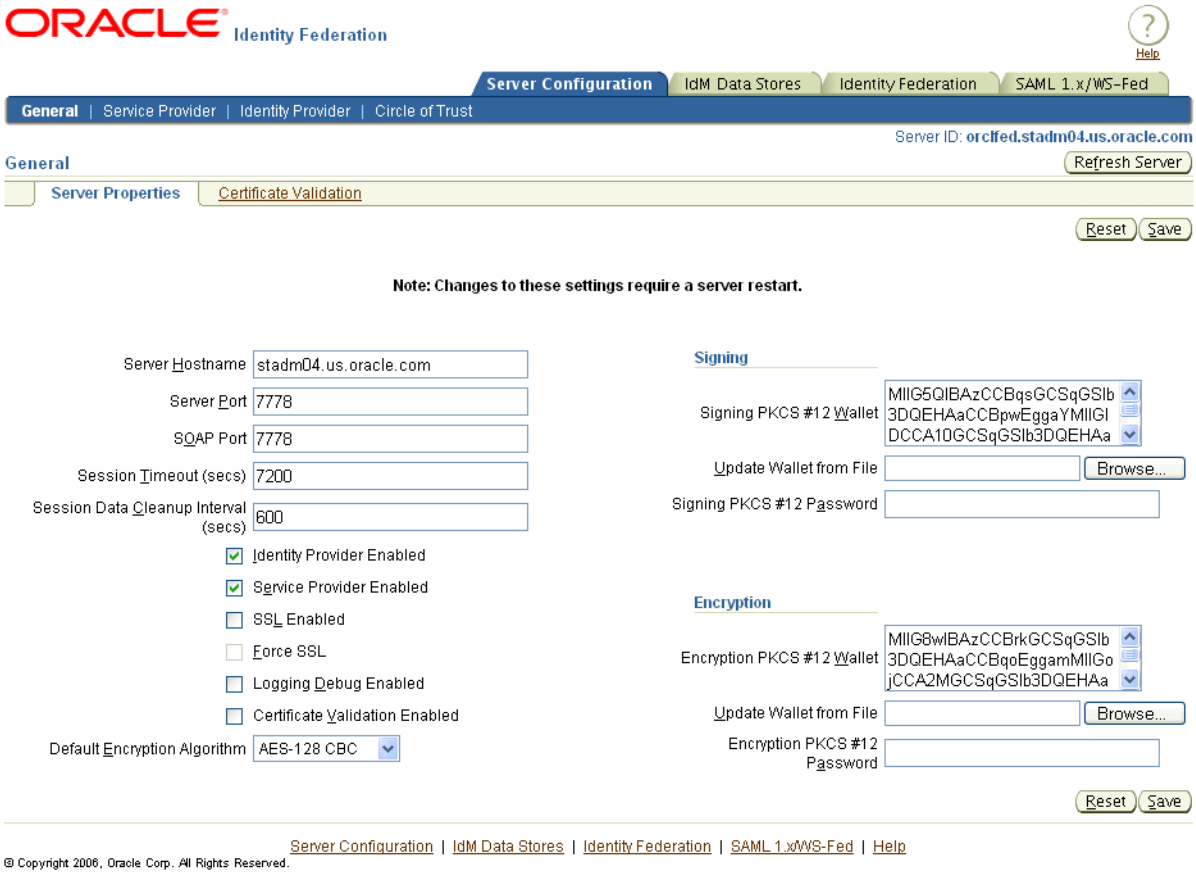
1. Start the login process by pointing to the login URL:

   http://machine-name:port/fedadmin

2. A login window appears:



3. Log into Oracle Identity Federation using the username and password supplied during installation. The Server Configuration home page appears:



To log out, close your browser window.

Starting and Stopping the Server

You can start and stop the Oracle Identity Federation server through the Enterprise Manager console or by using the `opmnctl` command-line tool (Oracle Process Manager and Notification Server Control Utility) on the OC4J_FED container.

For example, the command to restart Oracle Identity Federation is:

```
opmnctl restartproc process-type=OC4J_FED
```

If Oracle Identity Federation is using an RDMBS data store, the server must be started or stopped in a specific sequence relative to the database:

- The RDBMS must be started before Oracle Identity Federation is started.
- The RDBMS must be stopped after Oracle Identity Federation is stopped.

If you do not follow this sequence, Oracle Identity Federation will not operate properly since the data store will not be available.

See Also:

- "Starting Oracle Identity Federation" in the *Oracle Application Server Administrator's Guide*
- "Stopping Oracle Identity Federation" in the *Oracle Application Server Administrator's Guide*
- *Oracle Enterprise Manager Concepts*
- *Oracle Identity Management User Reference*

Changing your Administrator Password

You can change the federation server administration password through the Enterprise Manager console using these steps:

1. Click **OC4J_FED**.
2. Click **Applications**.
3. Click **fed**.
4. Click **Security**.
5. Click **oif/oif_admin**.

Note: You will need to restart the `OC4J_FED` instance for the password change to take effect.

Oracle Identity Federation Log Files

Oracle Identity Federation log files, including logs for SAML2.0/Liberty 1.x messages, are maintained in the `$ORACLE_HOME/fed/log` directory and provide useful information for managing and monitoring server instances. The log files include:

Table 5–1 Oracle Identity Federation Log Files

Log File Name	Description
`federation.log`	Contains the runtime log records for the Oracle Identity Federation server.
`federation-error.log`	Contains error messages generated by the Oracle Identity Federation server.

Table 5–1 (Cont.) Oracle Identity Federation Log Files

Log File Name	Description
`federation-msg.log`	Contains the SAML2.0/Liberty 1.x messages exchanged between Oracle Identity Federation and peer providers.
`install.log`	Contains a log of the installation session.
`uninstall.log`	Contains a log of the uninstall session.

Backups

You should back up your configurations/systems with the tools that you normally employ to back up your systems on a daily basis.

Windows

Use this backup regimen:

- Use the backup/restore system tools on window platforms.
- Back up everything on all components in the Oracle Identity Federation configuration.

Linux/solaris:

Use this backup regimen:

- Shut down all Oracle Application Server and Oracle Identity Federation components.
- Run the `tar` command all on components, including Oracle Application Server, Oracle Identity Federation, and the RDBMS data files that Oracle Identity Federation is using.

  For example:

  ```
  tar cvzf oif10141_backup oif_folder
  ```

Managing Identity Federations

Clicking on the **Identity Federation** tab of the Oracle Identity Federation administration console brings you to a page where you can manage federations. Use this page to edit configuration details for Oracle Identity Federation trusted providers and users.

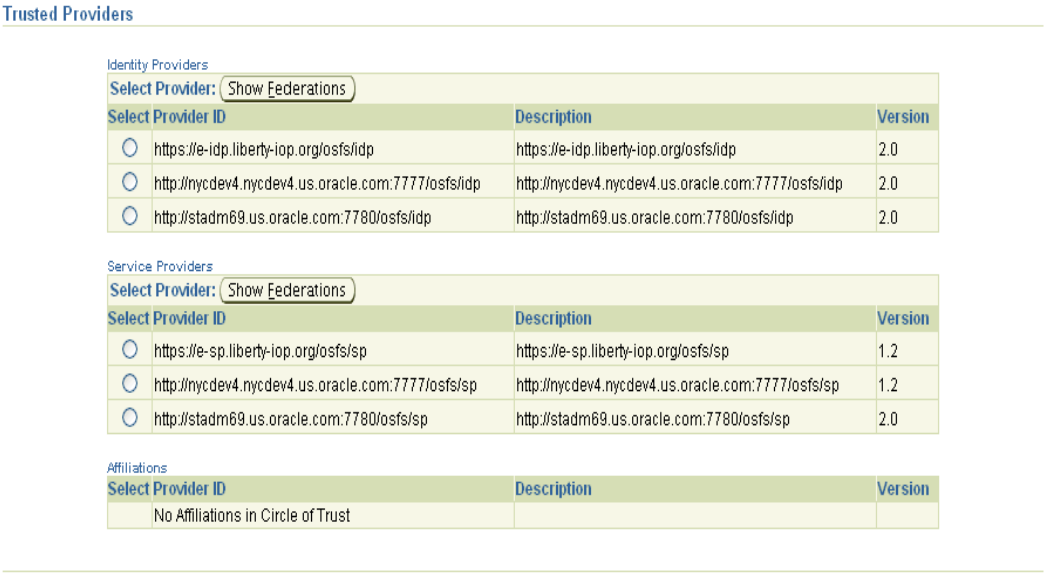
- Clicking on **Trusted Providers** (default when the page loads) displays the list of all providers in the Oracle Identity Federation circle of trust.
- Clicking on **Users** allows you to search for all users who have identity federations associated with this instance of Oracle Identity Federation.

> **See Also:** For information about accessing the administration console, see "Administration Console Overview" on page 6-5

This section contains these topics:

Edit Trusted Provider Configuration

This page displays all providers in the Oracle Identity Federation server's circle of trust.



Use the information on the page as follows:

- Select one of the available identity providers and click **Show Federations** to display the identity federations associated with this Oracle Identity Federation server for the selected provider.
- Select one of the available service providers and click **Show Federations** to display the identity federations associated with this Oracle Identity Federation server for the selected provider.
- Select an affiliation in the circle of trust.

The server will display federations between:

- this Oracle Identity Federation server as an SP, and remote IdPs from the Identity Provider section of the table
- this Oracle Identity Federation server as an IdP, and remote SPs from the Service Provider section of the table
- this Oracle Identity Federation server as an IdP, and affiliations from the Affiliation section of the table

The display will **not** include federations between remote IdPs and affiliations when this Oracle Identity Federation server acts as an SP and is participating in the affiliations.

Provider ID

This is the ID of a service provider or identity provider in the Oracle Identity Federation circle of trust.

Description

This is a brief description of the provider.

Version

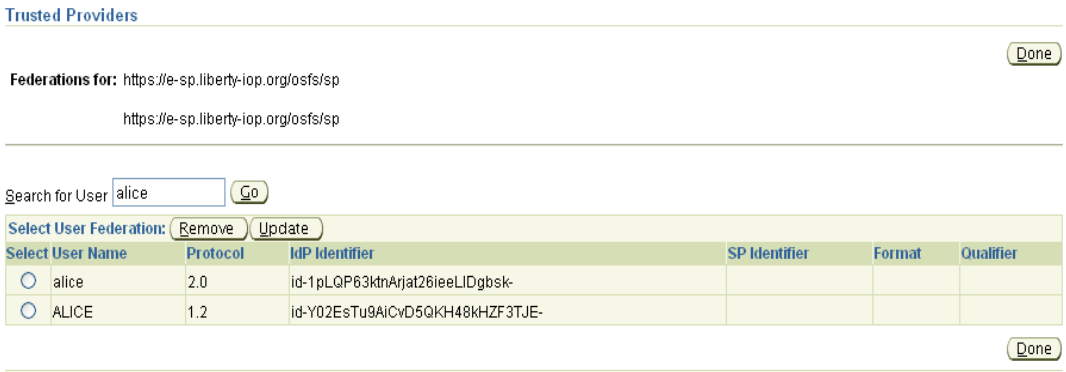
- This is the protocol version:
  - 1.1 (Liberty 1.1)
  - 1.2 (Liberty 1.2)
  - 2.0 (SAML 2.0)

Federations for [Provider]

This page allows you to search for and display users who have identity federations associated with a given trusted provider.

Enter a (partial or complete) name in the **Search for user** box, and click **Go**.

This example looks up user Alice:



Note: You can use substrings for user searches. In the example, entering either `al` or `alice` would return user `alice`; the search for `al` would return any additional users whose name contained that substring.

User Name

This is the local name of a federation user.

Protocol

This is the federation protocol for this user.

IdP Identifier

This is a value that the identity provider generates for the IdP name identifier.

SP Identifier

This is a value that the service provider generates for the SP name identifier.

Format

This is the NameID (Name Identifier) format. Example 1–2 on page 1-10 illustrates the use of name identifiers for SAML 2.0.

Qualifier

This is the NameQualifier attribute of a name identifier. This qualifier is used to avoid name collisions - for example, two users from different domains with the same username. (The qualifier therefore serves the same purpose as XML namespaces and Java package names.)

Actions

> **Note:** For all protocols, the **Remove** button works the same way. However, the **Update** action is protocol-dependent.

Buttons on the page provide these actions:

- **Remove** - Clicking on **Remove** initiates a federation termination protocol exchange between the Oracle Identity Federation instance and the trusted provider. When the exchange (FedTerm, RNI, MNI) completes, you are sent to an updated version of this page.
- **Update** - Actions resulting from this button depend on the associated protocol.
  - *Liberty 1.1 and Liberty 1.2*

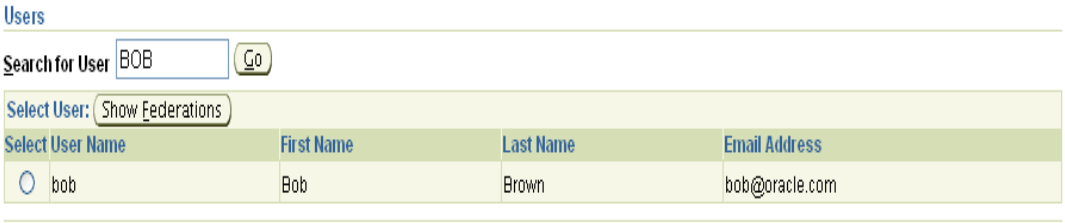
    Clicking on **Update** initiates a Register Name Identifier protocol exchange between Oracle Identity Federation and the trusted provider. The Federation Server will generate a new value for the name identifier.
  - *SAML 2.0*

    Clicking on **Update** initiates a Manage Name Identifier protocol exchange between Oracle Identity Federation and the trusted provider. If the Name Identifier Format is `urn:oasis:names:tc:SAML:2.0:nameid-format:persistent`, the Federation Server generates a new value for the name identifier. If the format is one of the other, non-opaque types, such as `urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress`, the Federation Server will take the new name identifier value from the corresponding attribute in the user repository.

Users

The Users page is accessed by choosing Users from the **Identity Federation** tab on the Administration Console. Enter all or part of a user name in the **Search for user** box and click **Go**.

The page allows you to view basic data defined for the Oracle Identity Federation user.



Enter a (partial or complete) name in the **Search for User** box, and click **Go**.

Note:

- You can use substrings for user searches. In the example, entering either `BOB` or `B` would return user `bob`; the search for `B` would return any additional users whose name contained that substring.
- The data displayed by a search depends on how the user data is stored. For database user stores, only the User Description attribute is returned. For non-database user stores, the First Name, Last Name, and Email Address are displayed.

When viewing user information, click on the **Show Federations** button to view and update federations associated with that user.

User Name

This is username assigned to the user.

First Name

This is the user's first name.

Last Name

This is the user's last name.

Email Address

This is the user's e-mail address.

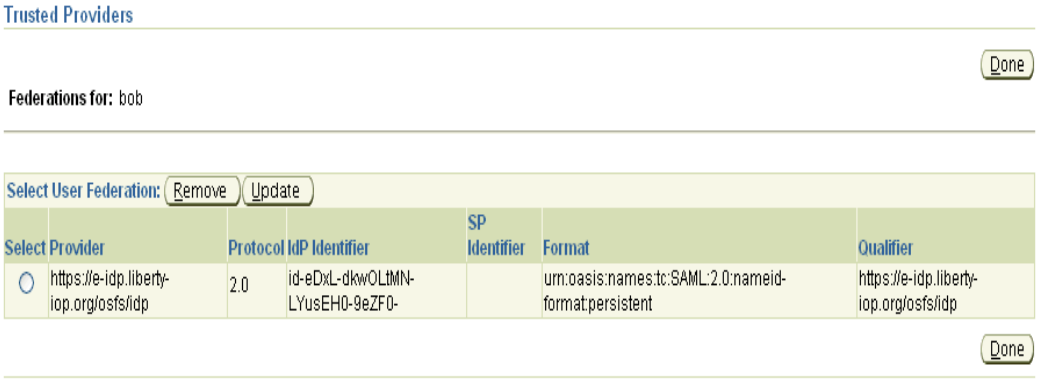
Federations for a User

When viewing user information, click on the **Show Federations** button to view and update federations associated with that user.

For a given user, the server displays federations between:

- this Oracle Identity Federation server as an SP, and remote IdPs
- this Oracle Identity Federation server as an IdP, and remote SPs
- this Oracle Identity Federation server as an IdP, and affiliations

- affiliations in which this Oracle Identity Federation server (as an SP) is a member, and remote IdPs



Provider

This is the provider name.

Protocol

This is the provider's federation protocol for this user.

IdP Identifier

This is a value that the identity provider generates for the IdP name identifier.

SP Identifier

This is a value that the service provider generates for the IdP name identifier.

Format

This is the SAML NameID (Name Identifier) format.

Qualifier

This is the provider URL, applicable for service providers.

Buttons on the page provide these actions:

- **Remove** - Clicking on **Remove** initiates a federation termination protocol exchange between the Oracle Identity Federation instance and the trusted provider. When the exchange (FedTerm, RNI, MNI) completes, you are sent to an updated version of this page.
- **Update** - Actions resulting from this button depend on the associated protocol.
  - *Liberty 1.1 and Liberty 1.2*

    Clicking on **Update** initiates a Register Name Identifier protocol exchange between Oracle Identity Federation and the trusted provider. Oracle Identity Federation will generate a new value for the name identifier.
  - *SAML 2.0*

    Clicking on **Update** initiates a Manage Name Identifier protocol exchange between Oracle Identity Federation and the trusted provider. If the Name Identifier Format is

urn:oasis:names:tc:SAML:2.0:nameid-format:persistent, the Federation Server generates a new value for the name identifier. If the format is one of the other, non-opaque types, such as urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress, the Federation Server will take the new name identifier value from the corresponding attribute in the user repository.

Reassociation

At some point in time you may need to point your Oracle Identity Federation instance to a different Infrastructure - for example, when Oracle Identity Federation is currently deployed in a test environment and is now ready to enter the production environment. This section describes key aspects of reassociation and contains the following topics:

- Changing the Federation Data Store
- Changing the User Data Store
- Changing the RDBMS Data Store
- Deleting Federation Data
- Changing the Oracle Access Manager Instance
- Deleting Policy Objects from Oracle Access Manager

Note: Existing federation data is not migrated during reassociation. To ensure that all federation data is erased, you must delete federation data records as the first step in reassociation.

See Also: For detailed instructions about pointing your Oracle Identity Federation instance to a different instance of Oracle Internet Directory or Oracle Application Server Single Sign-On, see "Reassociating the Server" on page 3-16

Changing the Federation Data Store

The procedure for changing the federation data store used by an Oracle Identity Federation instance involves these steps:

1. Update the server's LDAP schema to be compatible with the Oracle Identity Federation LDAP data type.

   LDIF Schema files for the supported directories are located in the $ORACLE_HOME/fed/setup/ldap folder of the Oracle Identity Federation installation.

2. Point your browser to the Oracle Identity Federation Administration Console.
3. Select **IdM Data Stores** - > **Edit Federation Data Store**.
4. Update the federation data store settings.

   See Also: "Edit Federation Data Store" on page 6-64

Changing the User Data Store

The procedure for changing the user data store that is used by an Oracle Identity Federation instance involves these steps:

1. Point your browser to the Oracle Identity Federation Administration Console.

2. Select **IdM Data Stores - > Edit User Data Store**.
3. Update the user data store settings.

See Also: "Edit User Data Store" on page 6-67

Changing the RDBMS Data Store

An RDBMS can be used to store transient data as well as identity data.

To change the RDBMS data store used by an Oracle Identity Federation instance, you will need to run the configuration assistant described in "Command-Line Configuration Assistant to Change the Transient Data Store" on page 9-9.

Note: You will need to invoke the configuration assistant with the `-transient rdbms` argument.

See Also: "Configuring an RDBMS as the User Data Store" on page 6-80.

Deleting Federation Data

You can delete federation data records or transient session data that were generated by an Oracle Identity Federation instance; it is also possible to remove the LDAP schema from the directory server used as the federation data store.

To delete federation data, delete transient data, or remove the LDAP schema, you will need to run the command-line configuration assistant described in "Command-Line Configuration Assistant for Uninstallation" on page 9-10.

Note: This configuration assistant does not uninstall Oracle Identity Federation; it only cleans up the data stores. You will need to invoke this configuration assistant before you change the existing federation and user data stores to ensure that all data is erased.

Changing the Oracle Access Manager Instance

To move Oracle Identity Federation from a test to a production Oracle Access Manager, simply repeat the Oracle Access Manager setup steps:

1. Configure an AccessGate in the production instance of Oracle Access Manager, reconfigure the Oracle Access Manager user data store for Oracle Identity Federation and the associated AccessGate configuration, and restart.
2. If, for some reason, the Access Server SDK that was installed in Oracle Identity Federation for the test environment is not compatible with the production environment (for example, it is an older version), uninstall the SDK and install a compatible SDK.

Deleting Policy Objects from Oracle Access Manager

This section provides instructions on how to delete Oracle Identity Federation-related policy objects from an Oracle Access Manager instance. This is needed if there was a previous installation of Oracle Identity Federation using Oracle Access Manager, which is being replaced by a new Oracle Identity Federation instance on a different host.

Take these steps:

1. If it is still running, stop the OC4J_FED instance for the previous Oracle Identity Federation server.
2. Log into the Access System Console as a Master Access Administrator.
3. Click on the **Access Manager** link to get the Policy Manager My Domains page.
4. Delete the policy domain named **Fed Domain**.
5. Change the authentication schemes for any policy domains that use the "Fed SSO" schemes. For each domain:
   - On the My Domains page, click on the domain name to get to the domain page.
   - Click on the **Default Rules** tab to get the default authentication rule for the domain.
   - If the authentication scheme for the rule is Fed SSO - SAML 1.x, Fed SSO - WS-Federation, or Fed SSO - SAML 2.0/Liberty 1.2:
     - Record the authentication scheme for this domain.
     - Click **Modify**.
     - Select another authentication scheme, preferably a basic scheme, from the drop-down list.
     - Click **Save** to return to the domain page.
   - Click on the **Policies** tab.
   - For each policy defined for the domain:
     - Click on the policy name to get the policy page.
     - Click on the **Authentication Rule** tab.
     - If the authentication scheme for the rule is Fed SSO - SAML 1.x, Fed SSO - WS-Federation, or Fed SSO - SAML 2.0/Liberty 1.2:

       - Record the authentication scheme for this policy for this domain.

       - Click **Modify**.

       - Select another authentication scheme, preferably a basic scheme, from the drop down list.

       - Click **Save** to go back to the policy page.

       - Click **Policies**.
6. Click on the Access System Console link.
7. Click the **Access System Configuration** tab.
8. Click on the Authentication Management link.
9. Delete all authentication schemes that begin with "Fed". This includes the Fed SSO - SAML 1.x, Fed SSO - WS-Federation, and Fed SSO - SAML 2.0/Liberty 1.2 schemes, and any scheme created for a configured SAML 1.x assertion mapping, such as Fed SASSO Mapping or Fed Minimal Mapping.
10. Configure the Oracle Access Management user data store, and restart the new Oracle Identity Federation server. Oracle Identity Federation will automatically recreate the Access policy objects.

11. Restore the Fed SSO authentication schemes for the policy domains and policies recorded in step 5. For each recorded policy domain:
    - On the My Domains page, click on the domain name to get the domain page.
    - If the domain originally used a Fed SSO scheme for its default authentication rule, click the **Default Rules** tab.
      - click **Modify**.
      - Select the original Fed SSO scheme from the drop-down list.
      - click **Save**.
    - Click the **Policies** tab.
    - For each policy in the domain that originally used a "Fed SSO" scheme:
      - Click the policy name to get the policy page.
      - Click the **Authentication Rule** tab.
      - Click **Modify**.
      - Select the original Fed SSO scheme from the drop-down list.
      - Click **Save**.
      - Click **Policies**.

Un-installing Oracle Identity Federation

This section explains how to uninstall an Oracle Identity Federation instance. It contains these topics:

Overview of Un-installation

Un-installing Oracle Identity Federation requires these high-level actions:

- Run the Oracle Identity Federation `uninstall` tool.
- Run the Oracle Application Server instance `deconfig` tool.
- Run Oracle Universal Installer for product deinstallation.
- Clean up any remaining files.

Table 5–2 shows what items need to be removed and the relevant tool for each item.

Table 5–2 Oracle Identity Federation Items to De-install

De-installed Item	Tool Used
Files from the `ORACLE_HOME` directory	Oracle Universal Installer If the installer cannot remove all the files, you can remove any remaining files using `rm` or an equivalent operating system command.
Entries for the deleted instance in the `Inventory` directory	Oracle Universal Installer
Instance name from the Farm page	Oracle Universal Installer
Entries for the deleted instance in the `/var/opt/oracle` directory	Remove these entries manually.
Entries for the deleted instance in Oracle Internet Directory	`deconfig` tool
Federation records	`uninstall` tool

Subsequent sections describe uninstallation steps and explain how to use these tools to complete uninstallation.

Uninstallation Steps

Oracle Universal Installer provides the ability to de-install products. It does not permit custom uninstallation of individual components.

Follow these uninstallation steps:

1. Log in as the operating system user who installed the instance that is to be un-installed.
2. Stop all processes associated with this instance.

   See the *Oracle Application Server Administrator's Guide* for details on how to stop the processes.
3. Run the Oracle Identity Federation `uninstall` tool:

   ```
   java -jar ORACLE_HOME/fed/lib/uninstall.jar <parameters>
   ```

   For an explanation of parameters and other details about the `uninstall` tool, see "Command-Line Configuration Assistant for Uninstallation" on page 9-10.
4. If an OracleAS Single Sign-On back-end is associated with the Oracle Identity Federation instance, run the Oracle Application Server `deconfig` tool:

   ```
   cd $ORACLE_HOME/bin
   $ORACLE_HOME/perl/bin/perl deconfig.pl <parameters>
   ```

   See Also: For details of the `deconfig` tool parameters, see "Oracle Application Server Instance Deconfig Tool"
5. Start Oracle Universal Installer:

   ```
   $ORACLE_HOME/oui/bin/runInstaller
   ```

If you are unable to start Oracle Universal Installer in this way, or when you select **Start** - > **Programs** - > **Oracle - OracleHomeName** - > **Oracle Installation Products** - > **Universal Installer**, run these commands at the system prompt to start the installer:

```
% cd ORACLE_HOME\oui\bin
```

```
% setup.exe -J-Dsun.java2d.noddraw=true
-Dsun.awt.nopixfmt=true
```

6. Take these steps in Oracle Universal Installer:
   a. On the Welcome screen, click **Deinstall Products**.
   b. On the Inventory screen, select the instance you want to uninstall, and click **Remove**.
   c. On the Confirmation screen, verify the components selected for uninstallation. Click **Yes** to continue.
   d. Monitor progress on the Uninstallation Progress screen.
   e. Exit the installer when the uninstallation is complete.
7. Delete any remaining files in the `ORACLE_HOME` directory of the deleted instance using an appropriate operating system command. For example:

   ```
   rm -rf $ORACLE_HOME
   ```

8. Remove the line for the un-installed middle tier from the `/var/opt/oracle/oratab` file.

   Towards the end of the file, you should see lines that specify the `ORACLE_HOME` directory. Remove the line for the `ORACLE_HOME` that you un-installed. For example, if your `ORACLE_HOME` is `/private1/oif`, the line would look like the following:

   ```
   *:/private1/oif:N
   ```

Uninstall Error Messages

You may see `unable to delete file` and `unable to find make file` errors in the `oraInstalltimestamp.err` file after you uninstall Oracle Identity Federation server instances. For example:

```
Ignoring Exception during de-install
oracle.sysman.oii.oiil.OiilDeinstallException:
An error occurred during runtime. oracle.sysman.oii.oiil.OiilDeinstallException:
An error occurred during runtime.
...
Ignoring Exception during de-install
oracle.sysman.oii.oiil.OiilDeinstallException:
Unable to delete file
/home/j2ee/sysman/emd/targets.xml
oracle.sysman.oii.oiil.OiilDeinstallException: Unable to delete file
/home/j2ee/sysman/emd/targets.xml
at instantiateFileEx.deinstallAction(instantiateFileEx.java:935)
...
Ignoring Exception during de-installoracle.sysman.oii.oiil.OiilDeinstallException:
Unable to find make file:
/home/j2ee/network/lib/ins_net_client.mk
oracle.sysman.oii.oiil.OiilDeinstallException: Unable to find make file:
/home/j2ee/network/lib/ins_net_client.mk
at ssmakeux.deinstallAction(ssmakeux.java:246)
```

```
...
```

These are harmless error messages and may safely be ignored.

Oracle Application Server Instance Deconfig Tool

The Oracle Application Server `deconfig` tool removes entries in the OracleAS Metadata Repository and Oracle Internet Directory for the Oracle Application Server instance that you are un-installing. If an OracleAS Single Sign-On back-end is associated with the Oracle Identity Federation instance, run the `deconfig` tool before de-installing from the Oracle Universal Installer.

To run the `deconfig` tool, run the Perl interpreter on the `ORACLE_HOME/bin/deconfig.pl` script. Use the Perl interpreter provided with Oracle Identity Federation:

```
cd $ORACLE_HOME/bin

$ORACLE_HOME/perl/bin/perl deconfig.pl <parameters>
```

If you run it without any parameters, the tool prompts you for the necessary information.

Deconfig Tool Syntax and Parameters

The syntax for the `deconfig` tool is:

```
$ORACLE_HOME/perl/bin/perl deconfig.pl [-u oid_user] [-w password]
   [-r realm] [-dbp sys_db_password]
```

You can also run the `deconfig` tool with the `-h` or `-help` parameter to display help:

```
$ORACLE_HOME/perl/bin/perl deconfig.pl -h
```

or

```
$ORACLE_HOME/perl/bin/perl deconfig.pl -help
```

The parameters are:

- `-u` specifies the Oracle Internet Directory user.

  You can specify the value using either the user's simple name or the user's distinguished name (DN). For example, the user's simple name can be jdoe@mycompany.com, which corresponds to the DN `cn=jdoe,l=us,dc=mycompany,dc=com`.

  The Oracle Internet Directory user needs to have privileges for un-installing the components that are configured in the Oracle Application Server instance that you are uninstalling. These privileges are the same as for installing and configuring the component.

  If you want to run the tool as the Oracle Internet Directory superuser, be sure to use `cn=orcladmin`, and not just `orcladmin`. Note that these are two different users. Both users are created when you install Oracle Internet Directory; `cn=orcladmin` is the Oracle Internet Directory superuser. For more information about this topic, see the *Oracle Internet Directory Administrator's Guide*.

- `-w` specifies the password for the Oracle Internet Directory user.
- `-r` specifies the realm in which to authenticate the user. This value is required only if your Oracle Internet Directory has more than one realm.

- `-dbp` is deprecated and is not needed, so do not specify this parameter.

Deconfig Tool Log Files

The `deconfig` tool writes its log file to the `ORACLE_HOME/cfgtoollogs/DeconfigureWrapper.log` file.

Un-installing OracleAS Cold Failover Cluster Installations

Take these steps to uninstall an OracleAS Cold Failover Cluster installation:

1. Stop the clusterware agents or packages that monitor and fail over the environment. See your clusterware documentation for details.
2. Perform the steps described in "Uninstallation Steps" on page 5-16.

If you do not follow this order and take the resources offline first, the installer will hang during the uninstallation because the clusterware agents will try to fail over the resources.

Cleaning Up Oracle Application Server Processes

If you forgot to shut down Oracle Identity Federation server processes before starting the installation, you will need to kill the processes manually because the files for these processes have been deleted. To check for processes that are still running, run an operating system command such as the Unix `ps` command:

```
% ps -ef
```

Note the process_id obtained from the command, then kill the process using a command such as the Unix `kill` command:

```
kill -9 process_id
```

If you need to shut down the `dcmctl` shell process, try exiting the shell by typing `exit`.

Reinstallation

Oracle Universal Installer does not allow you to reinstall an Oracle Identity Federation server instance in a directory that already contains an Oracle Identity Federation server instance. To reinstall Oracle Identity Federation server in the same directory, you must uninstall and then install it.

6

Configuring Oracle Identity Federation

This chapter explains how to use the Oracle Identity Federation administration console to configure and maintain server properties and federation data. It contains these sections:

Data Maintained by Oracle Identity Federation

The Oracle Identity Federation administrator acquires the data needed to manage and operate the server from a variety of sources, including third parties (other providers' administrators), agreements with the third parties, and from local configuration decisions. The administrator is responsible for loading and maintaining this information in the federation server.

Broadly speaking, the federation server maintains two categories of configuration details:

- Server Configuration Data, which includes properties that determine the runtime behavior of a federation server instance
- User Federation Data, including details about individual users' federated identities and usage information

Server Configuration Data

Each Oracle Identity Federation instance maintains two types of configuration data:

- Protocol data, including:
  - properties of the server instance as a whole, including the hostname and port, whether SSL is enabled, the PKCS #12 wallet location, and so on

  - how the server instance supports its enabled federation protocols when acting as an identity provider, including session time-outs, re-authentication time-outs, the default provider ID, and so on
  - how the server instance supports its enabled federation protocols when acting as a service provider. The data maintained in this case is very similar to the data stored when the server acts as an identity provider
- Circle of Trust data, consisting of information about peer providers that are members of the Circle of Trust to which this server instance belongs. Circle of Trust configuration data includes such parameters as:
  - name ID formats to use for assertions
  - attributes to send along with an authentication request
  - signing requirements for assertions and authentication requests
  - preferred bindings
  - validity periods of assertions and artifacts
  - other time-related parameters such as the allowable time difference between servers that are not synchronized.
  - account linking parameters

Configuration Settings and Provider Metadata

Note that relationships may exist between configuration settings and the provider metadata that the server generates. Some settings do not affect the metadata while others do. For example, changing the Session Timeout value does not affect the metadata, but changing the SOAP Port will require the administrator to re-publish his metadata to the other providers in his circle of trust. Likewise, the administrator must be aware of changes to peer providers' metadata.

Here is a list of properties that affect metadata:

- Server Properties
  - Server Hostname
  - Server Port
  - SOAP Port
  - IdP Enabled
  - SP Enabled
  - SSL Enabled
  - Signing PKCS #12 Wallet
  - Encryption PKCS #12 Wallet
- Global IdP Properties
  - ProviderID
  - Liberty 1.1 Enabled
  - Liberty 1.2 Enabled
  - SAML 2.0 Enabled
- Global SP Properties
  - ProviderID

  - Liberty 1.1 Enabled
  - Liberty 1.2 Enabled
  - SAML 2.0 Enabled
- Liberty 1.1 IdP Properties
  - Enable Profiles/Bindings
  - Federation Termination Enabled
  - Register NameID Enabled
- Liberty 1.2 IdP Properties
  - Enable Protocol Profiles
  - Federation Termination Enabled
  - Register NameID Enabled
- SAML 2.0 IdP Properties
  - Enable Protocol Profiles
  - Federation Termination Enabled
  - Register NameID Enabled
  - Attribute Responder Enabled
- Liberty 1.1 SP Properties
  - Enable Profiles/Bindings
  - Federation Termination Enabled
  - Register NameID Enabled
- Liberty 1.2 SP Properties
  - Enable Protocol Profiles
  - Federation Termination Enabled
  - Register NameID Enabled
- SAML 2.0 SP Properties
  - Enable Protocol Profiles
  - Federation Termination Enabled
  - Register NameID Enabled

The metadata URLs for the various protocols are as follows:

- IdP SAML 2.0 - `http(s)://hostname:port/fed/idp/metadatav20`
- IdP Liberty 1.2 - `http(s)://hostname:port/fed/idp/metadatav12`
- IdP Liberty 1.1 - `http(s)://hostname:port/fed/idp/metadatav11`
- SP SAML 2.0 - `http(s)://hostname:port/fed/sp/metadatav20`
- SP Liberty 1.2 - `http(s)://hostname:port/fed/sp/metadatav12`
- SP Liberty 1.1 - `http(s)://hostname:port/fed/sp/metadatav11`

See Also: For information about SAML 1.x and WS-Federation service URLs, see "Exchanging SAML 1.x and WS-Federation Configuration Data with Peers" on page 6-97

User Federation Data

An LDAP directory stores each user's identity federation data. In addition to the user's basic reference information, there are records for each unique identity federation associated with the user. A federation record is defined by:

- the remote provider
- the name identifier type (for example, an e-mail address or a **DN**)
- the protocol (for example, SAML 2.0)

This means that a user can have multiple identity federation records for the same remote provider, so long as the combination of these three attributes provides uniqueness. For example, the user's first record could be identified by a combination of ProviderX/myemail1/SAML 2.0, and the second record by ProviderX/myemail2/SAML 2.0.

Synchronization

As mentioned earlier, the federation records for a user are stored independently, and rely on a unique user attribute (such as a DN or a username) to link to the main user record.

An event that changes a user's unique attribute value - for example, if an employee moves to a new office location and her DN is updated - requires that the user's federations be dropped and re-established.

Deprovisioning

Likewise, if a user record is deleted, the federation data remains. This means that the administrator must be sure to delete the user's federation data when the user is deprovisioned.

Caution: Failure to delete the federation data in this situation can introduce a potential security problem. For example, consider a scenario where a new user is subsequently provisioned with the same unique attribute value - for example, the same DN or username; that user would inherit the previous user's account linkages if they had been left around.

The federation data can be deleted:

- using the LDAP server's administration tools. For data stored in Oracle Internet Directory, see *Oracle Identity Management User Reference* to obtain more information.
- using a command-line utility provided with Oracle Identity Federation. For details, see "Command line Federation Delete Tool" on page 9-12.

Administration Console Overview

The administration console of Oracle Identity Federation is a web-based graphical interface that enables the administrator to administer, configure, and maintain both server properties and user federation data.

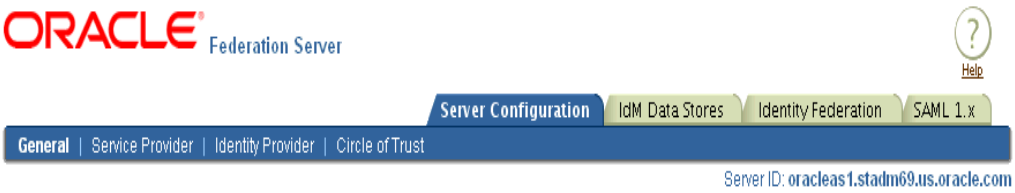
Basic Server Configuration

This section explains how to edit and update Oracle Identity Federation properties. It contains these sections:

- Server Configuration Tab
- Editing Server Properties
- Editing Global Properties
- Editing Protocol-specific IdP Properties
- Editing Protocol-specific SP Properties
- Service Provider - Attribute Requester
- Editing Circles of Trust
- Configuring and Using Affiliations
- Editing the Certificate Validation Store

Server Configuration Tab

Use the Server Configuration tab of the Administration Console to choose a category of Oracle Identity Federation properties to configure.



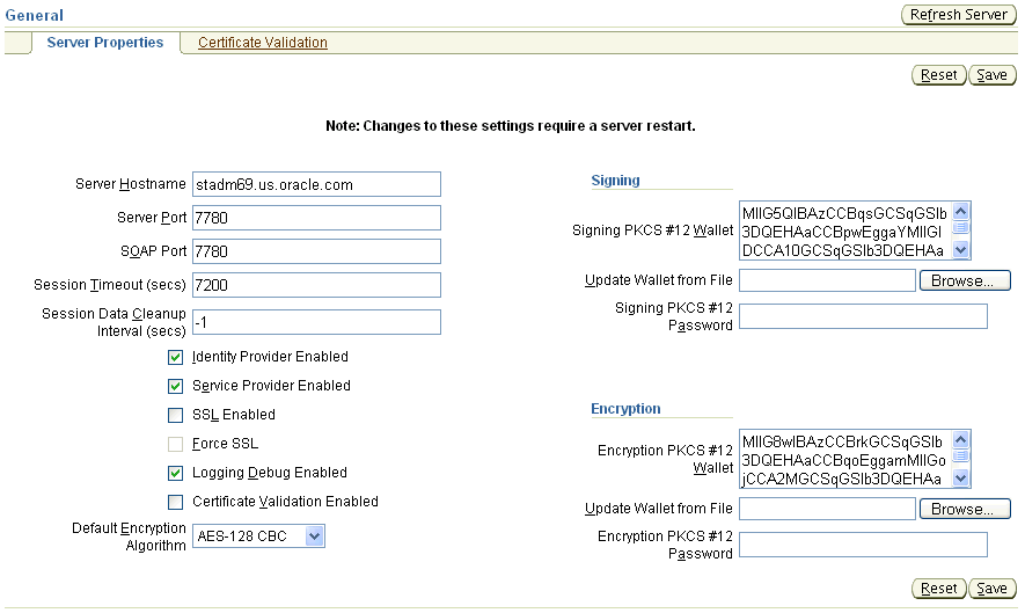
Select one of these configuration actions:

- Edit general server properties
- Edit global properties for a server acting as Identity Provider (IdP)
- Edit global properties for a server acting as Service Provider (SP)
- Edit protocol-specific properties and attribute responder mappings for a Liberty 1.1, Liberty 1.2, or SAML 2.0 compliant Service Provider (SP)
- Edit protocol-specific properties for a Liberty 1.1, Liberty 1.2, or SAML 2.0 compliant Identity Provider (IdP)
- Edit Circle of Trust (COT) provider information

Editing Server Properties

Use this page to edit Oracle Identity Federation properties.

Note: Any changes to these settings will require a server restart.



You can configure the following parameters:

Server Hostname

This is the host name of the Oracle Identity Federation instance.

Note: This property affects server metadata. When updating this property, distribute the updated metadata to all providers in your circle of trust.

Server Port

This is the port where Oracle Identity Federation listens.

Note:

- This setting only dictates what server port will be specified in the IdP and SP metadata when the metadata is generated. If there are several HTTP or HTTPS ports enabled for the OC4J instance in which Oracle Identity Federation is running, a user or peer provider can access Oracle Identity Federation through any of those ports, not just the port you specify here.
- This property affects server metadata. When updating this property, distribute the updated metadata to all providers in your circle of trust.

SOAP Port

This is the port where Oracle Identity Federation listens for SOAP messages.

Note:

- This setting only dictates what SOAP port will be specified in the IdP and SP metadata when the metadata is generated. If there are several HTTP or HTTPS ports enabled for the OC4J instance in which Oracle Identity Federation is running, a user or peer provider can access Oracle Identity Federation through any of those ports, not just the port you specify here.
- This property affects server metadata. When updating this property, distribute the updated metadata to all providers in your circle of trust.

Session Timeout

This parameter is used to determine the period, in seconds, for which an authenticated session is active. If the session remains inactive beyond the active period, the user must re-authenticate. The default value is 7200 seconds.

How this parameter is used depends on the server's role and the nature of the session in question.

Scenario 1: User Authenticated Locally

The user can be authenticated locally when:

- Oracle Identity Federation acts as an IdP
- Oracle Identity Federation is an SP, and the user needs to be prompted for its credentials because a new federation is being created

In this case, the expiration time of the authenticated session is set to the value of the Session Timeout parameter.

Scenario 2: Existing Federation

When Oracle Identity Federation is acting as an SP with an existing federation, the server receives a SAML assertion from the IdP containing user and authentication information. The assertion may include a `ReauthenticateOnOrAfter` attribute, indicating to Oracle Identity Federation that the user should be re-authenticated after the period specified by the attribute.

In this case, the Oracle Identity Federation server acting as SP sets the expiration time of the authenticated session to: the Session Timeout parameter or the `ReauthenticateOnOrAfter` assertion attribute, whichever is less.

Note: When Oracle Identity Federation uses Oracle Access Manager as its user data store, the Session Timeout has no effect on the user session. With Oracle Access Manager, the session timeout is determined by the configuration of the AccessGates protecting accessed resources.

Session Data Cleanup Interval

If Oracle Identity Federation is configured to use a relational database repository, this is the frequency with which the server cleans expired sessions from the database.

Note:

Identity Provider Enabled

Checking this box enables the server instance as an Identity Provider (IdP).

> **Note:** This property affects server metadata. When updating this property, distribute the updated metadata to all providers in your circle of trust.

Service Provider Enabled

Checking this box enables the server instance as a Service Provider (SP).

> **Note:** This property affects server metadata. When updating this property, distribute the updated metadata to all providers in your circle of trust.

SSL Enabled

Checking this box enables Secure Sockets Layer (SSL) encryption, allowing the server to listen in HTTPS mode.

> **Note:**
>
> - This property affects server metadata. When updating this property, distribute the updated metadata to all providers in your circle of trust.
> - This property only tells Oracle Identity Federation that the port on which the server is listening is SSL enabled; setting this property does not configure the SSL framework. Refer to these documents for details of how to enable SSL:
>
>   "Enabling SSL" in the *Oracle Application Server Quick Administration Guide*
>
>   "Configuring SSL" in the *Oracle HTTP Server Administrator's Guide*

Force SSL

Checking this box forces communications with the server to be conducted in HTTPS mode. If true, Oracle Identity Federation checks an incoming connection to ensure that it is done over SSL. If it is not, the server redirects the user to a URL supporting SSL; the URL is built with the `hostname` and `port` properties and the requested URL.

Logging Debug Enabled

Checking this box enables debug mode for Oracle Identity Federation and logs all operations.

If you do not check this box, debug mode is disabled, and only errors and incoming/outgoing Liberty and SAML messages are logged.

Certificate Validation Enabled

Check this box to enable validation of signing certificates using Certificate Revocation Lists (CRLs).

See Also: "Editing the Certificate Validation Store" on page 6-63

Fields for Signing Wallet

Use the following fields to assign a PKCS #12 signing certificate for the server instance:

- Signing PKCS #12 Wallet - This is the signing wallet in PKCS #12 format.

> **Note:** This property affects server metadata. When updating this property, distribute the updated metadata to all providers in your circle of trust.

- Update wallet from file - You can choose an operating system file containing the wallet.
- Signing PKCS #12 Password - Enter the password that was used to encrypt the private key.

Fields for Encryption Wallet

Use the following fields to assign a PKCS #12 encryption certificate for the server:

- Encryption PKCS #12 Wallet - This is the encryption wallet in PKCS #12 format.

> **Note:** This property affects server metadata. When updating this property, distribute the updated metadata to all providers in your circle of trust.

- Update wallet from file - You can choose an operating system file containing the wallet.
- Encryption PKCS #12 Password - Enter the password that was used to encrypt the private key.
- Default Encryption Algorithm - Select one of the available encryption algorithms:
  - AES-128 CBC
  - AES-192 CBC
  - AES-256 CBC
  - Triple DES CBC

The buttons at the bottom of the page perform the following functions:

- **Update** - saves changes made to server configuration properties.
- **Cancel** - restores the screen to its original values without saving any changes.

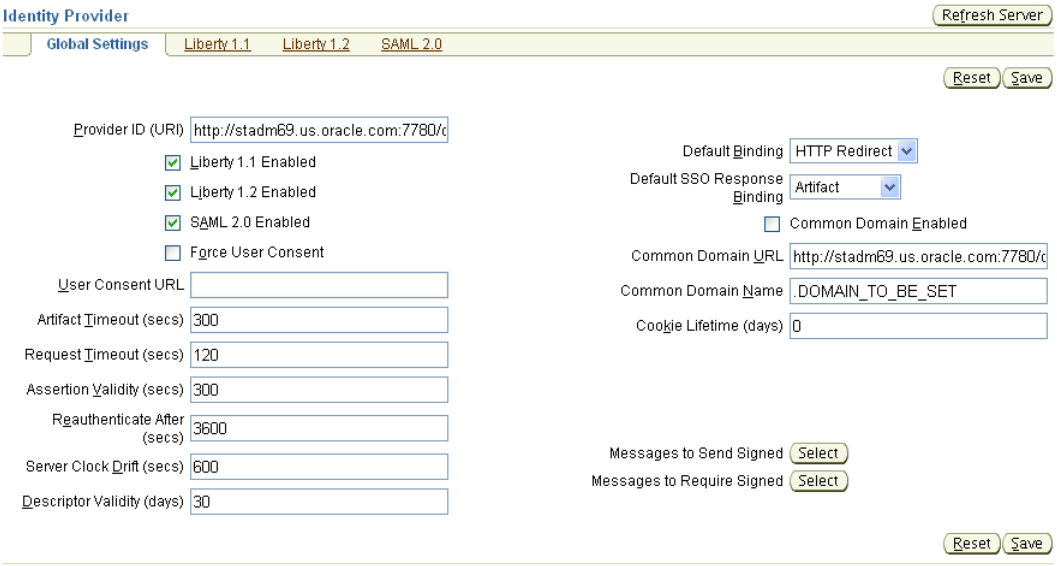
Editing Global Properties

This section explains how to accomplish the following tasks:

- Identity Provider - Global Settings
- Service Provider - Global Settings

Identity Provider - Global Settings

Use this page to specify global Identity Provider (IdP) properties for the SAML 2.0 and Liberty 1.x protocols for the server instance.



You can configure the following parameters:

Provider ID (URI)

This is the URI for the Oracle Identity Federation instance. If it is a URL, it need not point to an actual resource.

> **Note:** This property affects server metadata. When updating this property, distribute the updated metadata to all providers in your circle of trust.

Liberty 1.1 Enabled

Check this box to indicate that the server is enabled for Liberty 1.1.

> **Note:** This property affects server metadata. When updating this property, distribute the updated metadata to all providers in your circle of trust.

Liberty 1.2 Enabled

Check this box to indicate that the server is enabled for Liberty 1.2.

> **Note:** This property affects server metadata. When updating this property, distribute the updated metadata to all providers in your circle of trust.

SAML 2.0 Enabled

Check this box to indicate that the server is enabled for SAML 2.0.

> **Note:** This property affects server metadata. When updating this property, distribute the updated metadata to all providers in your circle of trust.

Force User Consent

Check this box to force consent for setting up a new federation. If this box is checked, a user who is redirected to the federation server will explicitly have to accept or deny account linking in order to proceed.

User Consent URL

The user is redirected to this URL if user consent is required. You must design a consent page for this purpose.

The server passes a number of query parameters to this URL:

Table 6–1 Parameters Passed to User Consent URL (IdP Global)

Parameter	Description
providerid	The peer provider id.
description	The description of the peer provider id.
returnurl	The URL to which the user should be directed once a consent decision has been made.
refid	Passed as a query parameter to the returnurl. Oracle Identity Federation require this parameter in order to resume the operation the server had been performing prior to redirection to the consent URL.

When the consent URL page directs the user back to the return URL (by way of a link, form submission, or other means) it must pass two query parameters: the `refid` parameter described in the table, and a consent parameter indicating if consent was granted by the user (values are true or false).

Here is an example of a consent page:

```
<%
    String prefix = request.getContextPath();
    String redirectURL = request.getParameter("returnurl");
    String refID = request.getParameter("refid");
    String providerID = request.getParameter("providerid");
    String desc = request.getParameter("description");
%>
<HTML>
<BODY>
Do you consent to create a federation with <%=providerID%> (<%=desc%>):<br>
<form method="POST" action="<%=redirectURL%>">
    <input type="checkbox" name="userconsent" value="true"/>I agree<br>
    <input type="submit" value="OK" />
    <input type="hidden" name="refid"  value="<%=refID%>"/>
</form>
</BODY>
</HTML>
```

Artifact Timeout

This is the validity time, in seconds, of an artifact object created by the Oracle Identity Federation. The default is 300 seconds.

Request Timeout

This is the validity time, in seconds, of an outgoing request from the Oracle Identity Federation. The default is 120 seconds.

Assertion Validity

This is the time, in seconds, during which an assertion issued by the identity provider is valid. An assertion is considered invalid if processed outside the validity period. The default is 300 seconds.

Reauthenticate After

This is the time, in seconds, after which the service provider must re-authenticate the user. Assertions containing an authentication statement by the identity provider are only valid for this period, after which the user is to be considered non-authenticated. The default is 3600 seconds.

Note: This feature is applicable to OracleAS Single Sign-On authentication. For a limitation on forced reauthentication, see "Deploying Oracle Identity Federation with OracleAS Single Sign-On" on page 4-2.

Server Clock Drift

This is the allowable time difference, in seconds, between Oracle Identity Federation, when acting as identity provider, and its peer servers. The default is 600 seconds.

Descriptor Validity

This is the time, in days, during which the server's published IdP metadata is considered valid. Peer servers can reload the metadata within this period. Enter the descriptor validity in days. The default value is 30 days.

Default Binding

Specifies the preferred binding to use, when possible, in sending messages to peer providers. Valid values are:

- HTTP Redirect
- HTTP POST
- SOAP

Default SSO Response Binding

Specifies the preferred binding for the identity provider to use, when possible, in sending an unsolicited assertion to the service provider. Valid values are:

- Artifact
- HTTP POST

Common Domain Enabled

When an identity federation network contains multiple identity providers, a service provider needs to have a way to determine the identity provider(s) in use by a principal. This is achieved by utilizing a domain that is common to IdPs and SPs in the federation network, and sending to the user's browser a cookie, written in this domain, that lists all the IdPs where the user is logged in. Such a domain is known as a common domain, and the cookie identifying the IdPs is called a common domain cookie or introduction cookie.

Check **Common Domain Enabled** to specify that this IdP should set the introduction cookie. After every local authentication, Oracle Identity Federation redirects the user

to the common domain, where the server can add its identifier to the introduction cookies at the user's browser.

Common Domain URL

When an identity federation network contains multiple identity providers, a domain common to all providers is a way for a service provider to determine the identity provider(s) in use by a principal. After every authentication, a cookie on the user's browser (written in this domain) is updated with the IdPs identifier; the cookie lists all the user's IdPs and can be read by the service provider.

Enter the URL where Oracle Identity Federation will read and set the IdP introduction cookie. The server listens on this URL, accepts requests, and updates the introduction cookie in the user's browser.

Set this value only if you checked **Common Domain Enabled**.

Common Domain Name

This is the common domain used for the IdP introduction cookie. It will be set as a cookie parameter on the introduction cookie. The value must begin with a dot (.) and must be of the form `.domain.suffix`. The default value is `.DOMAIN_TO_BE_SET`.

Cookie Lifetime

This is the lifetime, in days, of a Common Domain cookie issued by the IdP. If this field is set to 0 (default), the Common Domain Cookie will be a session cookie.

Messages to Send Signed

Click **Select** to display a list of federation protocol message types and specify the messages that Oracle Identity Federation sends, in IdP mode, that it must sign.

See Also: "Identity Provider - Select Messages to Send Signed"

Messages to Require Signed

Click **Select** to display a page which lists the federation protocol message types and specify the messages that Oracle Identity Federation receives, in IdP mode, that it requires to be signed.

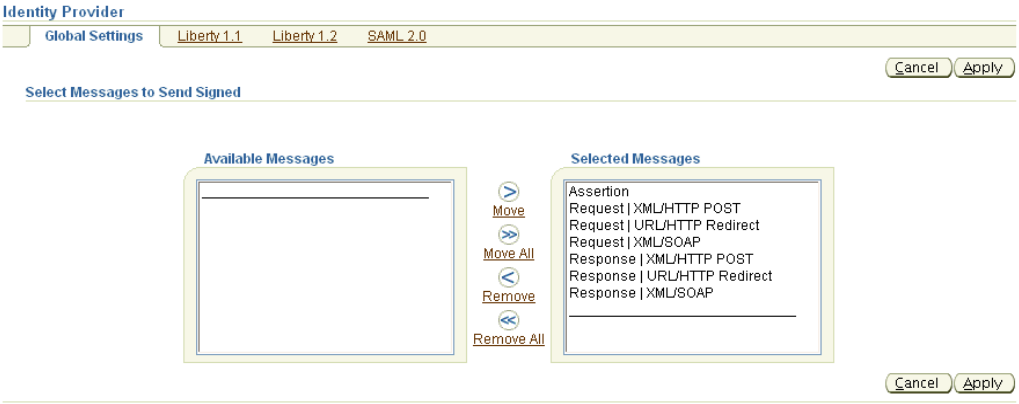
See Also: "Identity Provider - Select Messages to Require Signed"

The buttons at the bottom of the page perform the following functions:

- **Update** - saves changes made to Global IdP configuration properties.
- **Cancel** - Restores the screen to its original values without saving any changes.

Identity Provider - Select Messages to Send Signed

This page displays a list of federation protocol message types for messages sent by Oracle Identity Federation in Identity Provider mode. Use this page to specify which messages the server should sign. The messages are displayed along with their transport bindings.



Available Messages

Messages in this list will be sent unsigned.

Selected Messages

Messages in this list will be sent signed.

Actions

Buttons on the page provide these actions:

- **Move** - Moves the selected message types to the **Selected Messages** list. Use the control key to select multiple message types.
- **Move All** - Moves all messages currently in the **Available Messages** list to the **Selected Messages** list.
- **Remove** - Moves the selected message types from the **Selected Messages** list to the **Available Messages** list. Use the control key to select multiple message types.
- **Remove All** - Moves all messages currently in the **Selected Messages** list to the **Available Messages** list.

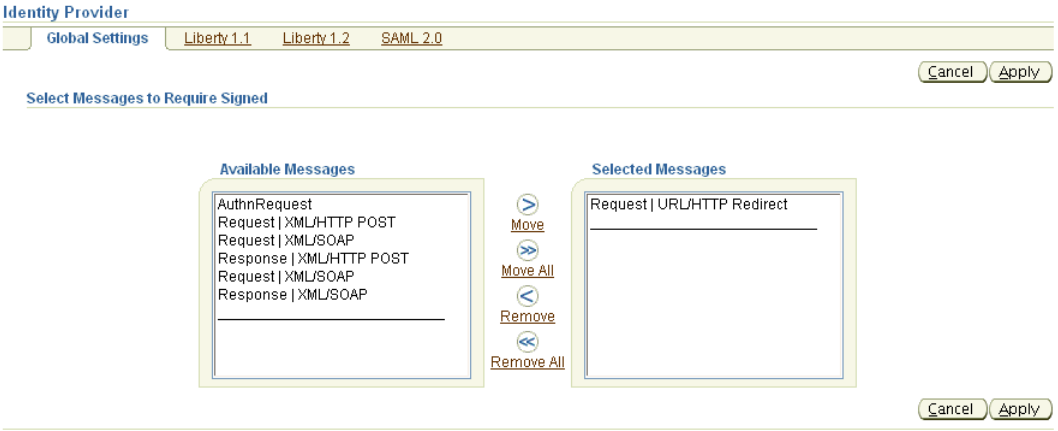
You can also double-click on a message type to move it from one list to the other.

The buttons at the bottom of the page perform the following functions:

- **Update** - saves changes made to the page for signed/unsigned messages.
- **Cancel** - Returns to the Identity Provider - Global Settings page without saving any changes.

Identity Provider - Select Messages to Require Signed

This page displays a list of federation protocol message types for messages received by Oracle Identity Federation in Identity Provider mode. Use this page to specify which types of messages the server requires to be signed. The messages are displayed along with their transport bindings.



Available Messages

Messages in this list are not required to be signed.

Selected Messages

Messages in this list are required to be signed.

Actions

Buttons on the page provide these actions:

- **Move** - Moves the selected message types to the **Selected Messages** list. Use the control key to select multiple message types.
- **Move All** - Moves all messages currently in the **Available Messages** list to the **Selected Messages** list.
- **Remove** - Moves the selected message types from the **Selected Messages** list to the **Available Messages** list. Use the control key to select multiple message types.
- **Remove All** - Moves all messages currently in the **Selected Messages** list to the **Available Messages** list.

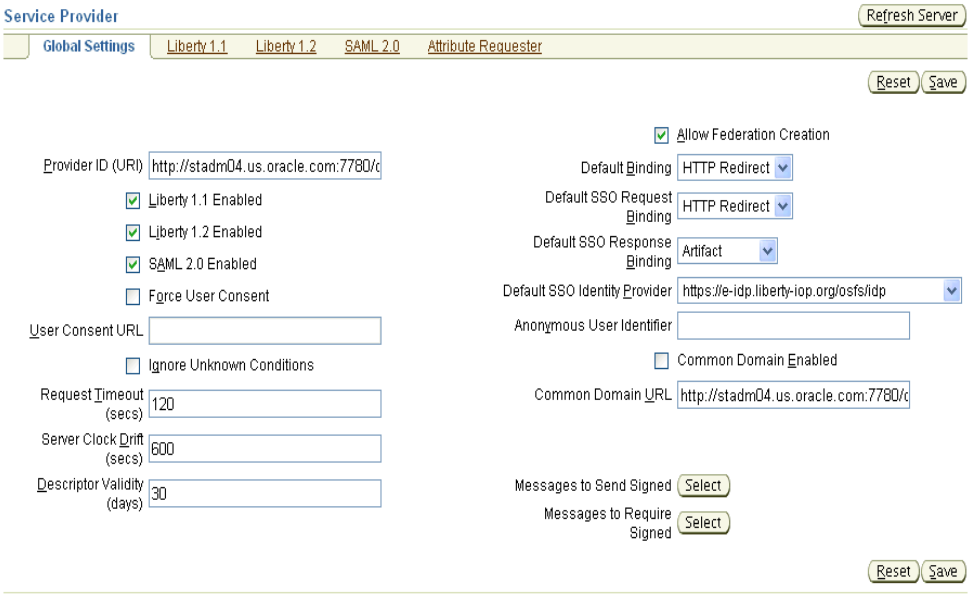
You can also double-click on a message type to move it from one list to the other.

The buttons at the bottom of the page perform the following functions:

- **Update** - saves changes made to the page for signed/unsigned messages.
- **Cancel** - Returns to the Identity Provider - Global Settings page without saving any changes.

Service Provider - Global Settings

Use this page to specify global Service Provider (SP) properties for the SAML 2.0 and Liberty 1.x protocols for the server instance.



You can configure the following parameters:

Provider ID (URI)

This is the URI for the Oracle Identity Federation instance. If it is a URL, it need not point to an actual resource.

> **Note:** This property affects server metadata. When updating this property, distribute the updated metadata to all providers in your circle of trust.

Liberty 1.1 Enabled

Check this box to indicate that the server is enabled for Liberty 1.1.

> **Note:** This property affects server metadata. When updating this property, distribute the updated metadata to all providers in your circle of trust.

Liberty 1.2 Enabled

Check this box to indicate that the server is enabled for Liberty 1.2.

> **Note:** This property affects server metadata. When updating this property, distribute the updated metadata to all providers in your circle of trust.

SAML 2.0 Enabled

Check this box to indicate that the server is enabled for SAML 2.0.

> **Note:** This property affects server metadata. When updating this property, distribute the updated metadata to all providers in your circle of trust.

Force User Consent

Check this box to force consent for setting up a new federation. If this box is checked, a user who is redirected to the federation server will explicitly have to accept or deny account linking in order to proceed.

User Consent URL

If the user must consent to setting up a new federation, this is the URL to which the user is redirected. You must design a consent page for this purpose.

The server passes a number of query parameters to this URL:

Table 6–2 Parameters Passed to User Consent URL (SP Global)

Parameter	Description
providerid	The peer provider id.
description	The description of the peer provider id.
returnurl	The URL to which the user should be directed once a consent decision has been made.
refid	Passed as a query parameter to the returnurl. Oracle Identity Federation require this parameter in order to resume the operation the server had been performing prior to redirection to the consent URL.

When the consent URL page directs the user back to the return url (by way of a link, form submission, or other means) it must pass two query parameters: the `refid` parameter described in the table, and a consent parameter indicating if consent was granted by the user (values are true or false).

Use this field only if **Force User Consent** is checked.

Here is an example of a consent page:

```
<%
    String prefix = request.getContextPath();
    String redirectURL = request.getParameter("returnurl");
    String refID = request.getParameter("refid");
    String providerID = request.getParameter("providerid");
    String desc = request.getParameter("description");
%>
<HTML>
<BODY>
Do you consent to create a federation with <%=providerID%> (<%=desc%>):<br>
<form method="POST" action="<%=redirectURL%>">
    <input type="checkbox" name="userconsent" value="true"/>I agree<br>
    <input type="submit" value="OK" />
    <input type="hidden" name="refid"  value="<%=refID%>"/>
</form>
</BODY>
</HTML>
```

Ignore Unknown Conditions

A condition is an extension point in the XML schema. Custom conditions can be defined according to specific needs - for example, to denote that assertion X is conditional in a given time zone. Such conditions may not be amenable to evaluation, and checking this box allows the server to ignore a condition it does not recognize in a received message.

Request Timeout

This is the validity time, in seconds, of an outgoing request from Oracle Identity Federation.

Server Clock Drift

This is the allowable time difference, in seconds, between Oracle Identity Federation, when acting as identity provider, and its peer servers.

Descriptor Validity

This is the time, in days, during which the server's published SP metadata is considered valid. Peer servers can reload the metadata within this period. Enter the descriptor validity in days. The default value is 30 days.

Allow Federation Creation

Use this field if your server instance, when operating as a service provider, should allow federation when initiating single sign-on with its peer identity providers in the federated network. Check this box to allow federation to be set up between the providers.

Default Binding

Specifies the preferred binding to use, when possible, in sending messages to peer providers. Valid values are:

- HTTP Redirect
- HTTP POST
- SOAP

Default SSO Request Binding

Specifies the preferred binding for the service provider to use, when possible, in sending authentication requests to the identity provider. Valid values are:

- HTTP Redirect
- HTTP POST

Default SSO Response Binding

Specifies the preferred binding for the identity provider to use, when possible, in sending an unsolicited assertion to the service provider. Valid values are:

- Artifact
- HTTP POST

Default SSO Identity Provider

Enter the default IdP to use in performing the single sign-on operation. Choose from the list of configured identity providers, or blank for none.

Note: Oracle Identity Federation provides a second mechanism to allow a service provider to select the IdP that will receive an authentication request to initiate single sign-on: the URL that is requested from the partner application to initiate single sign-on can include a query parameter, `providerid`, that specifies the desired IdP. This mechanism is useful when there are multiple IdP's in the circle of trust.

Anonymous User Identifier

This field allows the SP to provide services without knowing the identity of the consumer. It is used to identify a user when using Liberty 1.2 onetime/anonymous identifier or SAML 2.0 transient name ID in a Single Sign-On operation. Oracle Identity Federation receives an assertion containing an anonymous identifier; in order to create an authenticated session for that user in the IdM framework, an account needs to be associated with that session. The Anonymous User Identifier references the account to be used for such operations.

These steps are necessary to complete the configuration of the anonymous user identifier:

1. On the Circle Of Trust tab, select an IdP and click **Update** to go to **Edit Trusted Provider**. Uncheck **Default Authn Request NameID Format [2]**, and from the drop-down box, select **Transient/One-Time Identifier**.
2. Uncheck **NameID Formats** and click **Select**. On **Edit Trusted Provider: Select NameID Formats**:
   - Check **Enable** for **Transient/One time Identifier**.
   - For **Default Assertion NameID Format [1]**, select **Transient/One-Time Identifier**.
3. Return to Service Provider - Global Settings, and in the **Anonymous User Identifier** field, enter the username to use for anonymous login. This is the account that the anonymous user will use to "log in" to the system.

Common Domain Enabled

When an identity federation network contains multiple identity providers, a domain common to all providers is a way for a service provider to determine the identity provider(s) in use by a principal.

Check this box to indicate that common domain is enabled for the federation network.

Common Domain URL

Enter the URL to which Oracle Identity Federation will redirect the user to read the IdP introduction cookie and initiate a single sign-on session with the last recognized IdP where the user logged in. The server listens on this URL, accepts requests, and reads the introduction cookies sent by the user's browser.

Messages to Send Signed

Click **Select** to display a list of federation protocol message types and specify the messages that Oracle Identity Federation sends that it must sign.

See Also: "Service Provider - Select Messages to Send Signed"

Messages to Require Signed

Click **Select** to display a page which lists the federation protocol message types and specify the messages that Oracle Identity Federation receives that it requires to be signed.

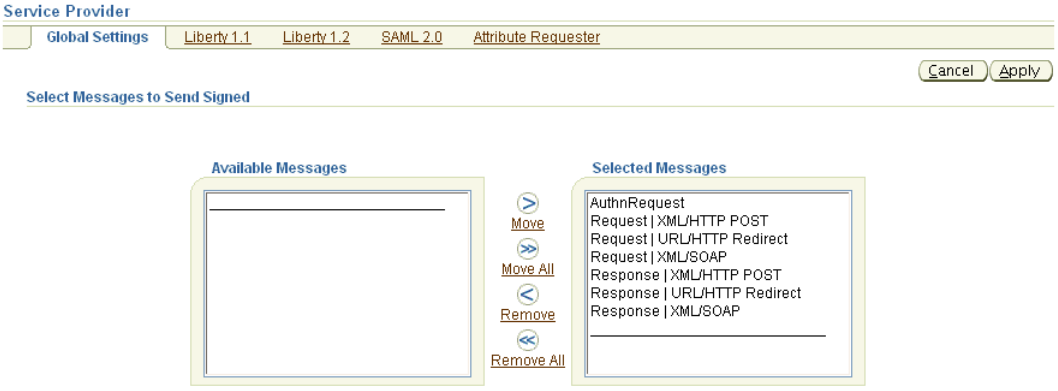
See Also: "Service Provider - Select Messages to Receive Signed"

The buttons at the bottom of the page perform the following functions:

- **Save** - saves changes made to Service Provider configuration properties.
- **Reset** - restores the screen to its original values without saving any changes.

Service Provider - Select Messages to Send Signed

This page displays a list of federation protocol message types for messages sent by Oracle Identity Federation in Service Provider mode. Use this page to specify which messages the server should sign. The messages are displayed along with their transport bindings.



Available Messages

Messages in this list will be sent unsigned.

Selected Messages

Messages in this list will be sent signed.

Actions

Buttons on the page provide these actions:

- **Move** - Moves the selected message types to the **Selected Messages** list. Use the control key to select multiple message types.
- **Move All** - Moves all messages currently in the **Available Messages** list to the **Selected Messages** list.
- **Remove** - Moves the selected message types from the **Selected Messages** list to the **Available Messages** list. Use the control key to select multiple message types.
- **Remove All** - Moves all messages currently in the **Selected Messages** list to the **Available Messages** list.

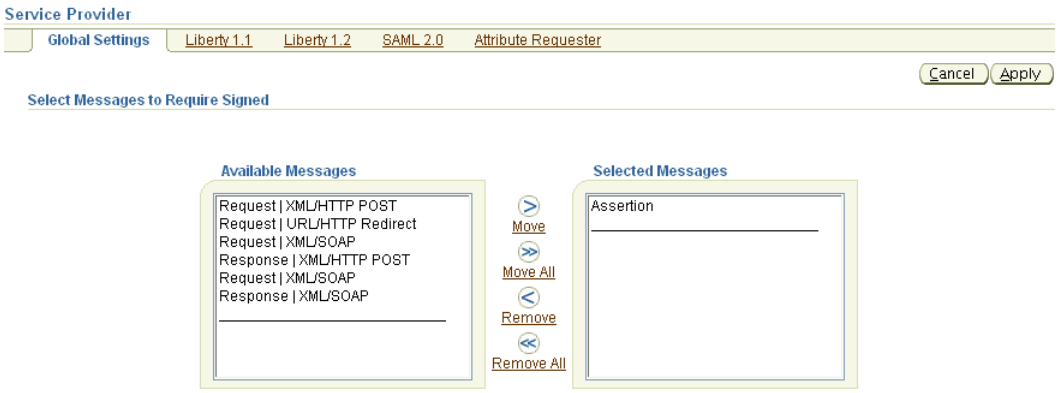
You can also double-click on a message type to move it from one list to the other.

The buttons at the bottom of the page perform the following functions:

- **Apply** - saves changes made to the page for signed/unsigned messages.
- **Cancel** - Returns to the Service Provider - Global Settings page without saving any changes.

Service Provider - Select Messages to Receive Signed

This page displays a list of federation protocol message types for messages received by Oracle Identity Federation in Service Provider mode. Use this page to specify which types of messages the server requires to be signed. The messages are displayed along with their transport bindings.



Available Messages

Messages in this list are not required to be signed.

Selected Messages

Messages in this list are required to be signed.

Actions

Buttons on the page provide these actions:

- **Move** - Moves the selected message types to the **Selected Messages** list. Use the control key to select multiple message types.
- **Move All** - Moves all messages currently in the **Available Messages** list to the **Selected Messages** list.
- **Remove** - Moves the selected message types from the **Selected Messages** list to the **Available Messages** list. Use the control key to select multiple message types.
- **Remove All** - Moves all messages currently in the **Selected Messages** list to the **Available Messages** list.

You can also double-click on a message type to move it from one list to the other.

The buttons at the bottom of the page perform the following functions:

- **Apply** - saves changes made to the page for signed/unsigned messages.
- **Cancel** - returns to the Service Provider - Global Settings page without saving any changes.

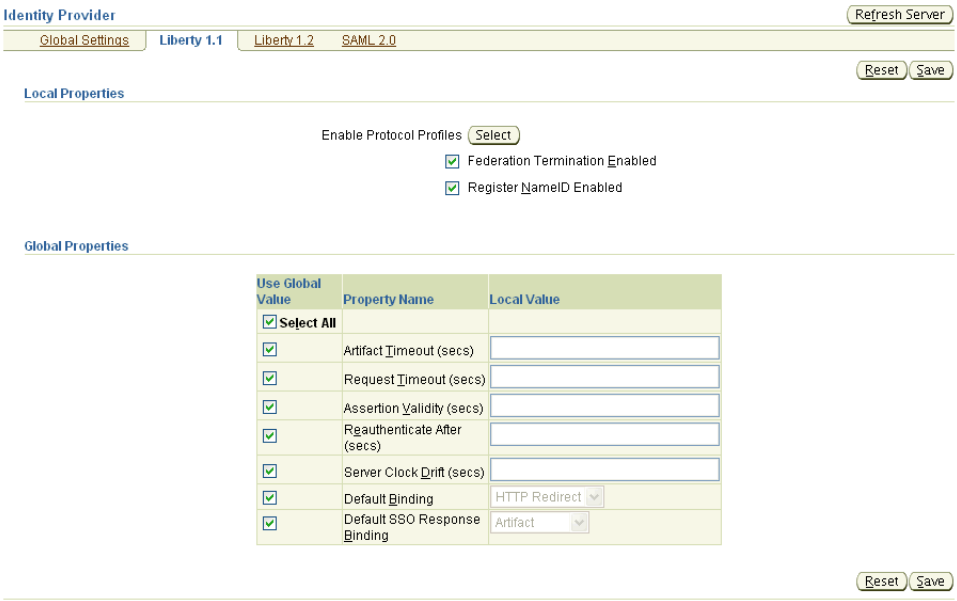
Editing Protocol-specific IdP Properties

This section describes how to edit and update the protocol-specific Identity Provider (IdP) properties in Oracle Identity Federation. It contains these sub-sections:

- Identity Provider - Liberty 1.1 Properties
- Enable Liberty 1.1 Identity Provider Profiles
- Identity Provider - Liberty 1.2 Properties
- Enable Liberty 1.2 Identity Provider Profiles
- Identity Provider - SAML 2.0 Properties
- Enable SAML 2.0 Identity Provider Profiles
- Select SAML 2.0 Identity Provider NameID Formats

Identity Provider - Liberty 1.1 Properties

Use this page to maintain IdP properties in the Liberty 1.1 protocol for the Oracle Identity Federation instance.



The page provides these options:

Enable Protocol Profiles

Click **Select** to choose the protocol profiles and transport bindings that you wish to enable for this federation server instance in IdP mode for Liberty 1.1.

Note: This property affects server metadata. When updating this property, distribute the updated metadata to all providers in your circle of trust.

See Also: "Enable Liberty 1.1 Identity Provider Profiles" on page 6-24

Federation Termination Enabled

Check this box to enable the federation termination capability.

See "Federation Termination Profile" on page 1-17 for an explanation of this feature.

> **Note:** This property affects server metadata. When updating this property, distribute the updated metadata to all providers in your circle of trust.

Register NameID Enabled

Check this box to enable name ID registration.

See "Name Identifier Profiles" on page 1-16 for an explanation of this feature.

> **Note:** This property affects server metadata. When updating this property, distribute the updated metadata to all providers in your circle of trust.

Use Global Value - Select All

Several IdP properties specific to Liberty 1.1 can be specified on the page. Check this box to specify that global IdP property settings will override the following local settings:

- Artifact Timeout
- Request Timeout
- Assertion Validity
- Reauthenticate After
- Server Clock Drift
- Default Binding
- Default SSO Response Binding

Use **Select All** in this way:

- Check the box to specify that global identity provider property settings override the local settings of all the listed properties.
- Uncheck the box to ensure that the local settings of all the properties override global settings.

To specify that the local setting of a given property overrides the global setting, uncheck the box for that property.

Artifact Timeout

This is the validity time, in seconds, of an artifact object created by the Oracle Identity Federation. The default is 300 seconds.

Request Timeout

This is the validity time, in seconds, of an outgoing request from the Oracle Identity Federation. The default is 120 seconds.

Assertion Validity

This is the time, in seconds, during which an assertion issued by the identity provider is valid. An assertion is considered invalid if processed outside the validity period. The default is 300 seconds.

Reauthenticate After

Assertions containing an authentication statement by the identity provider are only valid for a specific time, after which the user is considered non-authenticated. This is the time, in seconds, after which the service provider must re-authenticate the user. The default is 3600 seconds.

Note: This feature is applicable to OracleAS Single Sign-On authentication. For a limitation on forced reauthentication, see "Deploying Oracle Identity Federation with OracleAS Single Sign-On" on page 4-2.

Server Clock Drift

This is the time, in seconds, during which an assertion issued by the identity provider is valid. An assertion is considered invalid if processed outside the validity period. The default is 300 seconds.

Default Binding

Specifies the preferred binding to use, when possible, in sending messages to peer providers. Valid values are:

- HTTP Redirect
- HTTP POST
- SOAP

Default SSO Response Binding

Specifies the preferred binding for the identity provider to use, when possible, in sending an unsolicited assertion to the service provider. Valid values are:

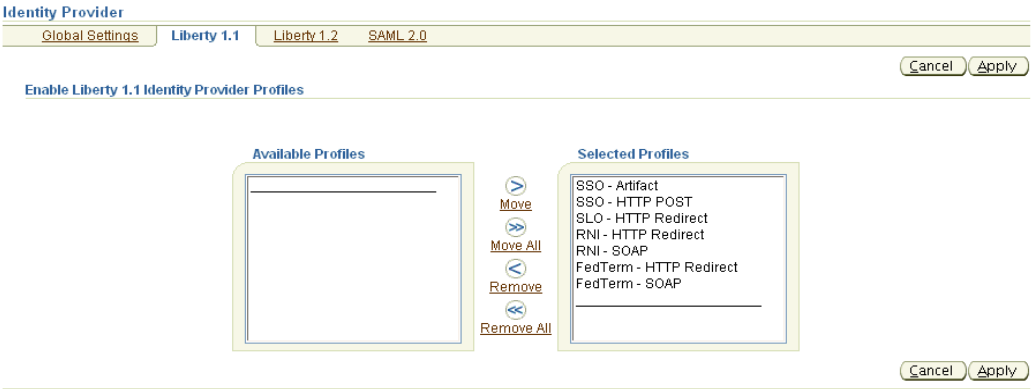
- Artifact
- HTTP POST

The buttons at the bottom of the page perform the following functions:

- **Save** - saves changes made to Identity Provider configuration properties.
- **Reset** - restores the screen to its original values without saving any changes.

Enable Liberty 1.1 Identity Provider Profiles

Use this page to select the Liberty 1.1 IdP profiles and bindings for the Oracle Identity Federation instance.



Available Profiles

Profiles and bindings in this list are disabled for the federation server instance.

Selected Profiles

Profiles and bindings in this list are enabled for the federation server instance.

Actions

Buttons on the page provide these actions:

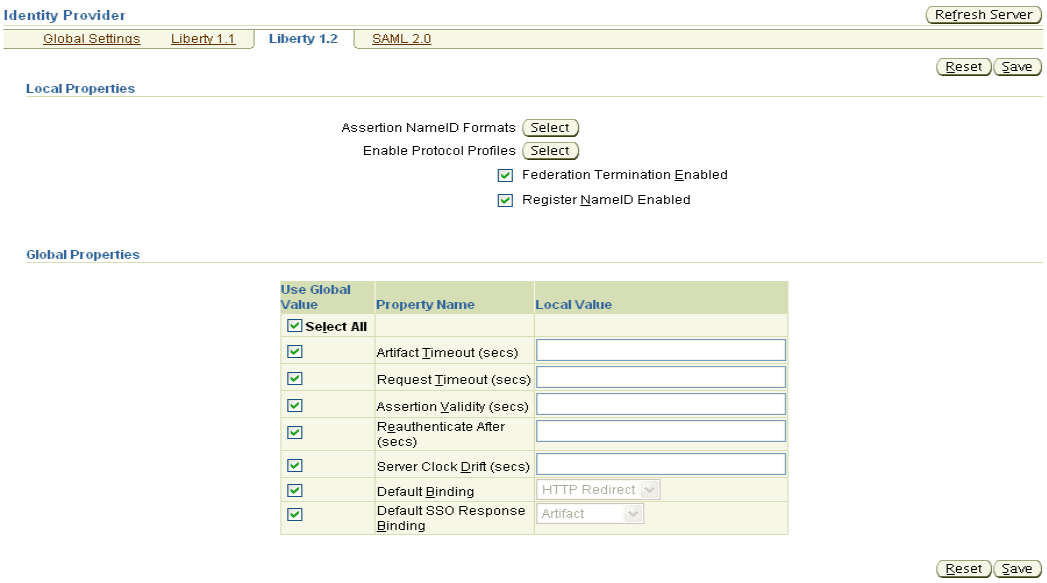
- **Move** - Moves the selected profiles/bindings to the **Selected Profiles** list. Use the control key to select multiple profiles/bindings.
- **Move All** - Moves all profiles/bindings currently in the **Available Profiles** list to the **Selected Profiles** list.
- **Remove** - Moves the selected profiles/bindings from the **Selected Profiles** list to the **Available Profiles** list. Use the control key to select multiple profiles/bindings.
- **Remove All** - Moves all profiles/bindings currently in the **Selected Profiles** list to the **Available Profiles** list.

The buttons at the bottom of the page perform the following functions:

- **Update** - saves changes made to the profiles/bindings.
- **Cancel** - returns you to the Identity Provider - Liberty 1.1 Properties page without saving any changes.

Identity Provider - Liberty 1.2 Properties

Use this page to maintain IdP properties in the Liberty 1.2 protocol for the Oracle Identity Federation instance.



The page provides these options:

Assertion NameID Formats

Click **Select** to choose the Liberty 1.2 protocol NameID formats that you wish to enable for this federation server instance in IdP mode.

> **See Also:** "Select Liberty 1.2 Identity Provider NameID Formats"

Enable Protocol Profiles

Click **Select** to choose the Liberty 1.2 protocol profiles and transport bindings that you wish to enable for this federation server instance in IdP mode.

> **Note:** This property affects server metadata. When updating this property, distribute the updated metadata to all providers in your circle of trust.

> **See Also:** "Enable Liberty 1.2 Identity Provider Profiles" on page 6-28

Federation Termination Enabled

Check this box to enable the federation termination capability.

See "Federation Termination Profile" on page 1-17 for an explanation of this feature.

> **Note:** This property affects server metadata. When updating this property, distribute the updated metadata to all providers in your circle of trust.

Register NameID Enabled

Check this box to enable name ID registration.

See "Name Identifier Profiles" on page 1-16 for an explanation of this feature.

Note: This property affects server metadata. When updating this property, distribute the updated metadata to all providers in your circle of trust.

The subsequent fields affect global properties.

Use Global Value - Select All

Several IdP properties specific to Liberty 1.2 can be specified on the page. Check this box to specify that global IdP property settings will override the following local settings:

- Artifact Timeout
- Request Timeout
- Assertion Validity
- Reauthenticate After
- Server Clock Drift
- Default Binding
- Default SSO Response Binding

Use **Select All** in this way:

- Check the box to specify that global IdP property settings override the local settings of all the above properties.
- Uncheck the box to ensure that the local settings of all the properties override global settings.

To specify that the local setting of a given property overrides the global setting, uncheck the box for that property.

Artifact Timeout

This is the validity time, in seconds, of an artifact object created by Oracle Identity Federation. The default is 300 seconds.

Request Timeout

This is the validity time, in seconds, of an outgoing request from the Oracle Identity Federation. The default is 120 seconds.

Assertion Validity

This is the time, in seconds, during which an assertion issued by the identity provider is valid. An assertion is considered invalid if processed outside the validity period. The default is 300 seconds.

Reauthenticate After

When the IdP creates an assertion containing an authentication statement, it sets a time limit for the validity of the authentication. This is the time, in seconds, after which the peer service provider should re-authenticate the user. The default is 3600 seconds.

Note: This feature is applicable to OracleAS Single Sign-On authentication. For a limitation on forced reauthentication, see "Deploying Oracle Identity Federation with OracleAS Single Sign-On" on page 4-2.

Server Clock Drift

This is the allowable time difference, in seconds, between Oracle Identity Federation, when acting as identity provider, and its peer servers. The default is 600 seconds.

Default Binding

Specifies the preferred binding to use, when possible, in sending messages to peer providers. Valid values are:

- HTTP Redirect
- HTTP POST
- SOAP

Default SSO Response Binding

Specifies the preferred binding for the identity provider to use, when possible, in sending an unsolicited assertion to the service provider. Valid values are:

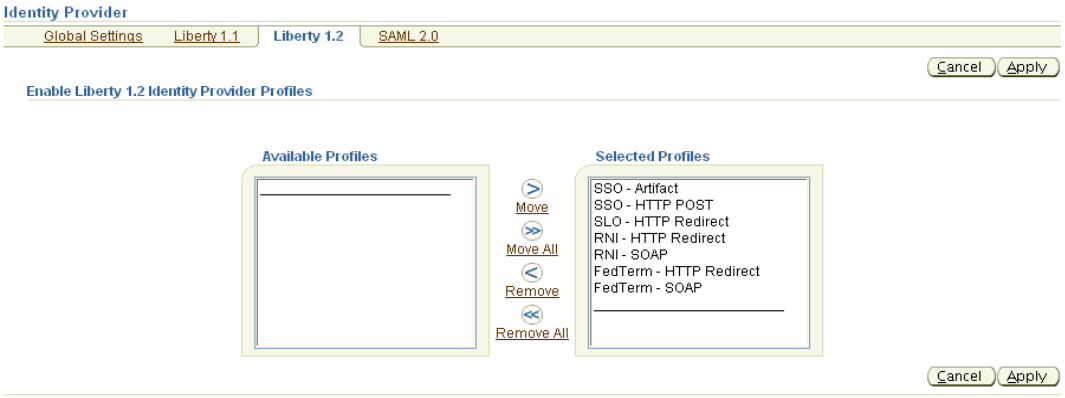
- HTTP POST
- SOAP

The buttons at the bottom of the page perform the following functions:

- **Save** - saves changes made to Identity Provider configuration properties.
- **Reset** - restores the screen to its original values without saving any changes.

Enable Liberty 1.2 Identity Provider Profiles

Use this page to select the Liberty 1.2 IdP profiles and bindings for the Oracle Identity Federation instance.



Available Profiles

Profiles and bindings in this list are disabled for the federation server instance.

Selected Profiles

Profiles and bindings in this list are enabled for the federation server instance.

Actions

Buttons on the page provide these actions:

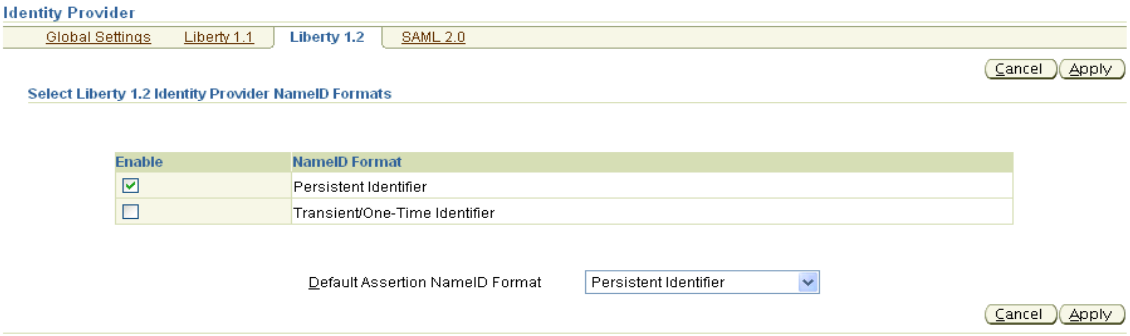
- **Move** - Moves the selected profiles/bindings to the **Selected Profiles** list. Use the control key to select multiple profiles/bindings.
- **Move All** - Moves all profiles/bindings currently in the **Available Profiles** list to the **Selected Profiles** list.
- **Remove** - Moves the selected profiles/bindings from the **Selected Profiles** list to the **Available Profiles** list. Use the control key to select multiple profiles/bindings.
- **Remove All** - Moves all profiles/bindings currently in the **Selected Profiles** list to the **Available Profiles** list.

The buttons at the bottom of the page perform the following functions:

- **Apply** - saves changes made to Service Provider configuration properties.
- **Cancel** - restores the screen to its original values without saving any changes.

Select Liberty 1.2 Identity Provider NameID Formats

Use this page to select the Liberty 1.2 IdP NameID formats for the Oracle Identity Federation instance.



On this page, you can:

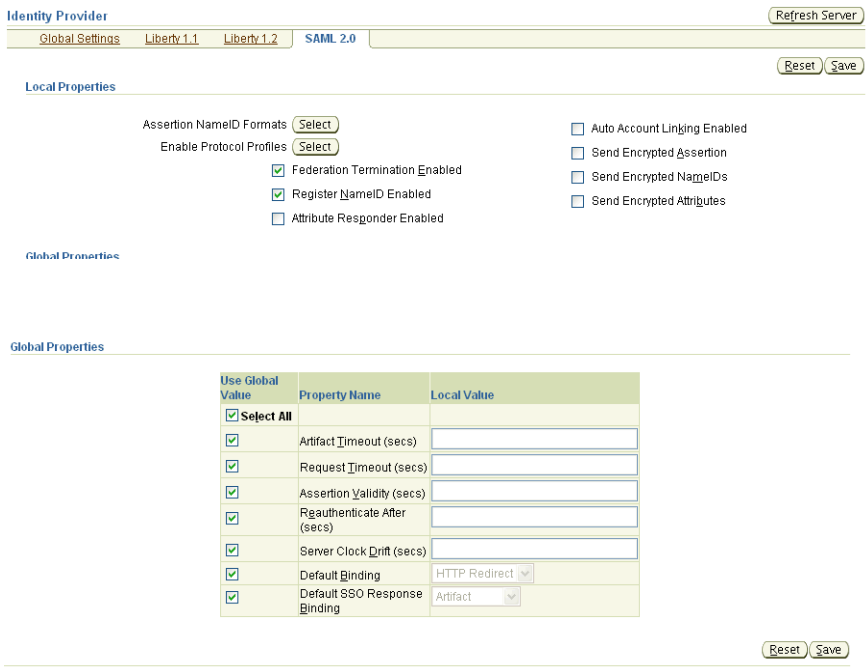
- enable one or more of the listed assertion NameID formats
- choose a default Assertion NameID format

The buttons at the bottom of the page perform the following functions:

- **Apply** - saves changes made to the list of enabled NameID formats.
- **Cancel** - restores the screen to its original values without saving any changes.

Identity Provider - SAML 2.0 Properties

Use this page to maintain IdP properties in the SAML 2.0 protocol for the Oracle Identity Federation instance.



The page provides these options:

Assertion NameID Formats

Click **Select** to choose the assertion NameID formats that you wish to enable for this federation server instance in IdP mode for SAML 2.0.

> **See Also:** "Select SAML 2.0 Identity Provider NameID Formats" on page 6-33

Enable Protocol Profiles

Click **Select** to choose the protocol profiles and transport bindings that you wish to enable for this federation server instance in IdP mode for SAML 2.0.

> **Note:** This property affects server metadata. When updating this property, distribute the updated metadata to all providers in your circle of trust.

> **See Also:** "Enable SAML 2.0 Identity Provider Profiles" on page 6-33

Federation Termination Enabled

Check this box to enable the federation termination capability.

See "Federation Termination Profile" on page 1-17 for an explanation of this feature.

> **Note:** This property affects server metadata. When updating this property, distribute the updated metadata to all providers in your circle of trust.

Register NameID Enabled

Check this box to enable name ID registration.

See "Name Identifier Profiles" on page 1-16 for an explanation of this feature.

> **Note:** This property affects server metadata. When updating this property, distribute the updated metadata to all providers in your circle of trust.

Auto Account Linking Enabled

Check this box to enable the auto account linking feature. When the IdP receives an authentication request containing a non-opaque nameid (for example, email address, X.500 name, and so on), it will look for a user with a matching nameid among its user accounts and automatically create a federation between the two user accounts if a federation does not already exist.

Send Encrypted Assertion

Check this box to enable Oracle Identity Federation to send encrypted assertions to peer providers.

Send Encrypted NameIDs

Check this box to enable Oracle Identity Federation to send encrypted name identifiers to peer providers.

Send Encrypted Attributes

Check this box to enable Oracle Identity Federation to send encrypted attributes to peer providers.

Attribute Responder Enabled

Check this box to enable attribute responder.

> **Note:** This property affects server metadata. When updating this property, distribute the updated metadata to all providers in your circle of trust.

Use Global Value - Select All

Several IdP properties specific to SAML 2.0 can be specified on the page. Check this box to specify that global IdP property settings will override the following local settings:

- Artifact Timeout
- Request Timeout
- Assertion Validity
- Reauthenticate After
- Server Clock Drift
- Default Binding
- Default SSO Response Binding

Use **Select All** in this way:

- Check the box to specify that global IdP property settings override the local settings of all the above properties.
- Uncheck the box to ensure that the local settings of all the properties override global settings.

To specify that the local setting of a given property overrides the global setting, uncheck the box for that property.

Artifact Timeout

This is the validity time, in seconds, of an artifact object created by the Oracle Identity Federation. The default is 300 seconds.

Request Timeout

This is the validity time, in seconds, of an outgoing request from the Oracle Identity Federation. The default is 120 seconds.

Assertion Validity

This is the time, in seconds, during which an assertion issued by the identity provider is valid. An assertion is considered invalid if processed outside the validity period. The default is 300 seconds.

Reauthenticate After

This is the time, in seconds, after which the service provider must re-authenticate the user. Assertions containing an authentication statement by the identity provider are only valid for this period, after which the user is to be considered non-authenticated. The default is 3600 seconds.

Note: This feature is applicable to OracleAS Single Sign-On authentication. For a limitation on forced reauthentication, see "Deploying Oracle Identity Federation with OracleAS Single Sign-On" on page 4-2.

Server Clock Drift

This is the allowable time difference, in seconds, between Oracle Identity Federation, when acting as identity provider, and its peer servers.

Default Binding

Specifies the preferred binding to use, when possible, in sending messages to peer providers. Valid values are:

- HTTP Redirect
- HTTP POST
- SOAP

Default SSO Response Binding

Specifies the preferred binding for the identity provider to use, when possible, in sending an unsolicited assertion to the service provider. Valid values are:

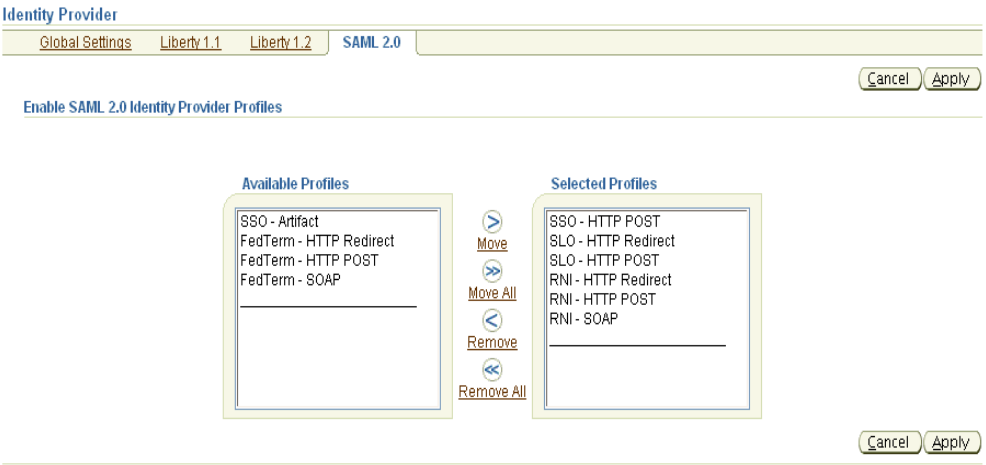
- Artifact
- HTTP POST

The buttons at the bottom of the page perform the following functions:

- **Update** - saves changes made to Identity Provider configuration properties.
- **Cancel** - restores the screen to its original values without saving any changes.

Enable SAML 2.0 Identity Provider Profiles

Use this page to maintain IdP profiles and bindings for the SAML 2.0 protocol.



Available Profiles

Profiles and bindings in this list are disabled for the federation server instance in IdP mode.

Selected Profiles

Profiles and bindings in this list are enabled for the federation server instance in IdP mode.

Actions

Buttons on the page provide these actions:

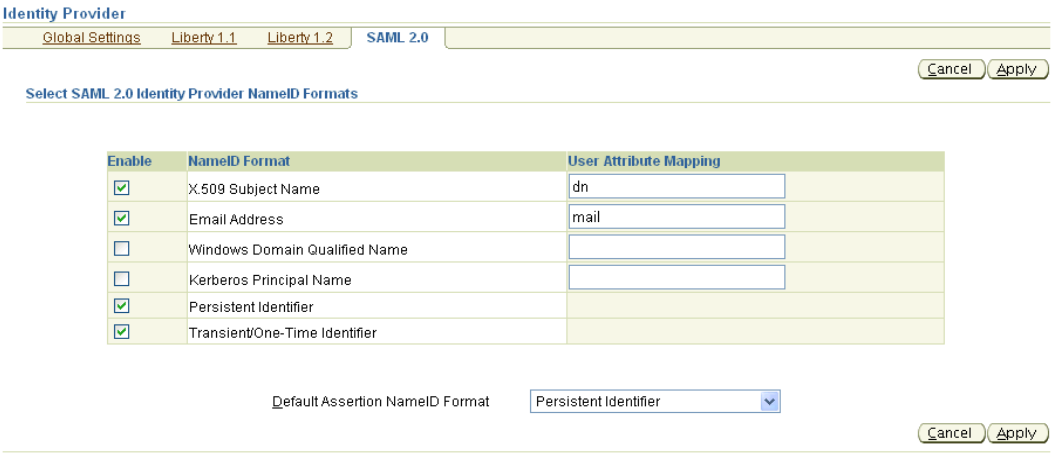
- **Move** - Moves the selected profiles/bindings to the **Selected Profiles** list. Use the control key to select multiple profiles/bindings.
- **Move All** - Moves all profiles/bindings currently in the **Available Profiles** list to the **Selected Profiles** list.
- **Remove** - Moves the selected profiles/bindings from the **Selected Profiles** list to the **Available Profiles** list. Use the control key to select multiple profiles/bindings.
- **Remove All** - Moves all profiles/bindings currently in the **Selected Profiles** list to the **Available Profiles** list.

The buttons at the bottom of the page perform the following functions:

- **Apply** - saves changes made to Identity Provider profile selections.
- **Cancel** - restores the screen to its original values without saving any changes.

Select SAML 2.0 Identity Provider NameID Formats

Use this page to maintain assertion name identifier formats for the SAML 2.0 protocol.



Check the corresponding box to enable the desired format(s) that the Oracle Identity Federation instance will use as the SAML 2.0 name identifier value in IdP mode. The formats are as follows:

Table 6–3 SAML 2.0 IdP NameID Formats

NameID Format	Default
X.509 Subject Name	DN
Email Address	e-mail
Windows Domain Qualified Name	empty
Kerberos Principal Name	empty
Persistent Identifier	
Transient/One-Time Identifier	

Default Assertion NameID Format

This is the SAML 2.0 assertion name identifier format to use when none is specified for the federation server in IdP mode.

The buttons at the bottom of the page perform the following functions:

- **Apply** - saves changes made to IdP NameID configuration properties.
- **Cancel** - returns you to Edit SAML 2.0 Identity Provider Properties without saving any changes.

Editing Protocol-specific SP Properties

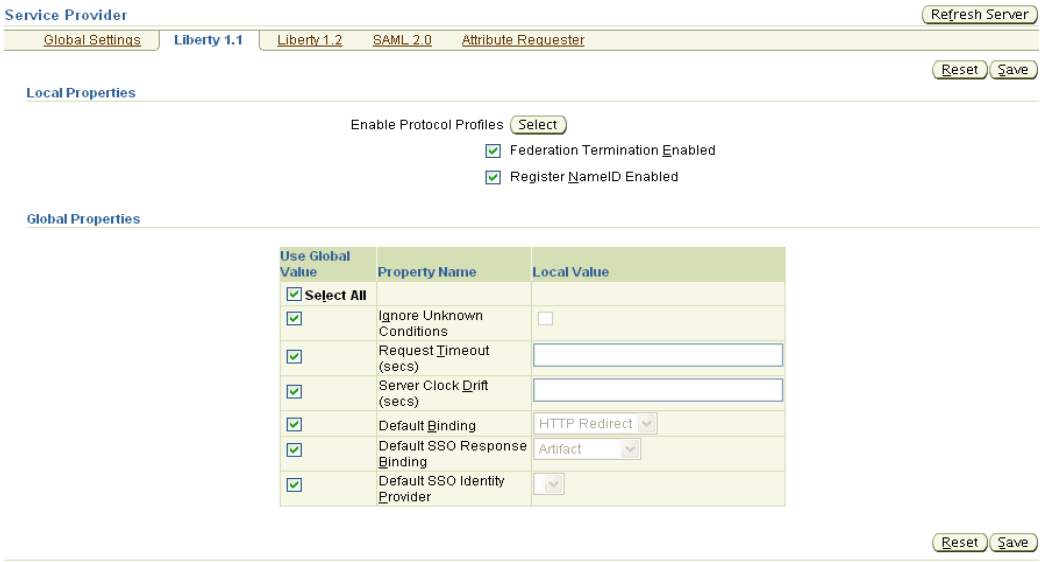
This section describes how to edit and update the protocol-specific Service Provider (SP) properties in Oracle Identity Federation. It contains these sub-sections:

- Service Provider - Liberty 1.1 Properties
- Enable Liberty 1.1 Service Provider Profiles
- Service Provider - Liberty 1.2 Properties
- Enable Liberty 1.2 Service Provider Profiles
- Service Provider - SAML 2.0 Properties

- Enable SAML 2.0 Service Provider Profiles
- Select SAML 2.0 Service Provider NameID Formats

Service Provider - Liberty 1.1 Properties

Use this page to maintain Service Provider properties for the Liberty 1.1 protocol.



The page provides these options:

Enable Protocol Profiles

Click **Select** to choose the protocol profiles and transport bindings that you wish to enable for this federation server instance in SP mode for Liberty 1.1.

Note: This property affects server metadata. When updating this property, distribute the updated metadata to all providers in your circle of trust.

See Also: "Enable Liberty 1.1 Service Provider Profiles" on page 6-37

Federation Termination Enabled

Check this box to enable the federation termination capability.

See "Federation Termination Profile" on page 1-17 for an explanation of this feature.

Note: This property affects server metadata. When updating this property, distribute the updated metadata to all providers in your circle of trust.

Register NameID Enabled

Check this box to enable name ID registration.

See "Name Identifier Profiles" on page 1-16 for an explanation of this feature.

Note: This property affects server metadata. When updating this property, distribute the updated metadata to all providers in your circle of trust.

Subsequent fields apply to global properties.

Use Global Value - Select All

Several SP properties specific to Liberty 1.1 can be specified on the page. Check this box to specify that global SP property settings will override the following local settings:

- Ignore Unknown Conditions
- Artifact Timeout
- Request Timeout
- Server Clock Drift
- Default Binding
- Default SSO Request Binding
- Default SSO Response Binding
- Default SSO Identity Provider

Use **Select All** in this way:

- Check the box to specify that global service provider property settings override the local settings of all the listed properties.
- Uncheck the box to ensure that the local settings of all the properties override global settings.

To specify that the local setting of a given property overrides the global setting, uncheck the box for that property.

Ignore Unknown Conditions

A condition is an extension point in the XML schema. Custom conditions can be defined according to specific needs - for example, to denote that assertion X is conditional in a given time zone. Such conditions may not be amenable to evaluation, and checking this box allows the server to ignore a condition it does not recognize.

Request Timeout

This is the validity time, in seconds, of an outgoing request from Oracle Identity Federation. The default is 120 seconds.

Server Clock Drift

This is the allowable time difference, in seconds, between Oracle Identity Federation, when acting as service provider, and its peer servers. The default is 600 seconds.

Default Binding

Specifies the preferred binding to use, when possible, in sending messages to peer providers. Valid values are:

- HTTP Redirect
- HTTP POST

- SOAP

Default SSO Request Binding

Specifies the preferred binding for the service provider to use, when possible, in sending authentication requests to the identity provider. Valid values are:

- HTTP Redirect
- HTTP POST

Default SSO Response Binding

Specifies the preferred binding for the identity provider to use, when possible, in sending an unsolicited assertion to the service provider. Valid values are:

- Artifact
- HTTP POST

Default SSO Identity Provider

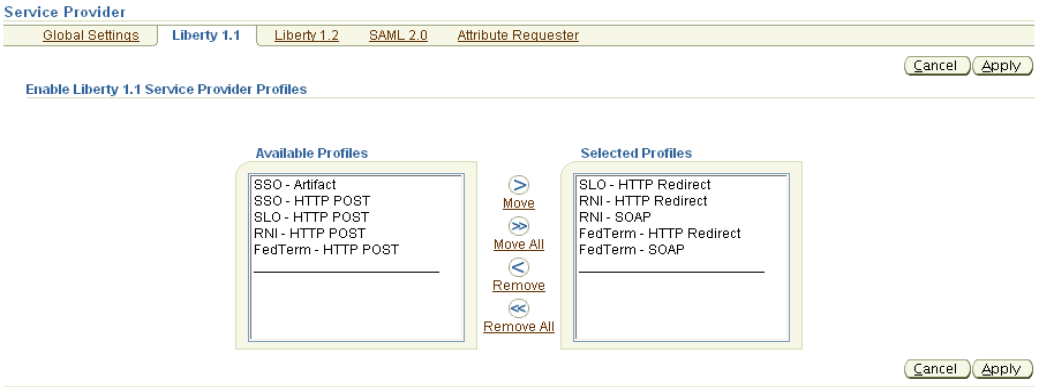
This is the default IdP to use in performing the single sign-on operation. Select a provider from the list of available IdPs.

The buttons at the bottom of the page perform the following functions:

- **Save** - saves changes made to Service Provider configuration properties.
- **Reset** - restores the screen to its original values without saving any changes.

Enable Liberty 1.1 Service Provider Profiles

Use this page to maintain Liberty 1.1 protocol profiles and bindings for Oracle Identity Federation in a service provider role.



Available Profiles

Profiles and bindings in this list are disabled for the federation server instance.

Selected Profiles

Profiles and bindings in this list are enabled for the federation server instance.

Actions

Buttons on the page provide these actions:

- **Move** - Moves the selected profiles/bindings to the **Selected Profiles** list. Use the control key to select multiple profiles/bindings.

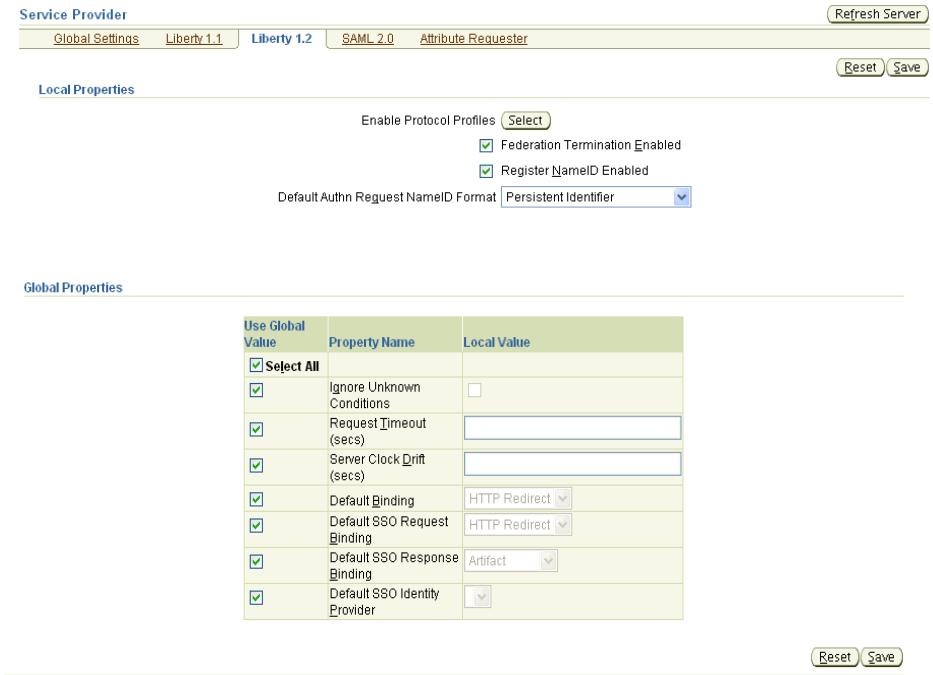
- **Move All** - Moves all profiles/bindings currently in the **Available Profiles** list to the **Selected Profiles** list.
- **Remove** - Moves the selected profiles/bindings from the **Selected Profiles** list to the **Available Profiles** list. Use the control key to select multiple profiles/bindings.
- **Remove All** - Moves all profiles/bindings currently in the **Selected Profiles** list to the **Available Profiles** list.

The buttons at the bottom of the page perform the following functions:

- **Apply** - saves changes made to Service Provider profiles/bindings.
- **Cancel** - returns you to Service Provider Liberty 1.1 Properties without saving any changes.

Service Provider - Liberty 1.2 Properties

Use this page to maintain Oracle Identity Federation properties as Service Provider in the Liberty 1.2 protocol.



The page provides these options:

Enable Protocol Profiles

Click **Select** to choose the protocol profiles and transport bindings that you wish to enable for this federation server instance in SP mode for Liberty 1.2.

Note: This property affects server metadata. When updating this property, distribute the updated metadata to all providers in your circle of trust.

See Also: "Enable Liberty 1.2 Service Provider Profiles" on page 6-40

Federation Termination Enabled

Check this box to enable the federation termination capability.

See "Federation Termination Profile" on page 1-17 for an explanation of this feature.

> **Note:** This property affects server metadata. When updating this property, distribute the updated metadata to all providers in your circle of trust.

Register NameID Enabled

Check this box to enable name ID registration.

See "Name Identifier Profiles" on page 1-16 for an explanation of this feature.

> **Note:** This property affects server metadata. When updating this property, distribute the updated metadata to all providers in your circle of trust.

Default Authn Request NameID Format

Use the list box to select a default name ID format for authentication requests. Choices are:

- Persistent identifier
- Transient/one-time identifier
- Unspecified

If set to Unspecified, the IdP will determine the format.

Subsequent fields relate to global SP settings.

Use Global Value - Select All

Several SP properties specific to Liberty 1.2 can be specified on the page. Check this box to specify that global SP property settings will override the following local settings:

- Ignore Unknown Conditions
- Artifact Timeout
- Request Timeout
- Server Clock Drift
- Default Binding
- Default SSO Request Binding
- Default SSO Response Binding
- Default SSO Identity Provider

Use **Select All** in this way:

- Check the box to specify that global service provider property settings override the local settings of all the listed properties.
- Uncheck the box to ensure that the local settings of all the properties override global settings.

To specify that the local setting of a given property overrides the global setting, uncheck the box for that property.

Ignore Unknown Conditions

A condition is an extension point in the XML schema. Custom conditions can be defined according to specific needs - for example, to denote that assertion X is conditional in a given time zone. Such conditions may not be amenable to evaluation, and checking this box allows the server to ignore a condition it does not recognize.

Request Timeout

This is the validity time, in seconds, of an outgoing artifact request from Oracle Identity Federation. The default is 120 seconds.

Server Clock Drift

This is the allowable time difference, in seconds, between Oracle Identity Federation, when acting as service provider, and its peer servers. The default is 600 seconds.

Default Binding

Specifies the preferred binding to use, when possible, in sending messages to peer providers. Valid values are:

- HTTP Redirect
- HTTP POST
- SOAP

Default SSO Request Binding

Specifies the preferred binding for the service provider to use, when possible, in sending authentication requests to the identity provider. Valid values are:

- HTTP Redirect
- HTTP POST

Default SSO Response Binding

Specifies the preferred binding for the identity provider to use, when possible, in sending an unsolicited assertion to the service provider. Valid values are:

- Artifact
- HTTP POST

Default SSO Identity Provider

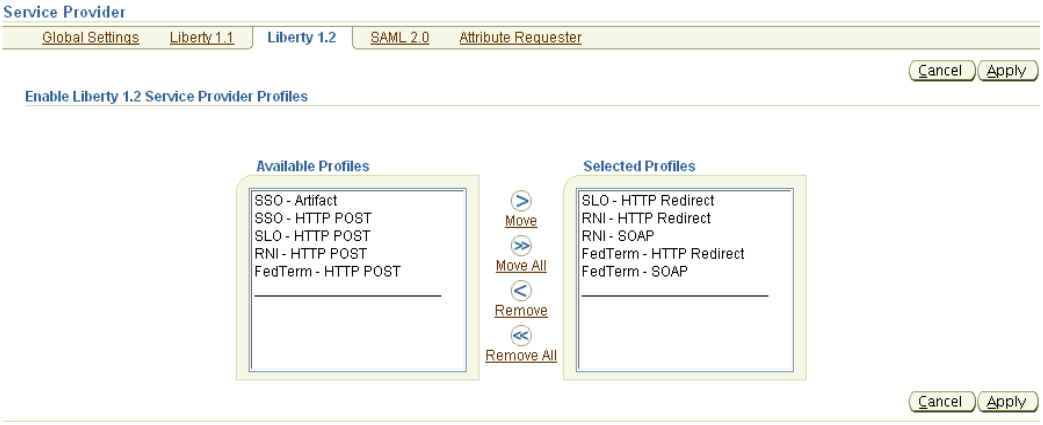
This is the default IdP to use in performing the single sign-on operation. Select a provider from the list of available IdPs.

The buttons at the bottom of the page perform the following functions:

- **Save** - saves changes made to Service Provider configuration properties.
- **Reset** - restores the screen to its original values without saving any changes.

Enable Liberty 1.2 Service Provider Profiles

Use this page to maintain Oracle Identity Federation profiles and bindings for the Liberty 1.2 protocol.



Available Profiles

Profiles and bindings in this list are disabled for the federation server instance.

Selected Profiles

Profiles and bindings in this list are enabled for the federation server instance.

Actions

Buttons on the page provide these actions:

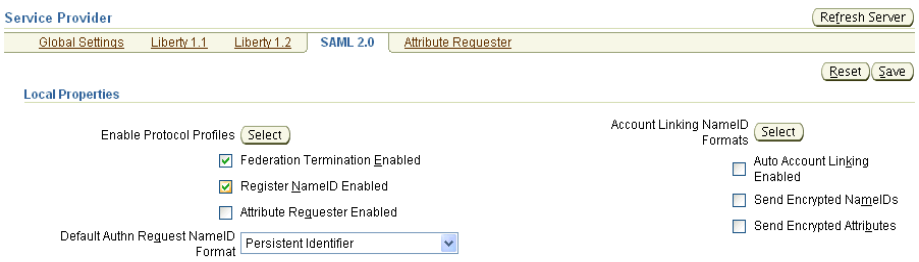
- **Move** - Moves the selected profiles/bindings to the **Selected Profiles** list. Use the control key to select multiple profiles/bindings.
- **Move All** - Moves all profiles/bindings currently in the **Available Profiles** list to the **Selected Profiles** list.
- **Remove** - Moves the selected profiles/bindings from the **Selected Profiles** list to the **Available Profiles** list. Use the control key to select multiple profiles/bindings.
- **Remove All** - Moves all profiles/bindings currently in the **Selected Profiles** list to the **Available Profiles** list.

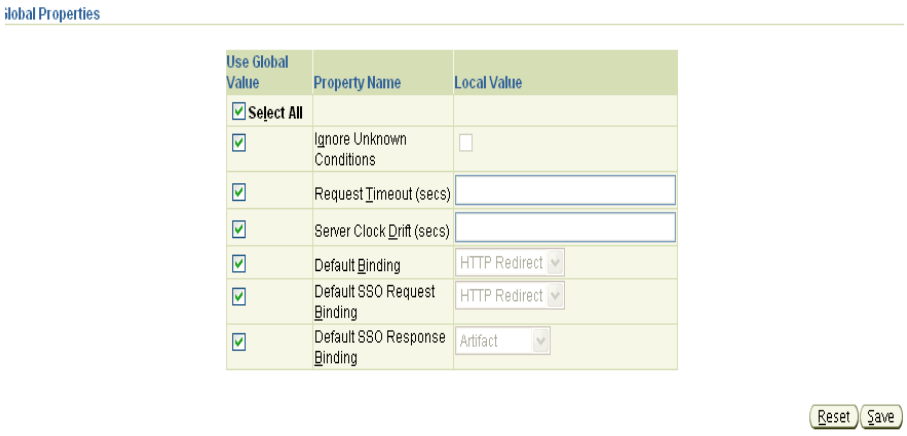
The buttons at the bottom of the page perform the following functions:

- **Apply** - saves changes made to Service Provider profiles/bindings.
- **Cancel** - returns you to Service Provider Liberty 1.2 Properties without saving any changes.

Service Provider - SAML 2.0 Properties

Use this page to maintain Oracle Identity Federation properties in service provider mode under the SAML 2.0 protocol.





The page provides these options:

Enable Protocol Profiles

Click **Select** to choose the protocol profiles and transport bindings that you wish to enable for this federation server instance in service provider mode for SAML 2.0.

> **Note:** This property affects server metadata. When updating this property, distribute the updated metadata to all providers in your circle of trust.

> **See Also:** "Enable SAML 2.0 Service Provider Profiles" on page 6-45

Federation Termination Enabled

Check this box to enable the federation termination functionality.

See "Federation Termination Profile" on page 1-17 for an explanation of this feature.

> **Note:** This property affects server metadata. When updating this property, distribute the updated metadata to all providers in your circle of trust.

Register NameID Enabled

Check this box to enable the name ID registration functionality.

See "Name Identifier Profiles" on page 1-16 for an explanation of this feature.

> **Note:** This property affects server metadata. When updating this property, distribute the updated metadata to all providers in your circle of trust.

Attribute Requester Enabled

Check this box to indicate that the SAML attribute sharing profile is enabled.

Default Authn Request NameID Format

Use the list box to select a default name ID format for authentication requests. Choices are:

- X.509 Subject Name
- Email Address
- Windows Domain Qualified Name
- Kerberos Principal Name
- Persistent identifier
- Transient/one-time identifier
- Unspecified

Account Linking NameID Formats

Click **Select** to maintain auto account linking name identifier formats for the SAML 2.0 SSO protocol.

> **See Also:** "Select SAML 2.0 Service Provider NameID Formats" on page 6-46

Auto Account Linking Enabled

When the service provider receives an authentication assertion from a peer identity provider, and the federation referenced in the assertion does not exist, the SP needs to locally authenticate the user to create the federation and perform the account linking operation.

If you check this box to enable the auto account linking feature, the SP will use the non-opaque name ID (e-mail, X.509, and so on) contained in the assertion to locate the user and automatically authenticate and perform the account linking steps.

Notes:

- Only the name identifier formats configured under Account Linking NameID Formats will be used for auto account linking.
- If the lookup operation returns two or more users, the SP will prompt the user for authentication.

Send Encrypted NameIDs

Check this box to enable Oracle Identity Federation to send encrypted name identifiers to peer providers.

Send Encrypted Attributes

Check this box to enable Oracle Identity Federation to send encrypted attributes to peer providers.

Use Global Value - Select All

Several service provider properties applicable to SAML 2.0 can be specified on the page:

- Ignore Unknown Conditions
- Artifact Timeout
- Request Timeout
- Server Clock Drift

- Default Binding
- Default SSO Request Binding
- Default SSO Response Binding
- Default SSO Identity Provider

Use **Select All** in this way:

- Check the box to specify that global service provider property settings override the local settings of all the listed properties.
- Uncheck the box to ensure that the local settings of all the properties override global settings.

To specify that the local setting of a given property overrides the global setting, uncheck the box for that property.

Ignore Unknown Conditions

A condition is an extension point in the XML schema. Custom conditions can be defined according to specific needs - for example, to denote that assertion X is conditional in a given time zone. Such conditions may not be amenable to evaluation, and checking this box allows the server to ignore a condition it does not recognize.

Request Timeout

This is the validity time, in seconds, of an outgoing request from the Oracle Identity Federation.

Server Clock Drift

This is the allowable time difference, in seconds, between Oracle Identity Federation, when acting as service provider, and its peer servers.

Default Binding

Specifies the preferred binding to use, when possible, in sending messages to peer providers. Valid values are:

- HTTP Redirect
- HTTP POST
- SOAP

Default SSO Request Binding

Specifies the preferred binding for the service provider to use, when possible, in sending authentication requests to the identity provider. Valid values are:

- HTTP Redirect
- HTTP POST

Default SSO Response Binding

Specifies the preferred binding for the identity provider to use, when possible, in sending an unsolicited assertion to the service provider. Valid values are:

- Artifact
- HTTP POST

Default SSO Identity Provider

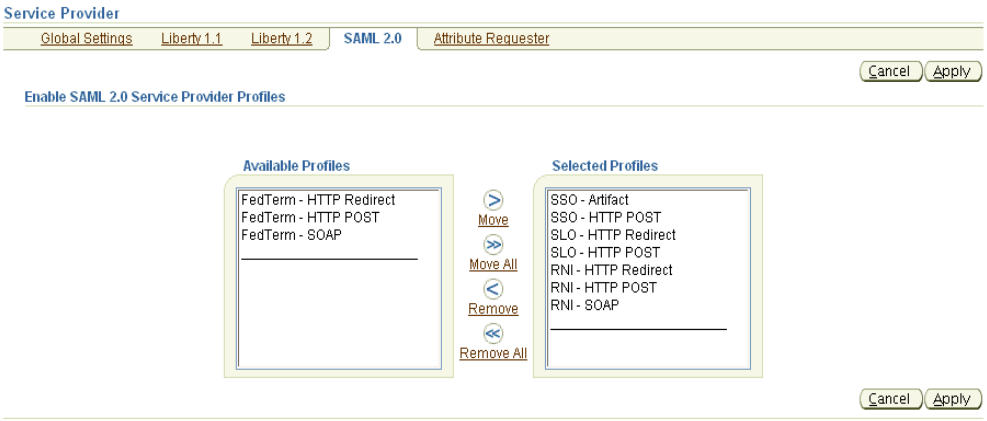
This is the default identity provider to use in performing the single sign-on operation. Select a provider ID from among the choices presented in the list box.

The buttons at the bottom of the page perform the following functions:

- **Save** - saves changes made to SAML 2.0 Service Provider profiles/bindings.
- **Reset** - restores the screen to its original values without saving any changes.

Enable SAML 2.0 Service Provider Profiles

Use this page to maintain SP profiles and bindings for the SAML 2.0 protocol.



Available Profiles

Profiles and bindings in this list are disabled for the federation server instance in SP mode.

Selected Profiles

Profiles and bindings in this list are enabled for the federation server instance in SP mode.

Actions

Buttons on the page provide these actions:

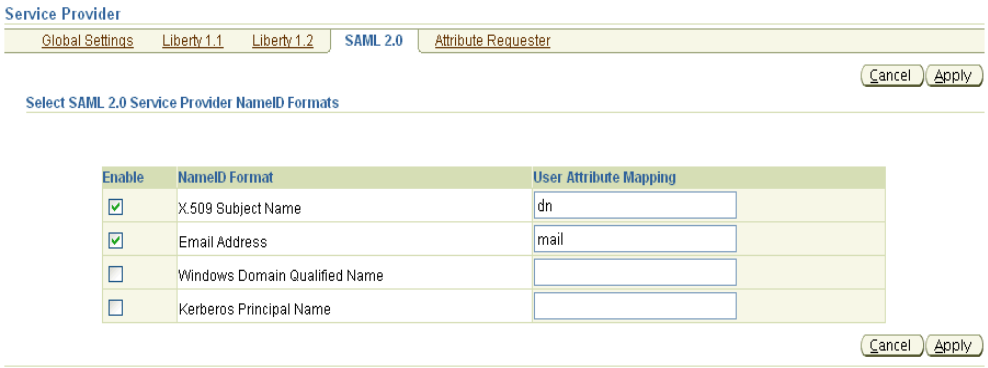
- **Move** - Moves the selected profiles/bindings to the **Selected Profiles** list. Use the control key to select multiple profiles/bindings.
- **Move All** - Moves all profiles/bindings currently in the **Available Profiles** list to the **Selected Profiles** list.
- **Remove** - Moves the selected profiles/bindings from the **Selected Profiles** list to the **Available Profiles** list. Use the control key to select multiple profiles/bindings.
- **Remove All** - Moves all profiles/bindings currently in the **Selected Profiles** list to the **Available Profiles** list.

The buttons at the bottom of the page perform the following functions:

- **Apply** - saves changes made to SAML 2.0 Service Provider profiles/bindings.
- **Cancel** - returns you to the Service Provider SAML 2.0 Properties page without saving any changes.

Select SAML 2.0 Service Provider NameID Formats

Use this page to maintain SAML 2.0 protocol name identifier formats for Oracle Identity Federation in SP mode.



Check the corresponding **Enable** box to enable the desired format(s) that the Oracle Identity Federation instance will use as the SAML 2.0 name identifier value in IdP mode.

NameID Format

This column displays the available SAML 2.0 NameID formats.

User Attribute Mapping

Enter the attribute name for the selected name ID format. Oracle Identity Federation will use this attribute name to perform a lookup in the user data store for a name ID in this format.

The name identifier formats are as follows:

Table 6–4 SAML 2.0 SP Name ID Formats

NameID Format	Mapping
X.509 Subject Name	DN
Email Address	e-mail
Windows Domain Qualified Name	empty
Kerberos Principal Name	empty

The buttons at the bottom of the page perform the following functions:

- **Save** - saves changes made to NameID format selections
- **Reset** - returns you to Service Provider SAML 2.0 Properties without saving any changes

Service Provider - Attribute Requester

Use this page to configure the mapping of DNs or sub-DNs to IdPs. This configuration is used with the Attribute Sharing Profile.

When an application sends an attribute request to this Oracle Identity Federation server with an X.509 SubjectDN, the server uses these mappings to determine to which IdP to send a SAML 2.0 AttributeQuery message on behalf of the application. Oracle Identity Federation will match the SubjectDN with the DNs configured here, from the most general to the most specific DNs.

Consider an example with two peer companies: PeerA has one identity provider and PeerB has three identity providers, for divisions Div1, Div2, and all other divisions. The attribute requester mappings might look like this:

Table 6–5 Example DN-to-IdP Mappings

DN	IdP
O=PeerA,C=US	http://fed.peera.com/fed/idp
OU=Div1,O=PeerB,C=US	http://fed.div1.peerb.com/fed/idp
OU=Div2,O=PeerB,C=US	http://fed.div2.peerb.com/fed/idp
O=PeerB,C=US	http://fed.other.peerb.com/fed/idp

Table 6–6 Example SubjectDN-to-IdP Mappings

SubjectDN	Maps to IdP
CN=John Doe,OU=DeptA,O=PeerA,C=US	http://fed.peera.com/fed/idp
CN=Jane Smith,OU=Div1,O=PeerB,C=US	http://fed.div1.peerb.com/fed/idp
CN=Bill Jones,OU=DeptY,OU=Div2,O=PeerB,C=US	http://fed.div2.peerb.com/fed/idp
CN=Sam McCoy,OU=Div5,O=PeerB,C=US	http://fed.other.peerb.com/fed/idp

DN Pattern

This is the DN or sub-DN being mapped.

Identity Provider/Attribute Responder

This is the identity provider or attribute responder URL to which the DN is being mapped.

Remove

Check this box and use the **Update** button to remove an existing mapping.

Note on Handling Unknown RDN Components

If the DN pattern contains RDN component types that are unknown to Oracle Identity Federation, you will not be able to add the entry to the responder mappings list.

You can resolve this by defining RDN component types in the Oracle Identity Federation server configuration file so that the server will recognize them. Take the following steps to implement this:

1. Open the `$ORACLE_HOME/fed/conf/config.xml` file
2. Search for the entry `<Config name="dnidpmapping">`. Add the RDN component type as shown:

```
<propertyset name="x500rdns">
    <propertyvalue>Email</propertyvalue>
    <propertyvalue>emailAddress</propertyvalue>
    <propertyvalue>NEW_RDN_COMPONENT</propertyvalue>
</propertyset>
<property name="Email">1.2.840.113549.1.9.1</property>
<property name="emailAddress">1.2.840.113549.1.9.1</property>
<property name="NEW_RDN_COMPONENT">NEW_OID</property>
```

3. Save the changes and restart the server.

Notes:

- Do not remove the entries already present in the file.
- You cannot use the Oracle Identity Federation Administrative console while making changes to the file, and should restart the server as soon as you are done with the changes.

Actions

Buttons on the page provide these actions:

- **Add Mapping** - Adds a new row for DN-to-IdP mapping information.
- **Apply** - adds a new mapping, deletes or saves changes you make to attribute responder mappings.

  After adding the information for a new mapping, press **Update** to store the new entry.

  To delete an existing mapping, check the corresponding box in the **Remove** column and press **Update**.
- **Cancel** - returns to the main Server Configuration page without saving any changes.

Editing Circles of Trust

This section describes how to edit and maintain Circle of Trust properties in Oracle Identity Federation. It contains these sub-sections:

- Circle of Trust
- Editing a Trusted Provider
- Edit Trusted Provider: Attribute Mappings
- Select Messages to Send Signed
- Select Messages to Require Signed
- Edit Trusted Provider: Select NameID Formats

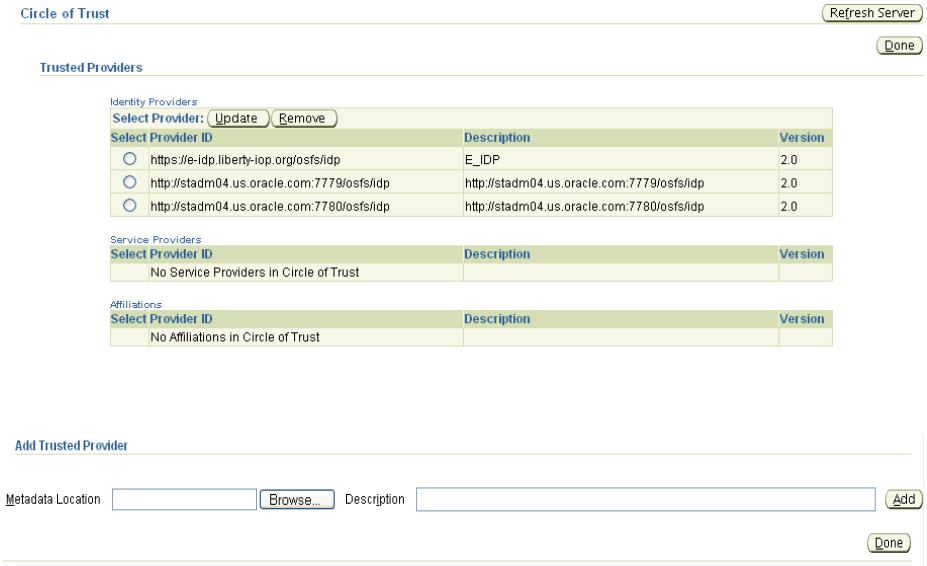
Note that configuration data is managed in the context of other providers in the circle of trust. Configuration details affect a provider's metadata, which must be re-published when certain data changes. There is a two-way exchange - a site administrator publishes metadata so that peer providers can load this data, and the administrator is also responsible for loading metadata published by peer providers in the circle of trust.

> **See Also:** For additional details, see "Server Configuration Data" on page 6-1

Circle of Trust

Use this page to add and update trusted providers in Oracle Identity Federation's circle of trust. The page displays three types of entities:

- Identity Providers
- Service Providers
- Affiliations, which are groupings of providers



Trusted Provider

Lists, in separate sets, the identity providers, service providers, and affiliations currently in the circle of trust. The fields in each set are:

- **Provider Id** - this is the provider ID
- **Description** - this is a brief description of the provider
- **Version** - this is the protocol version:
  - 1.1 (Liberty 1.1)
  - 1.2 (Liberty 1.2)
  - 2.0 (SAML 2.0)

Select a provider from the current trusted IdPs, SPs, and affiliations, and use these buttons to perform the following functions:

- **Update** - update provider or affiliation properties.
- **Remove** - remove the provider or affiliation from the circle of trust.

Note: If you remove a provider from the circle of trust which serves as the Default Single Sign-On (SSO) Identity Provider (defined on the "Service Provider - Global Settings" page), you must select a new default IdP on that page.

Add Trusted Provider

Allows additional peer providers to be included in the circle of trust. The fields are:

- **Metadata Location** - this is the location of the file containing the peer provider's metadata. Click **Browse** to choose a location.
- **Description** - this is a brief description of the provider.

Use the **Add** button to add the provider to the circle of trust.

Note: When you add a new provider to the circle of trust, it is automatically enabled and federation operations can be performed with the provider.

Metadata Signing Support Oracle Identity Federation 10.1.4.2 supports XML Digital Signatures in the XML Metadata documents that describe the services published by a compliant Federation Server. Oracle Identity Federation provides the following support for metadata signatures:

- digitally signing the metadata Oracle Identity Federation publishes
- verifying any XML digital signature present on a metadata document that is being uploaded to the server. If the verification fails, the metadata will not be uploaded.
- configuring the server to require an XML Digital Signature on the metadata in order to upload it into the circle of trust.

Take these steps to configure Oracle Identity Federation to sign the metadata it publishes:

1. Open the `$ORACLE_HOME/fed/conf/config.xml` file.
2. Locate the `FederationConfig` XML Element, and set its `useLocalConfig` attribute to `true`:

   ```
   <FederationConfig useLocalConfig="true">
   ```

3. Locate the XML `Config` element named `serverconfig`, and look for the `metadatasign` property:

   ```
   <Config name="serverconfig">
   ...
      <property name="metadatasign">false</property>
   ...
   </Config>
   ```

   Change the value of the property to `true` to configure Oracle Identity Federation to sign the XML Metadata documents, or to `false` to not sign the metadata.
4. Save the file and exit.
5. Restart the OC4J_FED instance.

Take these steps to configure Oracle Identity Federation to require signed metadata when importing a descriptor to the circle of trust:

1. Open the `$ORACLE_HOME/fed/conf/config.xml` file.
2. Locate the `FederationConfig` XML element, and set its `useLocalConfig` attribute to `true`:

   ```
   <FederationConfig useLocalConfig="true">
   ```

3. Locate the XML `Config` element named `serverconfig`, and look for the `metadatarequiresigned` property:

   ```
   <Config name="serverconfig">
   ...
      <property name="metadatarequiresigned">false</property>
   ...
   </Config>
   ```

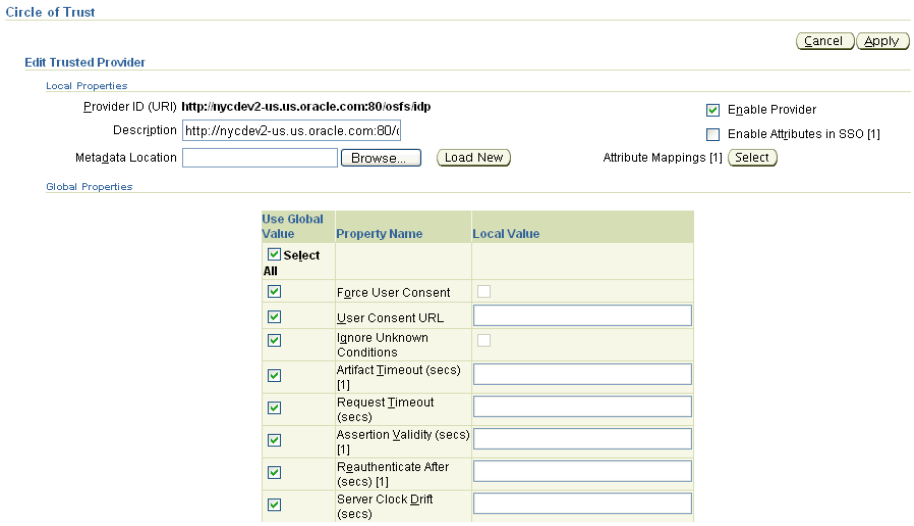
   Change the value of the property to `true` to require signed XML Metadata documents. Change the value to `false` to accept signed and unsigned metadata.

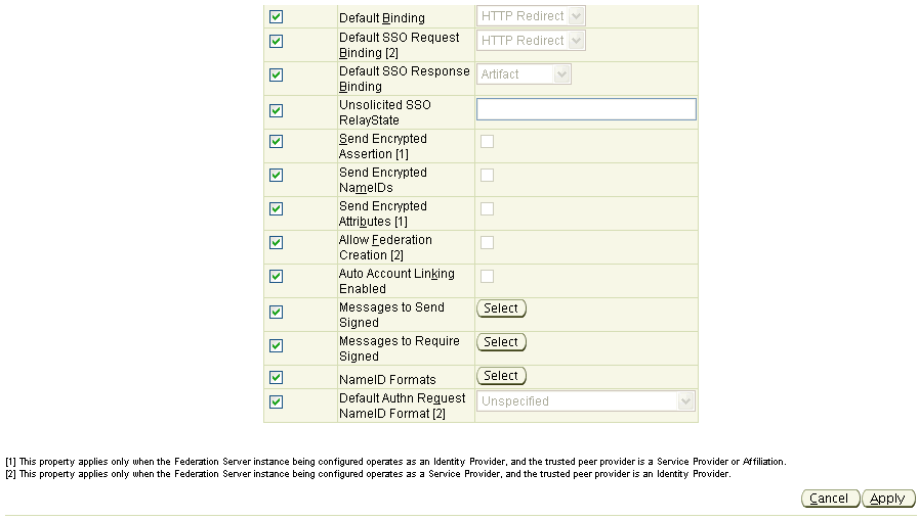
4. Save the file and exit.
5. Restart the OC4J_FED instance.

Editing a Trusted Provider

Clicking **Update** for a selected provider on the Circle of Trust page displays this page, on which you can edit properties for a trusted provider in the circle of trust.

Local properties (which are defined here) and global properties (inherited from global or protocol settings) are displayed separately.





The local properties are:

Provider ID

This is the provider ID in URI format.

Description

This is a description of the provider.

Metadata Location

Enter the file path to the metadata.

If the metadata has changed, you can use the **Load New** button to upload the new metadata.

Enable Provider

Check this box to enable this provider in the circle of trust.

Note: The next two fields, **Enable Attributes in SSO** and **Attribute Mappings**, apply only if your server instance operates as an identity provider, and the trusted provider (being configured here) is a service provider.

Enable Attributes in SSO

Check this box to specify that Oracle Identity Federation should send attributes with the authentication assertion when acting as an IdP.

Attribute Mappings

If you specified that the server should send attributes with the authentication assertion, click **Select** to define the attributes.

If the IdP acts as an Attribute Authority accepting attribute query requests and sending back attribute assertions, you will need to configure the attribute mappings.

See Also: Edit Trusted Provider: Attribute Mappings

Subsequent fields relate to global properties.

Use Global Value - Select All

Several global properties can be defined on this page.

Use **Select All** in this way:

- Check the box to specify that global trusted provider settings override the local settings of all the listed properties.
- Uncheck the box to ensure that the local settings of all the properties override global settings.

Force User Consent

Check this box to force consent for setting up a new federation. A user who is redirected to the federation server will explicitly have to accept or deny account linking in order to proceed.

User Consent URL

Enter the URL to be displayed to the user to obtain consent for federation.

The server passes a number of query parameters to this URL:

Table 6–7 Parameters Passed to User Consent URL (Local Setting)

Parameter	Description
providerid	This is the peer provider id.
description	This is the description of the peer provider id.
returnurl	This is the URL to which the user should be directed once a consent decision has been made.
refid	This is passed as a query parameter to the returnurl. Oracle Identity Federation require this parameter in order to resume the operation the server had been performing prior to redirection to the consent URL.

Use this field only if **Force User Consent** is checked.

Here is an example of a consent page:

```
<%
    String prefix = request.getContextPath();
    String redirectURL = request.getParameter("returnurl");
    String refID = request.getParameter("refid");
    String providerID = request.getParameter("providerid");
    String desc = request.getParameter("description");
%>
<HTML>
<BODY>
Do you consent to create a federation with <%=providerID%> (<%=desc%>):<br>
<form method="POST" action="<%=redirectURL%>">
    <input type="checkbox" name="userconsent" value="true"/>I agree<br>
    <input type="submit" value="OK" />
    <input type="hidden" name="refid"  value="<%=refID%>"/>
</form>
</BODY>
</HTML>
```

Ignore Unknown Conditions

A condition is an extension point in the XML schema. Custom conditions can be defined according to specific needs - for example, to denote that assertion X is conditional in a given time zone. Such conditions may not be amenable to evaluation, and checking this box allows the server to ignore a condition it does not recognize.

Artifact Timeout

This is the validity time, in seconds, of an artifact object created by Oracle Identity Federation. Use only if this server instance is acting as an identity provider.

Request Timeout

This is the validity time, in seconds, of an outgoing request from Oracle Identity Federation.

Assertion Validity

This is the time, in seconds, during which an assertion issued by an identity provider is valid. An assertion is considered invalid if processed outside the validity period. Use only if this server instance is acting as an identity provider.

Reauthenticate After

This is the time, in seconds, after which the service provider must re-authenticate the user. Assertions containing an authentication statement by the identity provider are only valid for this period, after which the user is to be considered non-authenticated.

Use only if this server instance is acting as an identity provider.

Note: This feature is applicable to OracleAS Single Sign-On authentication. For a limitation on forced reauthentication, see "Deploying Oracle Identity Federation with OracleAS Single Sign-On" on page 4-2.

Server Clock Drift

This is the allowable time difference, in seconds, between Oracle Identity Federation, when acting as identity provider, and its peer servers.

Default Binding

Specifies the preferred binding to use, when possible, in sending messages to peer providers. Valid values are:

- HTTP Redirect
- HTTP POST
- SOAP

Default SSO Request Binding

Specifies the preferred binding for the service provider to use, when possible, in sending authentication requests to the identity provider. Use only if this server instance is acting as a service provider. Valid values are:

- HTTP Redirect
- HTTP POST

Default SSO Response Binding

Specifies the preferred binding for the identity provider to use, when possible, in sending an unsolicited assertion to the service provider. Valid values are:

- Artifact
- HTTP POST

Unsolicited SSO RelayState

When an IdP performs unsolicited single sign-on operations, a relay state can be communicated to the service provider. Uncheck the box and enter the relay state.

This field is also used when the server instance is acting as a service provider. If the service provider does not receive a relay state in an authentication response from the identity provider, the service provider will use the relay state you provide here.

The Unsolicited SSO RelayState can be used as a final target URL to which the user is sent after the Single Sign-On operation successfully ends.

When an Oracle Identity Federation SP receives an unsolicited assertion, it sends the user to the relay state specified by the assertion following the SSO operation; if the relay state field in the assertion is empty, it will use the Unsolicited SSO RelayState to redirect the user.

Send Encrypted Assertion

Check this box to enable Oracle Identity Federation to send encrypted assertions to peer providers. Use only if your server instance operates as an identity provider, and the trusted provider (being configured here) is a service provider.

Note: This field applies only to SAML 2.0.

Send Encrypted NameIDs

Check this box to enable Oracle Identity Federation to send encrypted name identifiers to peer providers.

Note: This field applies only to SAML 2.0.

Send Encrypted Attributes

Check this box to enable Oracle Identity Federation to send encrypted attributes to peer providers. Use only if your server instance operates as an identity provider, and the trusted provider (being configured here) is a service provider.

Note: This field applies only to SAML 2.0.

Allow Federation Creation

Use this field if your server instance operates as a service provider, and the trusted provider (being configured here) is an identity provider. Check this box to allow federation between the providers.

Do not check this box if you already have the federations you need, and you do not want new federations to be created between your SP/IdP and the peer provider.

Auto Account Linking Enabled

Check this box to enable automatic account linking.

See Also: "Service Provider - SAML 2.0 Properties" on page 6-41

Messages to Send Signed

Click **Select** to display a list of federation protocol message types and specify the messages that Oracle Identity Federation sends, in IdP mode, that it can sign.

See Also: "Select Messages to Send Signed" on page 6-57

Messages to Require Signed

Click **Select** to display a page which lists the federation protocol message types and specify the messages that Oracle Identity Federation receives, in IdP mode, that it requires to be signed.

See Also: "Select Messages to Require Signed" on page 6-58

NameID Formats

Click **Select** to choose the NameID format to use when none is specified. Choices are:

- X.509 Subject Name
- Email Address
- Windows Domain Qualified Name
- Kerberos Principal Name
- Persistent Identifier [1]
- Transient/One-Time Identifier [1]

See Also: "Edit Trusted Provider: Select NameID Formats" on page 6-59

Default Authn Request NameID Format

Use the list box to select a default name ID format for authentication requests. Choices are:

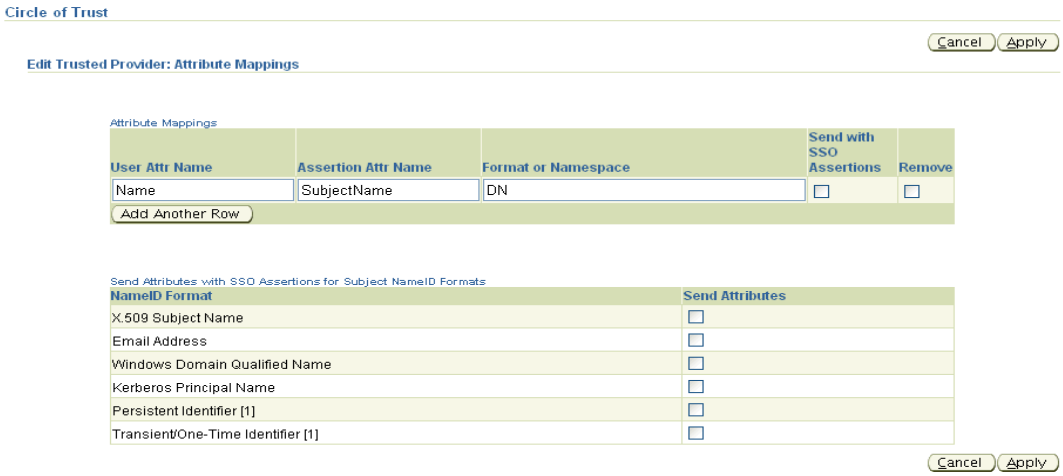
- X.509 Subject Name
- Email Address
- Windows Domain Qualified Name
- Kerberos Principal Name
- Persistent identifier
- Transient/one-time identifier
- Unspecified

The buttons at the bottom of the page perform the following functions:

- **Apply** - saves changes made to Trusted Provider properties.
- **Cancel** - restores the screen to its original values without saving any changes.

Edit Trusted Provider: Attribute Mappings

If enabling attributes for a trusted identity provider, use this page to maintain SSO attribute mappings and NameID formats.



Attribute Mappings

Each attribute consists of these properties:

- The user attribute name in the data store from which the value will be retrieved.
- The SAML attribute name that will be included in the assertion.
- The attribute format, used solely in the Liberty 1.1, Liberty 1.2, and SAML 2.0 protocols to specify the Attribute Namespace, which consists of any URI.

Send with SSO Assertions

Check this box to send the attribute in an authentication assertion.

Send Attributes with SSO Assertions for Subject NameID Formats

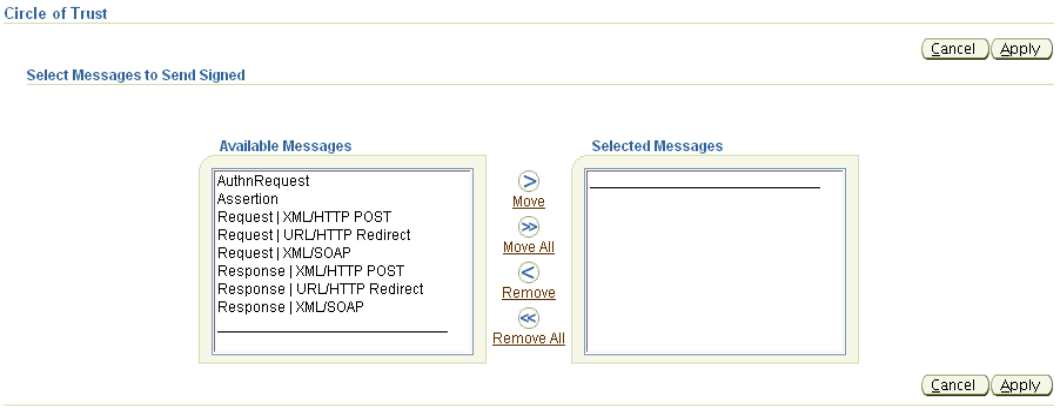
Click the desired check boxes to specify which attributes to include with SSO assertions.

The buttons at the bottom of the page perform the following functions:

- **Update** - saves changes made to attribute mappings.
- **Cancel** - returns you to Edit Trusted Provider without saving any changes.

Select Messages to Send Signed

Use this page when configuring a trusted peer provider, to specify which message types that provider should send signed.



Use this page when configuring a trusted peer provider, to specify which message types that provider should require signed.

Available Messages

Messages in this list are not required to be signed.

Selected Messages

Messages in this list are required to be signed.

Actions

Buttons on the page provide these actions:

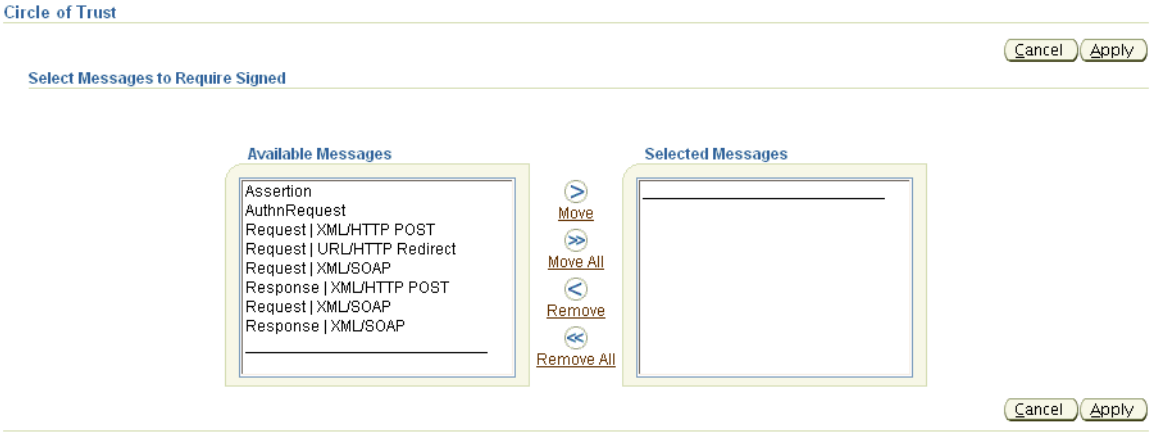
- **Move** - Moves the selected message types to the **Selected Messages** list. Use the control key to select multiple message types.
- **Move All** - Moves all messages currently in the **Available Messages** list to the **Selected Messages** list.
- **Remove** - Moves the selected message types from the **Selected Messages** list to the **Available Messages** list. Use the control key to select multiple message types.
- **Remove All** - Moves all messages currently in the **Selected Messages** list to the **Available Messages** list.

The buttons at the bottom of the page perform the following functions:

- **Apply** - saves changes made to the page for signed/unsigned messages.
- **Cancel** - returns to the Circle of Trust - Edit Trusted Provider page without saving any changes.

Select Messages to Require Signed

Use this page when configuring a trusted peer provider, to specify which message types that provider should require signed.



Available Messages

Messages in this list are not required to be signed.

Selected Messages

Messages in this list are required to be signed.

Actions

Buttons on the page provide these actions:

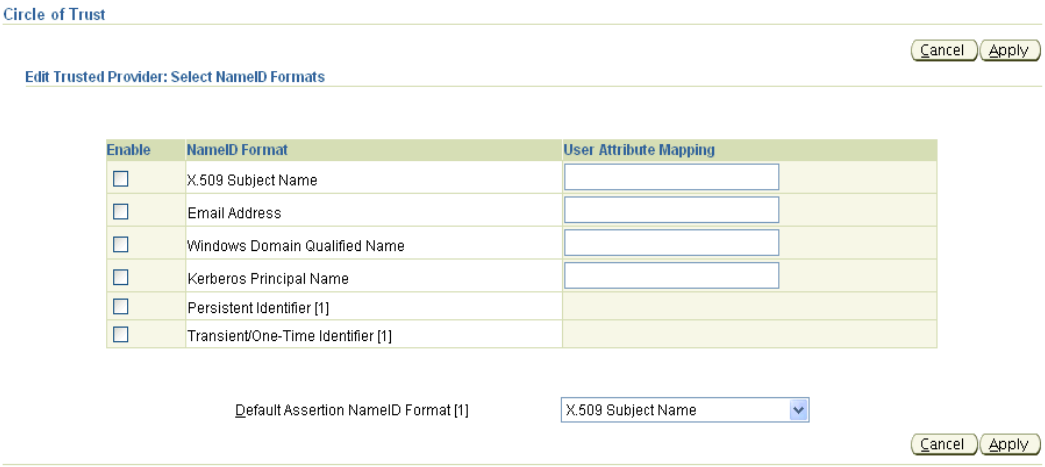
- **Move** - Moves the selected message types to the **Selected Messages** list. Use the control key to select multiple message types.
- **Move All** - Moves all messages currently in the **Available Messages** list to the **Selected Messages** list.
- **Remove** - Moves the selected message types from the **Selected Messages** list to the **Available Messages** list. Use the control key to select multiple message types.
- **Remove All** - Moves all messages currently in the **Selected Messages** list to the **Available Messages** list.

The buttons at the bottom of the page perform the following functions:

- **Apply** - saves changes made to the page for signed/unsigned messages.
- **Cancel** - returns to the Circle of Trust - Edit Trusted Provider page without saving any changes.

Edit Trusted Provider: Select NameID Formats

Use this page when configuring a trusted peer provider, to specify the NameID formats that are to be enabled for that provider.



NameID Format

This column displays the available NameID formats.

User Attribute Mapping

Enter the attribute name for the selected name ID format. Oracle Identity Federation will use this attribute name to perform a lookup in the user data store for a name ID in this format.

The name identifier formats are as follows:

Table 6–8 Trusted Provider Name ID Formats

NameID Format	Mapping
X.509 Subject Name	DN
Email Address	e-mail
Windows Domain Qualified Name	empty
Kerberos Principal Name	empty
Persistent Identifier	empty
Transient/One-Time Identifier	empty

Default Assertion NameID Format

This is the assertion name identifier format to use for this trusted provider.

The buttons at the bottom of the page perform the following functions:

- **Apply** - saves changes made to NameID format selections.
- **Cancel** - returns you to Edit Trusted Provider without saving any changes.

Configuring and Using Affiliations

This section explains affiliations and how they are used in Oracle Identity Federation. It contains these topics:

- About Affiliations
- Affiliation Support in Oracle Identity Federation

About Affiliations

A Liberty 1.2/SAML 2.0 affiliation consists of service providers that are part of a logical group.

An affiliation is not a concrete entity or server, but a logical provider; thus, no server can act as an affiliation. Rather, service providers use an affiliation when performing protocol message exchanges - in this case, the affiliation will be viewed as a logical provider, but the sender/received of messages for this affiliation will be a concrete service provider. In this way, the service providers participating in the affiliation act as the affiliation, and will have access to all the federation information about that logical provider.

An affiliation is described by an Affiliation Descriptor (also known as metadata for affiliation). It can be viewed as an abstract service provider that interacts with peer identity providers for SSO operations, and with service providers for Web services operations.

For example, consider an affiliation `Aff1` consisting of two service providers `SP1` and `SP2`. These two service providers, an identity provider `IDP`, and the affiliation are contained in a common circle of trust. User `ORAUSER` has established a federation, `FED_IDP_AFF1`, between `IDP` and the affiliation `Aff1`, the logical provider.

Taking advantage of the affiliation features, `SP1` and `SP2` - acting through `Aff1` - can perform single sign-on operations with `IDP`, using the `FED_IDP_AFF1` federation, even if they do not have any direct federations with `IDP`. They have access to `Aff1`'s federation information, and `IDP` views these service providers as `Aff1`.

The benefits of affiliations lies in reduced number of federations, since the service providers share the same federations. At the same time, this forces the SPs to share their federation information stores, as well as their user data stores, since the federation records are linked to user records.

Affiliation Support in Oracle Identity Federation

Oracle Identity Federation provides the following support for affiliations:

- Oracle Identity Federation, as an IdP, can interact with a service provider which is using an affiliation.
- Oracle Identity Federation, as an SP, can be part of an affiliation, and can use it to interact with an IdP.

Oracle Identity Federation does not provide any facilities to create and manage affiliations. As such, there is no support for operations like:

- adding or removing members
- setting owner and contact information of the affiliation
- creating metadata for the affiliation

Configuring Affiliations

Adding an affiliation to Oracle Identity Federation's circle of trust requires the same steps as adding an IdP or SP metadata to the COT: go to the circle of trust page, and add the affiliation's metadata.

See Also: "Circle of Trust" on page 6-49

If Oracle Identity Federation is an IdP, in order to interact with service providers that are members of an affiliation, the circle of trust needs to include metadata for both the affiliation and the service provider.

Runtime Behavior of Affiliations

The run-time functioning of affiliations depend on whether the Oracle Identity Federation server is acting as an IdP or an SP.

Oracle Identity Federation Acting as IdP

When Oracle Identity Federation is an IdP, provided the affiliation/SP is present and enabled in the circle of trust, the Oracle Identity Federation server is ready to process any requests originating from service providers using the affiliation.

Oracle Identity Federation Acting as SP

As an SP, you can trigger a single sign-on operation with an IdP using an affiliation to which the SP belongs. To do so, just include a `federationid` query parameter in the URL protected by the IdM back-end, and set the parameter value to the affiliation ID.

For example with an OracleAS Single Sign-On back-end, assuming that a resource is protected by `mod_osso` and configured for Oracle Identity Federation authentication, requesting the URL of this resource with the `federationid` query parameter will instruct Oracle Identity Federation to use an affiliation when performing single sign-on with a peer IdP. Here is an example of such a URL:

```
http://protected_res_host:protected_res_
port/path?federationid=http://affiliationid
```

It is also possible to directly access the `http://oif_host:oif_port/fed/sp/initiatesso` URL with the same `federationid` query parameter. In this case, Oracle Identity Federation will trigger a single sign-on operation, and will use the Unsolicited SSO RelayState for the peer IdP as the URL to which the user is redirected after successful authentication.

Note: The Unsolicited SSO RelayState is set on the **Server Configuration** -> **Circle of Trust** -> **Edit Trusted Provider** page of the Oracle Identity Federation administration console.

How Affiliations are Displayed

The Oracle Identity Federation administration console displays, under the Identity Federation tab, federations between the Oracle Identity Federation server and any supported entity type for which metadata is loaded, including federations with affiliations. When listing federations, the server uses these rules to display federations between itself and remote providers or affiliations, depending on the nature of the affiliation:

In the **Identity Federation** - > **Trusted Providers** section, the server lists federations between:

- Oracle Identity Federation server as an IdP and remote SPs from the Service Provider portion of the table
- Oracle Identity Federation server as an IdP and affiliations from the Affiliation portion of the table
- Oracle Identity Federation server as an SP and remote IdPs from the Identity Provider portion of the table

However, it will not display federations between remote IdPs and affiliations when this Oracle Identity Federation server acts as an SP and is part of the affiliations.

In the Identity Federation -> Users section, for a specific user, the server lists federations between:

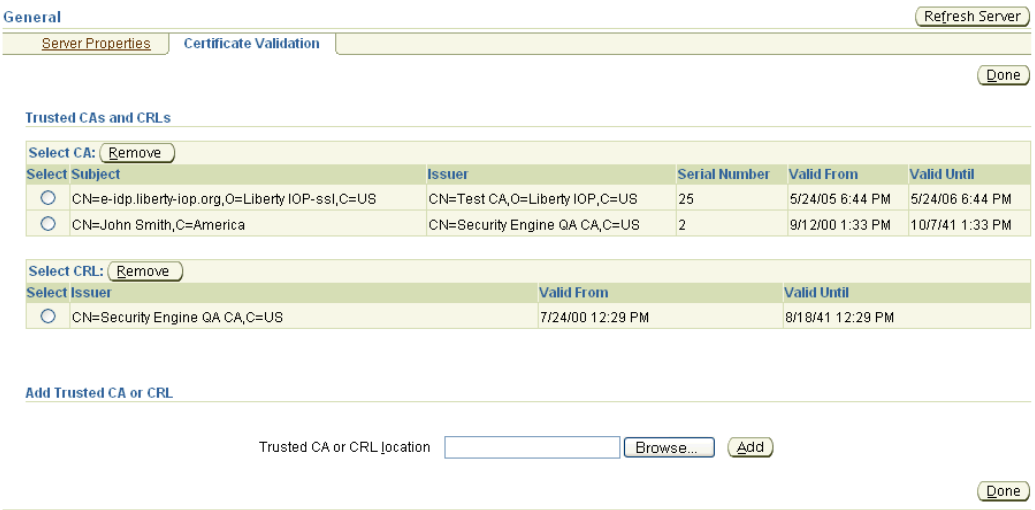
- Oracle Identity Federation server as an IdP and remote SPs
- Oracle Identity Federation server as an IdP and affiliations
- Oracle Identity Federation server as an SP and remote IdPs
- Affiliations of which this Oracle Identity Federation/SP server is a member, and remote IdPs

Editing the Certificate Validation Store

Use this page to maintain the certificate validation store, which enables you to manage:

- trusted certificate authorities (CAs)
- certificate revocation lists (CRLs)

> **Note:** Certificate validation applies only to the certificates used with the SAML 2.0 and Liberty 1.x protocols, not to SAML 1.x or WS-Federation.



CA Maintenance Fields

The CA table shows a list of CAs trusted by Oracle Identity Federation. Select a CA and click **Remove** to delete the CA from the list of trusted CAs.

The CA fields are:

- **Subject** - this is the CA certificate subject
- **Issuer** - this is the certificate issuer
- **Serial Number** - this is the certificate's serial number
- **Valid From** - this is the start time of the certificate validity period

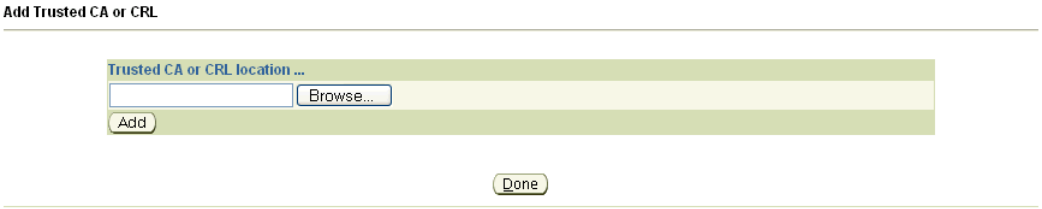
- **Valid Until** - this is the end time of the certificate validity period

CRL Maintenance Fields

The CRL table shows a list of Certificate Revocation Lists (CRLs) known to Oracle Identity Federation. Select a CRL and click **Remove** to delete the CRL from the list.

The CRL fields are:

- **Issuer** - this is the CA that issued the CRL
- **Valid From** - this is the start time of the CRL validity period
- **Valid Until** - this is the end time of the CRL validity period



Add Trusted CA or CRL

Use this field to add either a trusted CA or a CRL location. Use the **Browse** button to locate an X.509 Certificate or a CRL. X.509 certificates and CRLs can be loaded in DER or PEM format.

> **Note:** You must be sure that the CA certificates and CRLs are coming from a trusted source, since these files will be used to validate the signatures of the incoming messages.

Certificate Validation Repository

Oracle Identity Federation stores the entries for CAs and CRLs in the XML file `$ORACLE_HOME/fed/conf/cacert-store.xml`.

CA certificates are stored in Base64 encoded format.

Actions

Click the **Done** button at the bottom of the page to return to the main Server Configuration page.

Configuring IdM Data Stores

Use the IdM Data Stores tab to specify information about data stores. From this tab you can access the following pages:

- Edit Federation Data Store
- Edit User Data Store

Edit Federation Data Store

Use this page to maintain information about the repository which contains your federation data records. Start by choosing the active repository for the data:

- Select **LDAP Directory** if you wish to store the federation records in an LDAP server.

- Select **None (SAML 2.0 only)** if Oracle Identity Federation uses only non-opaque SAML 2.0 name identifiers.

Fields available on this page are dynamically determined by the type of repository you select:

- Federation Data in LDAP Directory
- No Active Repository (SAML 2.0)

> **Note:** Refer to "Federation Data Store" on page 2-17, in the discussion titled "A Note About LDAP Schema" for details on how to configure the LDAP schema.

Federation Data in LDAP Directory

When the Oracle Identity Federation federation records reside in an LDAP directory, you can specify the following:

Connection URL

This is the LDAP URL to connect to the server. For example,

```
ldap://ldap.oif.com:389
```

Bind DN

This is the administrator account DN to use to connect to the LDAP server. For example,

```
cn=orcladmin
```

Password

This is the administrator account password to use to connect to the LDAP server.

User Federation Record Context

This is the LDAP container entry under which all the federation records will be stored. For example,

```
cn=fed,dc=us,dc=oracle,dc=com
```

Notes:

The User Federation Record Context must be compatible with the **LDAP Container Object Class**, as explained in the description of that field.

If the User Federation Record Context container object does not exist in the LDAP directory, Oracle Identity Federation will create it at runtime the first time it needs to store a federation record.

LDAP Container Object Class

This is the type of the **User Federation Record Context** class that Oracle Identity Federation should use when creating the LDAP container, if one does not already exist. If this field is empty, its value will be set to `applicationProcess`.

Note: The chosen LDAP container object class must have `cn` as an attribute, as this is used for federation records created in the container.

For Microsoft Active Directory, this field has to be set (to `container` for example) depending on the **User Federation Record Context** since `applicationProcess` will not work under Microsoft Active Directory.

To see how these fields are related, note that the **User Federation Record Context** references the LDAP container entry under which federation records will be stored, and the **LDAP Container Object Class** defines the LDAP container attribute used in the DN. In the **User Federation Record Context**, you specify the DN of the container where the federation records will be stored. That DN contains the parent of an already existing container, and an attribute of the federation record context that is part of its object class. For example, if the container parent is `dc=us,dc=oracle,dc=com` and the record context attribute is `cn=orclfed`, the requirement that `cn` must be an attribute of the object class set in the **LDAP Container Object Class field** (or the `applicationProcess` object class if not set) ultimately produces a DN such as:

```
cn=orclfed,dc=us,dc=oracle,dc=com
```

If you choose to express the DN of the **Federation Record Context** as `ou=fed,dc=us,dc=oracle,dc=com`, you will need to set the **LDAP Container Object Class** to an object class that has `ou` as an attribute, like `applicationProcess`.

And if the DN is:

```
cn=fed,dc=us,dc=oracle,dc=com
```

then the **LDAP Container Object Class** must define the `cn` attribute.

Here are examples of the **LDAP Container Object Class** for different types of directory servers:

- Oracle Internet Directory: *empty*
- Sun Java System Directory Server: *empty*
- Microsoft Active Directory: *container*

Unique Federation ID Attribute

This is the LDAP attribute that Oracle Identity Federation uses to uniquely identify a federation record. If empty, its value will be DN, which is the DN of the LDAP federation record.

Here are examples of the **Unique Federation ID Attribute** for different types of directory servers:

- Oracle Internet Directory: *empty*
- Sun Java System Directory Server: *empty*
- Microsoft Active Directory: *empty*

Maximum Connections

This is the maximum number of LDAP connections that Oracle Identity Federation will simultaneously open to the LDAP server.

Connection Wait Timeout

This is the timeout, in minutes, to use when Oracle Identity Federation opens a connection to the LDAP server.

No Active Repository (SAML 2.0)

Select **None** as the Active Repository if:

1. you do not wish to use an LDAP server to store federation records, and
2. Oracle Identity Federation will be configured to use federation assertions with non-opaque name identifiers based on the SAML 2.0 profile (email, X.500, Kerberos, or WindowsNameQualifier)

   or

   Oracle Identity Federation will be configured to use only SAML 1.x or WS-Federation

No additional configuration is required for this option.

The buttons at the bottom of the page perform the following functions:

- **Save** - saves changes made to federation data store properties.
- **Reset** - restores the original values without saving any changes.

Edit User Data Store

Use this page to edit the configuration of the user data store. Oracle Identity Federation uses this information to retrieve user attributes from the data store.

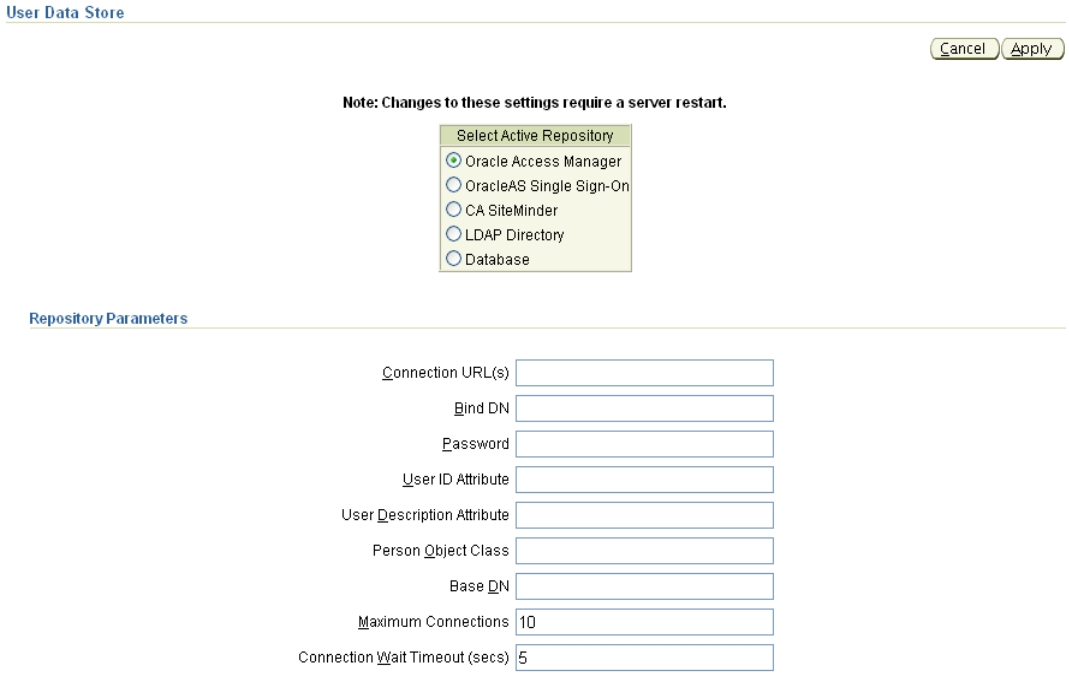
> **Note:** This information is used to retrieve user attributes, not for authentication.

Fields are displayed dynamically depending on the repository type you choose:

- User Data in Oracle Access Manager
- User Data in OracleAS Single Sign-On
- User Data in CA eTrust SiteMinder

- User Data in LDAP Directory
- User Data in a Database

User Data in Oracle Access Manager



Repository Parameters

Connection URL(s)

This is the LDAP URL to connect to Oracle Access Manager.

Bind DN

This is the administrator account DN to use to connect to Oracle Access Manager.

Password

This is the administrator account password to use to connect to Oracle Access Manager.

User ID Attribute

This is the LDAP attribute used to identify the user during authentication, for example `uid`.

Here are examples of the **User ID Attribute** for different types of directory servers:

- Oracle Internet Directory: `uid`
- Sun Java System Directory Server: `uid`
- Microsoft Active Directory: `sAMAccountName`

User Description Attribute

This is the human-readable LDAP attribute used to identify the owner of a federation record, for example `uid`.

Here are examples of the **User Description Attribute** for different types of directory servers:

- Oracle Internet Directory: `uid`
- Sun Java System Directory Server: `uid`
- Microsoft Active Directory: `sAMAccountName`

Person Object Class

Object classes define what data or attributes are associated with an object. A person object class refers to the attributes of a "person" object; in our context, it is the owner of a federated identity. A directory may utilize one or more object classes to hold person data (names, addresses, and so on).

Enter the LDAP object class representing an LDAP user entry in the server. For example:

`inetOrgPerson`

Here are examples of the **Person Object Class** for different types of directory servers:

- Oracle Internet Directory: `inetOrgPerson`
- Sun Java System Directory Server: `inetOrgPerson`
- Microsoft Active Directory: `user`

Base DN

This is the domain of the LDAP server containing the user records. For example:

`dc=us,dc=oracle,dc=com`

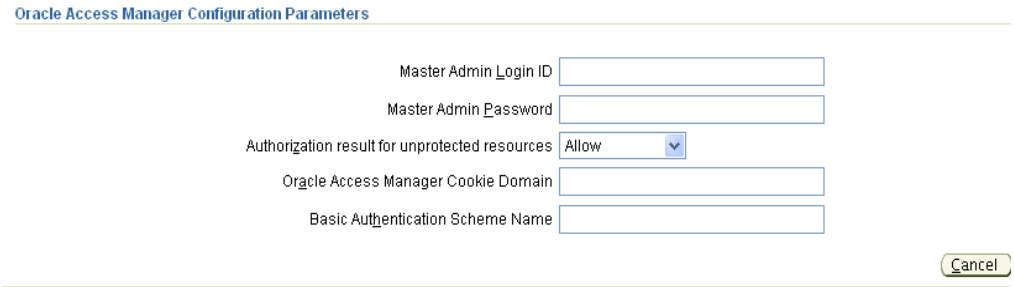
Maximum Connections

This is the maximum number of LDAP connections that Oracle Identity Federation will simultaneously open to Oracle Access Manager.

Connection Wait Timeout

This is the timeout, in minutes, to use when Oracle Identity Federation opens a connection to Oracle Access Manager.

Oracle Access Manager Configuration Parameters



Master Admin Login ID

This is the master administrator login. Oracle Identity Federation uses this login to create Oracle Access Manager policy objects related to federation.

Master Admin Password

This is the master administrator password.

Authorization result for unprotected resources

This is the result returned for a SAML 1.x `AuthorizationDecisionQuery` for a resource that is not protected by an Oracle Access Manager policy. Choices are:

- Allow
- Deny
- Indeterminate

> **Note:** If you choose **Deny**, SAML 1.x SSO access to unprotected resources will be blocked. This is equivalent to setting Oracle Access Manager AccessGate Deny On Not Protected to **On**.

Oracle Access Manager Cookie Domain

This is the domain for which the cookie is valid.

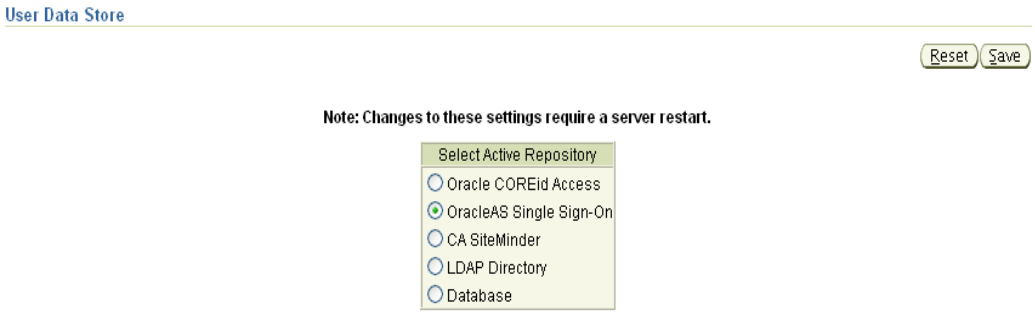
> **See Also:** "Integrate Oracle Identity Federation and Oracle Access Manager" on page 4-9 for a discussion of this field

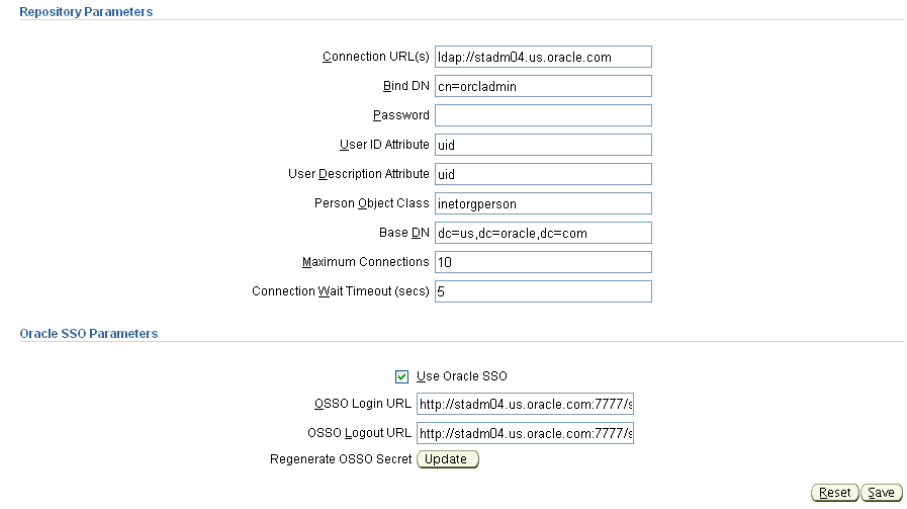
Basic Authentication Scheme Name

This is the name of an authentication scheme using the Basic challenge method, for example, Oracle Access and Identity Basic Over LDAP. This scheme is used in the Fed Domain policy domain.

> **See Also:** "Integrate Oracle Identity Federation and Oracle Access Manager" on page 4-9 for a discussion of this field

User Data in OracleAS Single Sign-On





Repository Parameters

Oracle Internet Directory is the user data repository for OracleAS Single Sign-On.

Connection URL(s)

This is the LDAP URL to connect to Oracle Internet Directory.

Bind DN

This is the administrator account DN to use to connect to Oracle Internet Directory.

Password

This is the administrator account password to use to connect to Oracle Internet Directory.

User ID Attribute

This is the LDAP attribute used to identify the user during authentication, for example `uid`.

User Description Attribute

This is the human-readable LDAP attribute used to identify the owner of a federation record. For example:

```
uid
```

Person Object Class

Object classes define what data or attributes are associated with an object. A person object class refers to the attributes of a "person" object; in our context, it is the owner of a federated identity. A directory may utilize one or more object classes to hold person data (names, addresses, and so on).

Enter the LDAP object class that represents an LDAP user entry. For example:

```
inetOrgPerson
```

Base DN

This is the directory to which the search for users should be confined. For example:

```
dc=us,dc=oracle,dc=com
```

Maximum Connections

This is the maximum number of LDAP connections that Oracle Identity Federation will simultaneously open to Oracle Internet Directory.

Connection Wait Timeout

This is the timeout, in minutes, to use when Oracle Identity Federation opens a connection to Oracle Internet Directory.

OracleAS Single Sign-On Parameters

OracleAS Single Sign-On parameters include:

- **Use Oracle SSO** - Check this box to use Oracle Single Sign-On as the authentication module.

  Note: The Oracle Application Server (on which Oracle Identity Federation server is installed) needs to be associated with the Oracle SSO Server. To do this, go to the OracleAS Enterprise Manager console, then to Infrastructure, and in the Identity Management section, click the **Configure** button. Refer to the *Oracle Application Server Administrator's Guide* for details.

- **OSSO Login URL** - This is the OracleAS Single Sign-On server URL to present at login. For example:

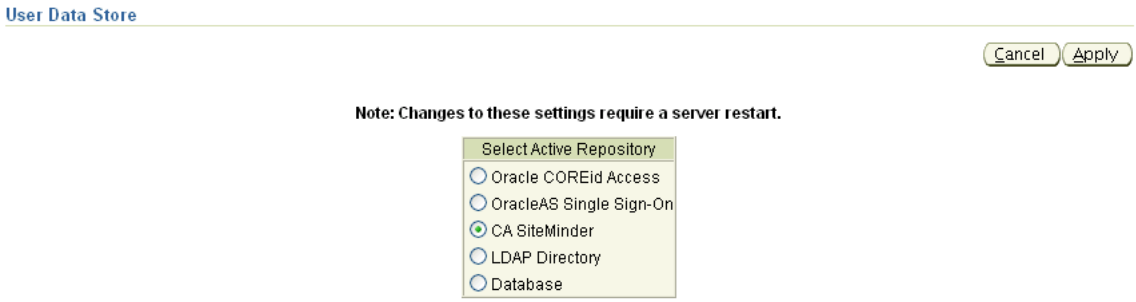
  ```
  http://sso_host:sso_port/sso/auth
  ```

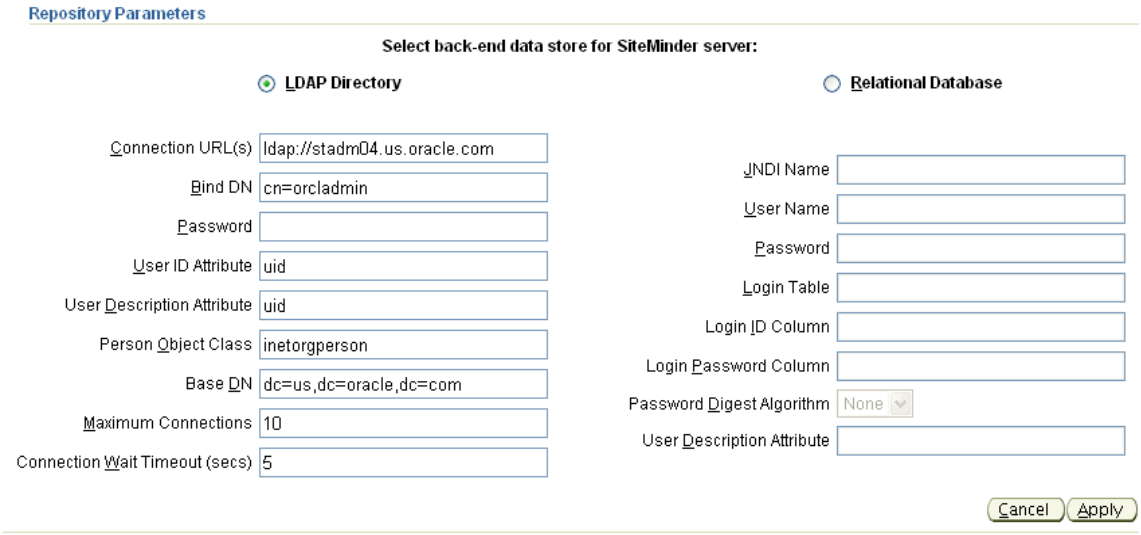
- **OSSO Logout URL** - This is the OracleAS Single Sign-On server URL to present at logout. For example:

  ```
  http://sso_host:sso_port/sso/logout
  ```

- **Regenerate OSSO Secret** - Click **Update** to regenerate the key.

User Data in CA eTrust SiteMinder





Either an LDAP directory or a relational database can serve as the back-end repository for eTrust SiteMinder.

> **Note:** When employing eTrust SiteMinder with an RDBMS back-end, you can use Oracle Identity Federation as a service provider similar to the way it is configured as an identity provider. The difference is that when you configure it as a service provider, the RDBMS to use should be the one configured for eTrust SiteMinder.

eTrust SiteMinder with LDAP Directory Data Store

Use these fields to update parameters for communication with a eTrust SiteMinder data store residing in an LDAP directory.

Connection URL(s)

This is the LDAP URL to connect to eTrust SiteMinder.

Bind DN

This is the administrator account DN to use to connect to eTrust SiteMinder.

Password

This is the administrator account password to use to connect to eTrust SiteMinder.

User ID Attribute

This is the LDAP attribute used to identify user during authentication, for example `uid`.

Here are examples of the **User ID Attribute** for different types of directory servers:

- Oracle Internet Directory: `uid`
- Sun Java System Directory Server: `uid`
- Microsoft Active Directory: `sAMAccountName`

User Description Attribute

This is the human-readable LDAP attribute used to identify the owner of a federation record, for example `uid`.

Here are examples of the **User Description Attribute** for different types of directory servers:

- Oracle Internet Directory: `uid`
- Sun Java System Directory Server: `uid`
- Microsoft Active Directory: `sAMAccountName`

Person Object Class

Object classes define what data or attributes are associated with an object. A person object class refers to the attributes of a "person" object; in our context, it is the owner of a federated identity. A directory may utilize one or more object classes to hold person data (names, addresses, and so on).

Enter the LDAP object class representing an LDAP user entry in the server.

Here are examples of the **Person Object Class** for different types of directory servers:

- Oracle Internet Directory: `inetOrgPerson`
- Sun Java System Directory Server: `inetOrgPerson`
- Microsoft Active Directory: `user`

Base DN

This is the domain of the LDAP server containing the user records. For example:

```
dc=us,dc=oracle,dc=com
```

Maximum Connections

This is the maximum number of LDAP connections that Oracle Identity Federation will simultaneously open to eTrust SiteMinder.

Connection Wait Timeout

This is the timeout, in minutes, to use when Oracle Identity Federation opens a connection to eTrust SiteMinder.

Buttons on this page provide the following domain maintenance features:

- **Update** - Presents a second page for eTrust SiteMinder server and agent configuration.
- **Reset** - Cancels the operation.

eTrust SiteMinder with Database Data Store

Use these fields to update parameters for communication with an eTrust SiteMinder data store residing in a relational directory.

JNDI Name

This is the EJB JNDI Name of the data source used to locate and authenticate users. For example:

```
jdbc/RDBMSUserDataSource
```

User Name

This is the administrator account username used to connect to the database server.

Password

This is the administrator account password to use to connect to the database server.

Login Table

This is the database table of the data source containing the user login information.

Login ID Column

This is the table column containing the user identifier needed to authenticate and locate users.

Login Password Column

This is the table column containing the user's passwords needed for authentication.

Password Digest Algorithm

This is the password digest algorithm to use during authentication to match the password data contained in the password column. Choices are:

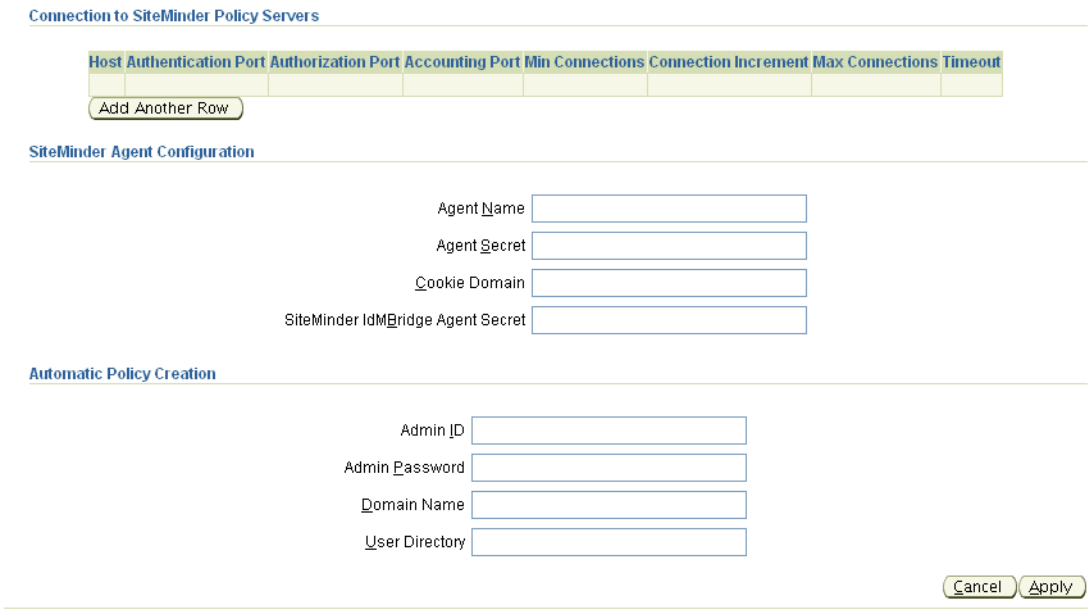
- MD5
- SHA
- None

User Description Attribute

This is the database table column containing a human-readable attribute used to identify the owner of a record.

eTrust SiteMinder Connection Data

After you supply eTrust SiteMinder repository parameters, a separate screen collects information used to connect to the policy server and for agent configuration.



Connection to eTrust SiteMinder Policy Servers

Update eTrust SiteMinder server connection details:

- Host - This is the host where the policy server is installed.
- Authorization Port - This is the policy server port used for authentication requests. The default value is 44442.
- Authentication Port - This is the policy server port used for authentication requests. The default value is 44443.
- Accounting Port - This is the policy server port used for accounting requests. The default value is 44441.
- Max Connections - This is the maximum number of agent connections to the policy server.
- Min Connections - This is the minimum number of agent connections to the policy server.
- Step Connections - This is the number of connections the agent can open to the policy server at one time.
- Timeout - This is the time, in seconds, that the agent will wait for a response from the policy server before it returns a failure.

eTrust SiteMinder Agent Configuration

eTrust SiteMinder agents control access to protected resources. Update eTrust SiteMinder agent configuration with these fields:

- Agent Name - This is the name of the eTrust SiteMinder agent.
- Agent Secret - This is the password for the pre-existing agent registered with the eTrust SiteMinder policy server.
- Cookie Domain - This is the cookie domain for the agent.
- SiteMinder IdMBridge Agent Secret - This is the secret string shared between the agent and the policy server.

Automatic Policy Creation

The eTrust SiteMinder bridge automatically creates the policy objects it requires by logging into eTrust SiteMinder as an administrator. Update these eTrust SiteMinder policy details:

- Admin ID - This is the eTrust SiteMinder administrator ID.
- Admin Password - This is the eTrust SiteMinder administrator password.
- Domain Name - This is the eTrust SiteMinder to contain the policy objects.
- User Directory - This is the eTrust SiteMinder-configured name of the user directory for the domain.

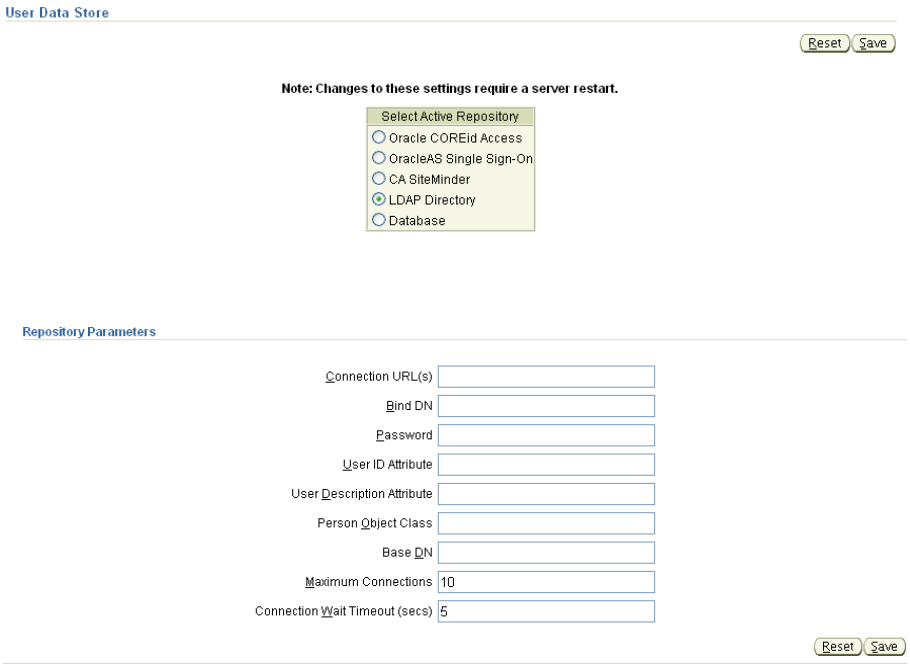
Assertion Mapping Using a Secondary IdMBridge

If an incoming assertion contains the name of the local eTrust SiteMinder user, it can be used directly. However, if the user name is not supplied in the assertion, a secondary bridge is necessary to map the assertion SubjectName and Attribute values to a user. Provide this information for eTrust SiteMinder assertion mapping:

- Select Secondary IdMBridge - Specify the type of bridge. Choices are:
  - LDAP Bridge

  - RDBMS Bridge
  - None
- User Name Attribute in Directory - Enter one of these:
  - For LDAP bridge, enter an attribute in the user's directory entry.
  - For RDBMS Bridge, enter a column in the user database table.
  - For None, specify an attribute from the received assertion.

User Data in LDAP Directory



Connection URL(s)

This is the LDAP URL to connect to the LDAP directory.

Bind DN

This is the administrator account DN to use to connect to the LDAP directory.

Password

This is the administrator account password to use to connect to the LDAP directory.

User ID Attribute

This is the LDAP attribute used to identify user during authentication, for example `uid`.

Here are examples of the **User ID Attribute** for different types of directory servers:

- Oracle Internet Directory: `uid`
- Sun Java System Directory Server: `uid`
- Microsoft Active Directory: `sAMAccountName`

User Description Attribute

This is the human-readable LDAP attribute used to identify the owner of a federation record, for example `uid`.

Here are examples of the **User Description Attribute** for different types of directory servers:

- Oracle Internet Directory: `uid`
- Sun Java System Directory Server: `uid`
- Microsoft Active Directory: `sAMAccountName`

Person Object Class

Object classes define what data or attributes are associated with an object. A person object class refers to the attributes of a "person" object; in our context, it is the owner of a federated identity. A directory may utilize one or more object classes to hold person data (names, addresses, and so on).

Enter the LDAP object class representing an LDAP user entry in the server.

Here are examples of the **Person Object Class** for different types of directory servers:

- Oracle Internet Directory: `inetOrgPerson`
- Sun Java System Directory Server: `inetOrgPerson`
- Microsoft Active Directory: `user`

Base DN

This is the directory to which the search for users should be confined.

Maximum Connections

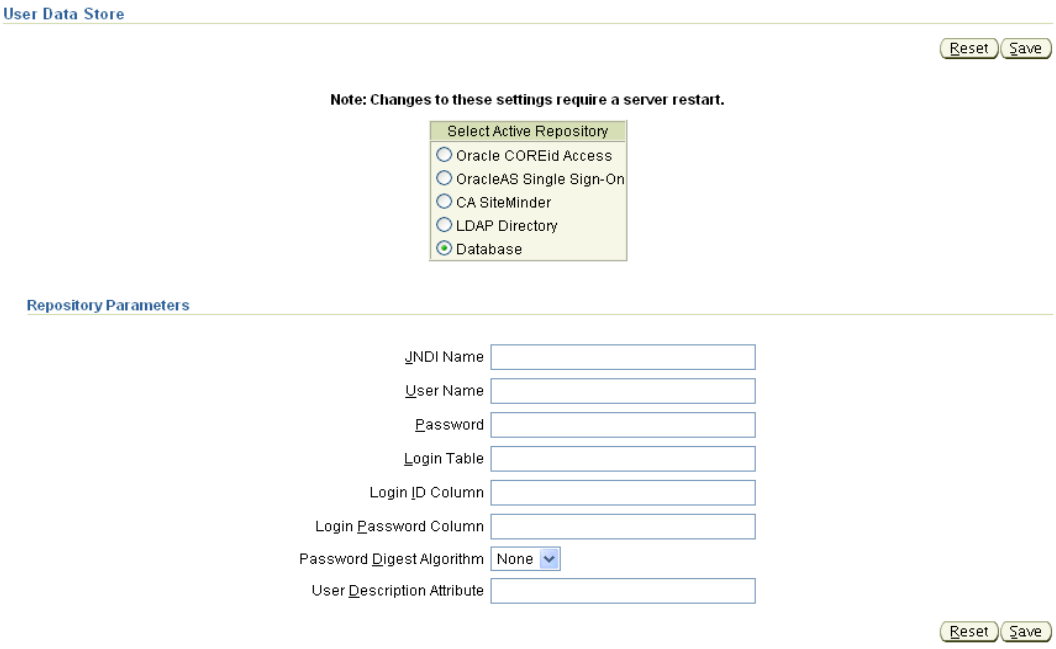
This is the maximum number of LDAP connections that Oracle Identity Federation will simultaneously open to the directory.

Connection Wait Timeout

This is the timeout, in minutes, to use when Oracle Identity Federation opens a connection to the directory.

User Data in a Database

> **See Also:** For details about the tasks required to configure a database, see "Configuring an RDBMS as the User Data Store" on page 6-80



JNDI Name

This is the EJB JNDI Name of the data source to use to locate and authenticate users. For example:

`jdbc/RDBMSUserDataSource`

User Name

This is the administrator account username to use to connect to the database server.

Password

This is the administrator account password to use to connect to the database server.

Login Table

The database table of this data source containing the user login information.

Login ID Column

This is the table column containing the user identifier needed to authenticate and locate users.

Login Password Column

This is the table column containing the user's passwords needed for authentication.

Password Digest Algorithm

This is the password digest algorithm to use during authentication to match the password data contained in the password column. Choices are:

- MD5
- SHA
- None

User Description Attribute

This is the database table column containing a human-readable attribute used to identify the owner of a federation record.

Configuring an RDBMS as the User Data Store

This section explains how to configure a relational database as the user data store for Oracle Identity Federation.

> **Note:** If a data source has been already created at the OC4J_FED instance level, skip steps 1 and 2.

1. Log in to the EM console of your Oracle Identity Federation instance and navigate to **OC4J_FED** - > **Administration** - > **Data Sources**.
2. Create a new data source using the following example as a guide:

```
Name:                myDS
Data Source Class:   com.evermind.sql.DriverManagerDataSource
JDBC URL:            jdbc:oracle:thin:@stahs08.us.oracle.com:1521:ORCL
JDBC Driver:         oracle.jdbc.driver.OracleDriver
Username:            CUSTDATA
Password:            PASSWORD
Location:            jdbc/RDBMSUserDataSource
Transactional Loc:   jdbc/xa/RDBMSUserDataSource
EJB Location:        jdbc/RDBMSUserDataSource
```

> **Note:** The database connection information must be updated to reflect your deployment configuration; refer to the Oracle Application Server and Oracle JDBC documentation for more information.

   Apply the changes.

3. Log into the Oracle Identity Federation administration console and navigate to **IdM Data Stores** - > **User Data Store**.
4. Select 'Database' as the Active Repository and enter the appropriate values, for example:

```
JNDI Name:             jdbc/RDBMSUserDataSource
User Name:             CUSTDATA
Password:              PASSWORD
Login Table:           EMPLOYEES
Login ID Column:       EMPLOYEE_ID
Login Pwd Column:      USERPASSWORD
User Desc. Attribute:  EMPLOYEE_ID
```

> **Note:** This is an example and is meant to be a guide. For example, you need to substitute the relevant user name for CUSTDATA and so on.

   Save the changes.

5. Restart the OC4J_FED instance.

Configuring SAML 1.x and WS-Federation Properties

Use the **SAML 1.x/WS-Fed** tab to specify configuration details for Oracle Identity Federation SAML 1.x domains.

From **SAML 1.x/WS-Fed** sub-tabs you can access the following pages:

- Certificate Store

  Enables you to configure digital signatures for secure communication with other providers.

- Regenerate Encryption Key

  The encryption key protects your sessions. Using this page, you can regenerate the key on a regular basis.

- Audits and Logs

  Enables you to set up collection of log data of system events, and audit data of assertions sent and received by your domain. Text descriptions of system events and assertions are written to an audit file.

- Assertion Profiles

  Enables you to configure your **domain** as a source domain, so that your users can access other domains in a federated transaction.

- Destination Mappings

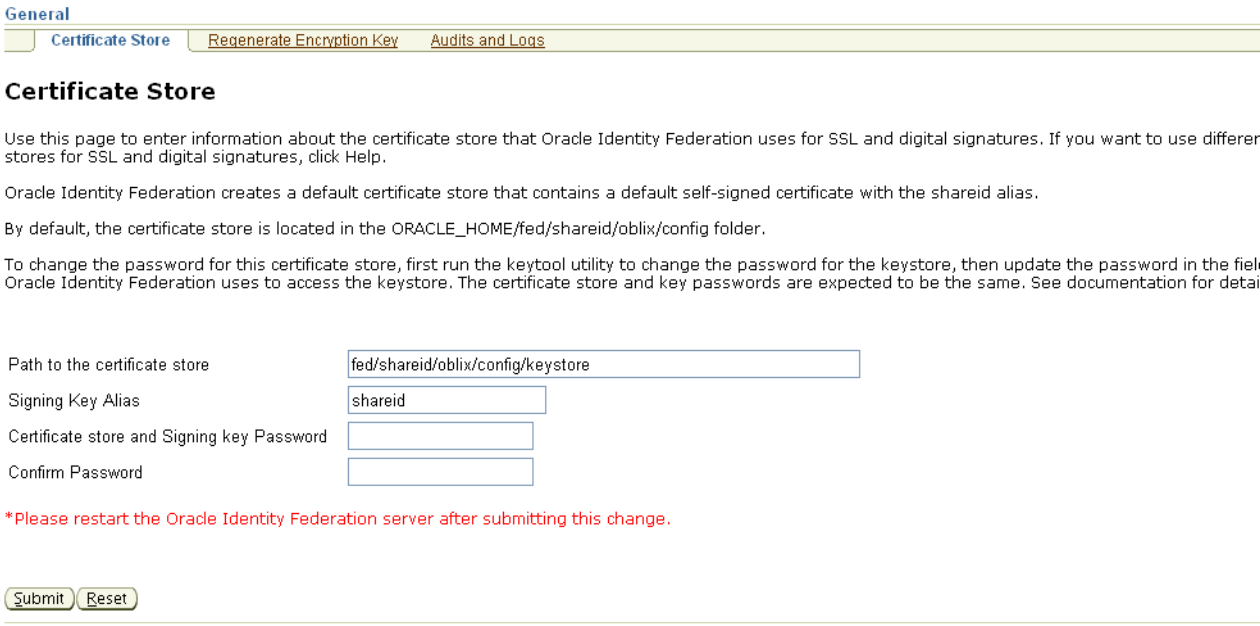
  Enables you to configure your **domain** as a destination domain, so that users from other domains can access your site in a federated transaction.

- Domains, which allows you to configure your **domain** as well as hosts, ports, and other properties for other domains

This section contains these topics:

Certificate Store

Use this page to enter information about the certificate store that Oracle Identity Federation uses for SSL and digital signatures. If you want to use different certificate stores for SSL and digital signatures, click **Help**.



Oracle Identity Federation creates a default key and certificate store, for use with the SAML 1.x and WS-Federation protocols, containing a default self-signed certificate with the `shareid` alias.

By default, the certificate store is located in the `/fed/shareid/oblix/config` folder of your Oracle Identity Federation installation directory.

The initial password for this certificate store is the same as the `ias_admin` password set during Oracle Identity Federation installation. To change the password for this certificate store, first run the `keytool` utility to change the password for the keystore; then update the password that Oracle Identity Federation uses to access the keystore. The certificate store and key passwords are expected to be the same.

> **See Also:** For an example of`keytool` utility usage, see "Working with Affiliations" on page 7-12

Path to the certificate store

This is the path to the certificate store.

Signing Key Alias

This is a shorthand name for the signing key.

Certificate Store and Signing Key Password

This is the password that Oracle Identity Federation uses to access the keystore.

Confirm Password

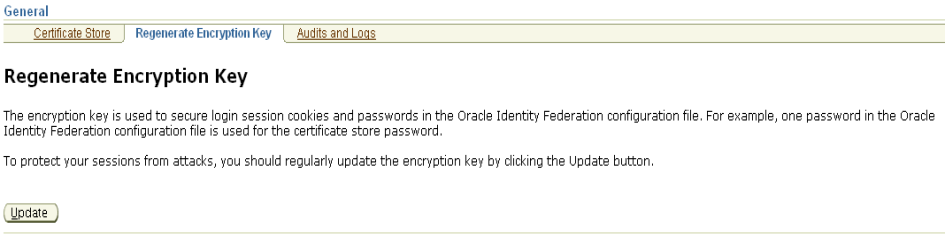
Re-enter the password to confirm the password entry.

Buttons on this page provide the following:

- **Submit** - Updates the certificate store information.
- **Reset** - Cancels and resets the fields.

Regenerate Encryption Key

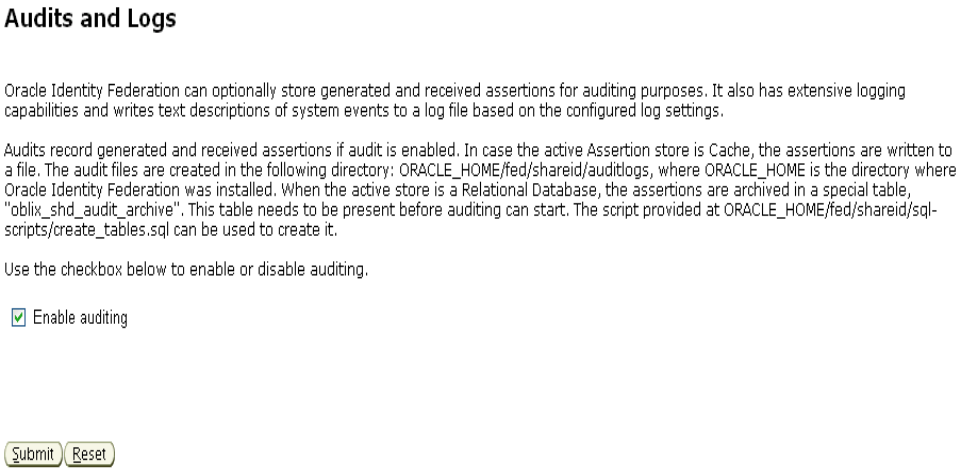
The encryption key is used to secure login session cookies and passwords in the Oracle Identity Federation configuration file. For example, one password in the configuration file is used for the certificate store password.



To protect your sessions from attacks, you should regularly update the encryption key by clicking the **Update** button.

Audits and Logs

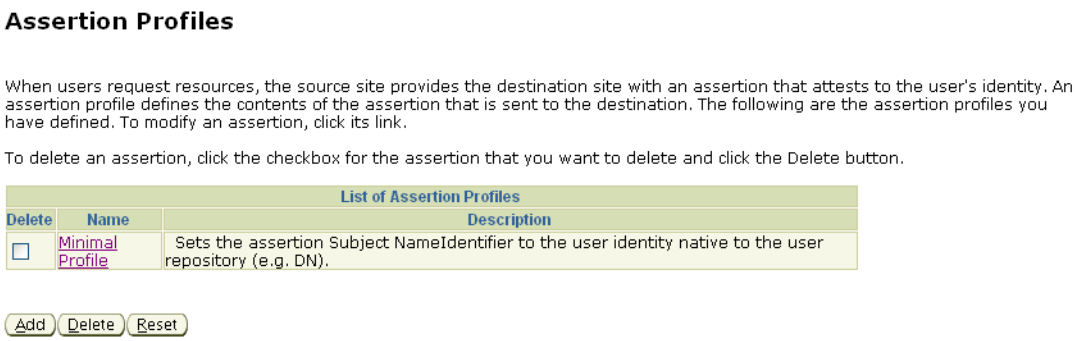
Use this page to allow Oracle Identity Federation to retain generated and received assertions for auditing.



Click **Enable Auditing** and submit to allow the server to save assertion records.

Assertion Profiles

This page displays the SAML assertion profiles you have defined for your source site. An assertion profile defines the contents of the assertion that is sent to the destination.



Name

This is the assertion profile name.

Description

This is a brief description of the assertion profile.

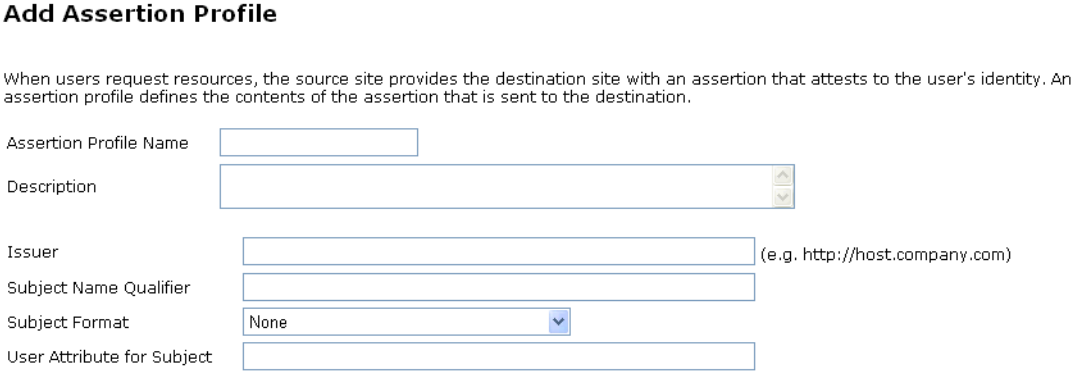
Buttons on this page provide the following:

- **Add** - allows you to add a new assertion profile.
- **Delete** - deletes assertions for which the Delete checkbox is marked.
- **Reset** - cancels and resets the fields.

Add Assertion Profile

Use this page to add a new assertion profile to use for identity assertions from your source site. An assertion profile defines the contents of the assertion that is sent to the destination.

Basic Profile

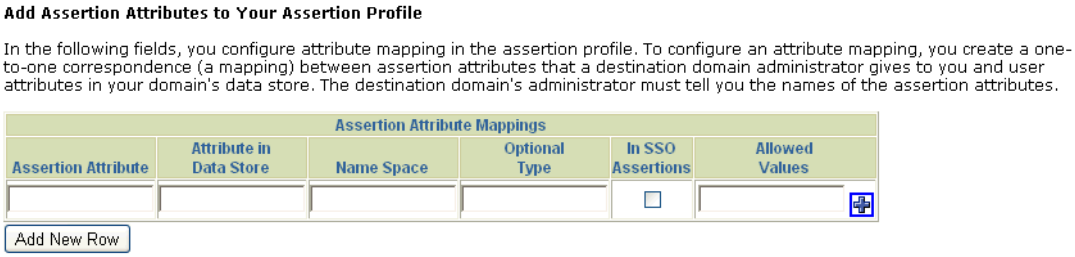


Basic profile information includes:

- the profile name
- a brief description of the profile
- the profile issuer
- the subject name qualifier

- the subject format; options include email, X509 subject name, windows domain, unspecified, or none
- user attribute for subject; if blank, the user DN is used

Assertion Attributes



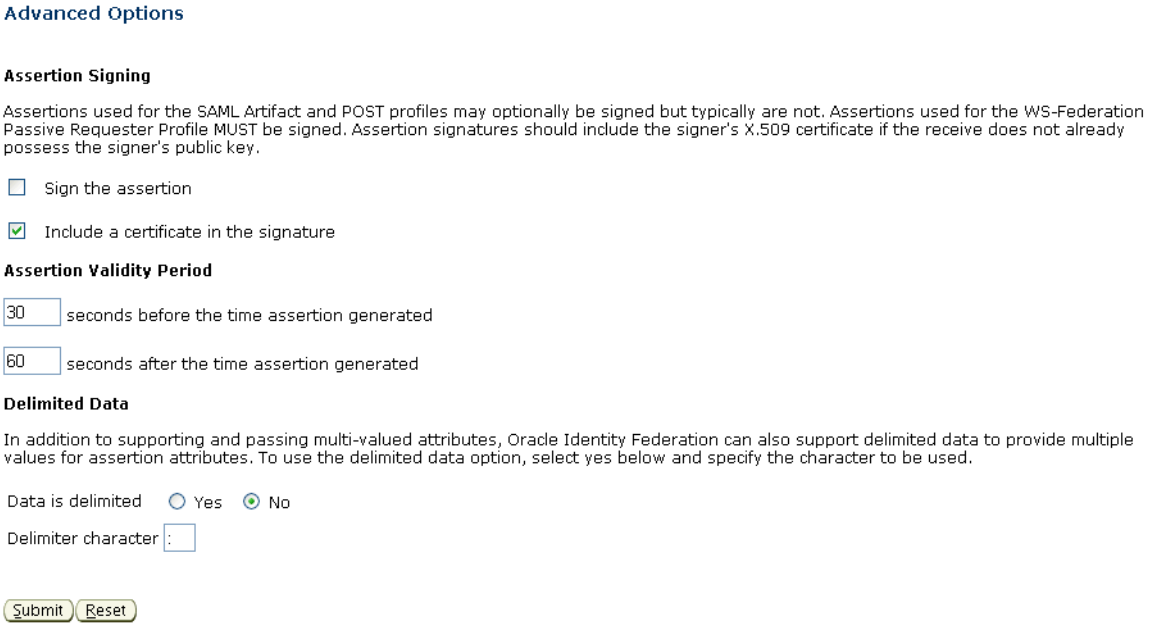
Using the attributes provided by the administrator of the destination domain, provide attribute information for each required attribute. The fields are:

- Assertion Attribute
- Attribute in Data Store
- Name Space
- Optional Type
- In SSO Assertions checkbox
- Allowed Values - These are the permissible values of the attribute.

Assertion Signing

Indicate whether the assertion needs to be signed, and whether to include a certificate in the signature.

For the WS-Federation protocol, the assertion is always signed regardless of the **Sign the Assertion** setting.



Delimited Data

If you check **Data is delimited**, Oracle Identity Federation will break an attribute value, retrieved from the user data store that contains the specified delimiter characters, into multiple assertion attribute values. The delimiter character can be escaped by a backslash ("\"). For example, if the delimiter character is ":" and a user data store attribute value is "A:B\:C:D", the assertion will contain three attribute values - "A", "B:C", and "D".

Edit Assertion Profile

Use this page to modify an existing assertion profile.

Basic Profile Information

Basic profile data includes:

- Assertion Profile Name
- Description - This is a brief description of the profile.
- Issuer - This is the profile issuer, for example `http://host.company.com`.
- Subject Name Qualifier
- Subject Format - Choose from Email Address, X.509 Subject Name, Windows Domain, Unspecified, None, and Other.
- User Attribute for Subject

Add Assertion Attributes to Your Assertion Profile

Use this set of fields to map assertion attributes from a peer site to your data store's user attributes. Attribute mappings include:

- Assertion Attribute Name
- Attribute in Data Store - This is the corresponding attribute in your user store.
- Name Space
- Optional Type
- In SSO Assertions - Specify if this attribute is used in SSO assertions.
- Allowed Values - Enter one or more allowed values (click on the icon to add more values.

Advanced Options

Advanced options include:

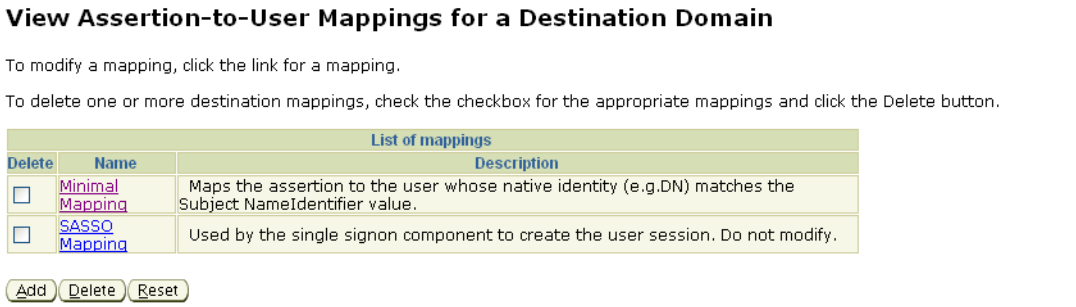
- Assertion Signing - enables you to specify whether and how assertions should be signed.
- Assertion Validity Period - defines how long, in seconds, the assertion is valid prior to and following generation.
- Delimited Data - lets you specify whether multi-valued assertion data is delimited, and if so, specify the delimiter character.

Buttons on this page provide the following:

- **Submit** - updates the assertion profile with your changes.
- **Reset** - cancels and resets the fields.

Destination Mappings

Use this page to view destination mappings, which associate an assertion with a user identity. The mapping allows the destination domain to determine the identity of the local user.



Name

This is the assertion-to-user mapping name. Click on the name to modify its assertion-to-user mappings.

Description

This is a brief description of the assertion mapping.

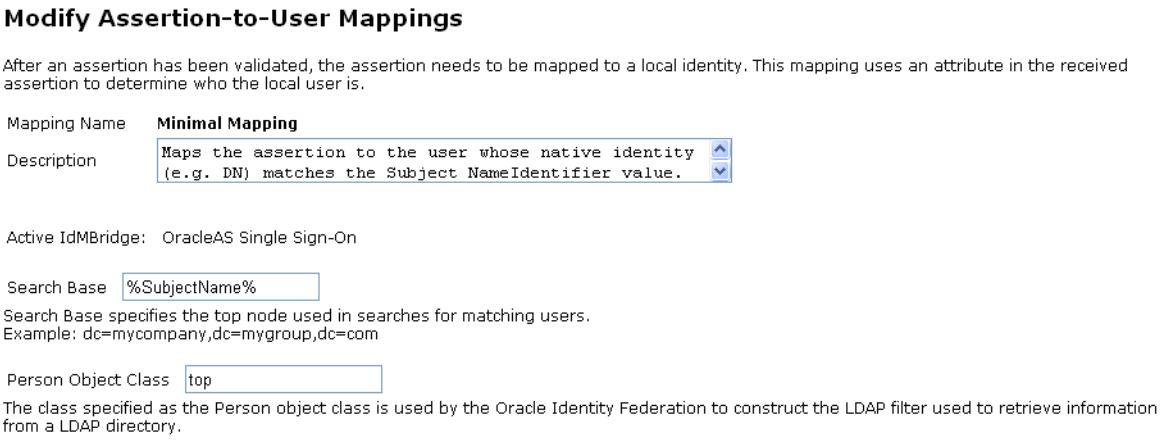
Buttons on this page provide the following:

- **Add** - allows you to add a new assertion mapping.
- **Delete** - deletes mappings for which the Delete checkbox is marked.
- **Reset** - cancels and resets the fields.

Modify Destination Mappings

Use this page to modify an assertion-to-user destination mapping.

> **Note:** If you modify the authentication level for a mapping, you must also modify the authentication level of the SASSO mapping to match.



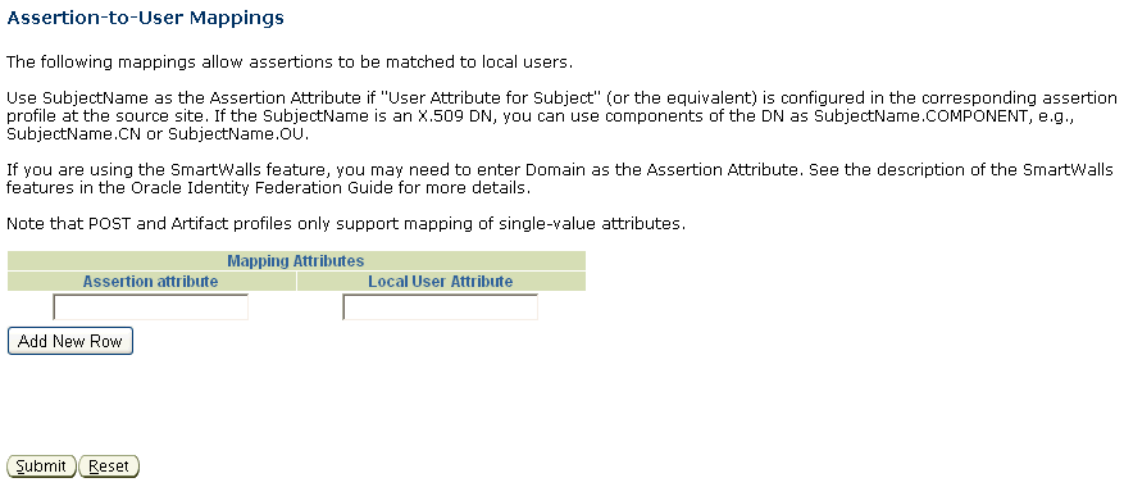
Description

A brief description of the mapping.

Active IdMBridge

Specifies the IdMBridge for the mapping. Valid choices are:

- Oracle Access Manager IdMBridge
- eTrust SiteMinder IdMBridge
- OracleAS Single Sign-On



Use this portion of the page to establish a mapping between an assertion and a local user.

> **Note:** Mapping an assertion attribute containing multiple values to a local user is not supported.

Assertion Attribute

Use an assertion attribute consistent with the corresponding assertion profile at the source.

Local User Attribute

Enter the corresponding local user attribute.

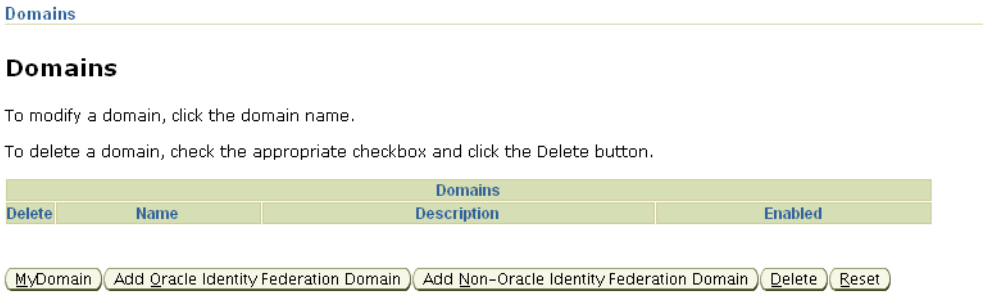
Terminology Note

Local user mapping, known in the previous release as SmartWalls, was a best practice related to mapping an incoming SSO assertion to a user. The technique was intended to thwart a user from one IdP from impersonating a user from another IdP by falsely asserting attributes for that user.

As a rule, SAML 2.0 and Liberty do not use attribute mapping; instead they use opaque name identifiers that are not susceptible to this problem.

Domains

This is the main page for domain configuration.



Name

This is the domain name.

Description

A brief description of the domain.

Enabled

Indicates if this domain is enabled.

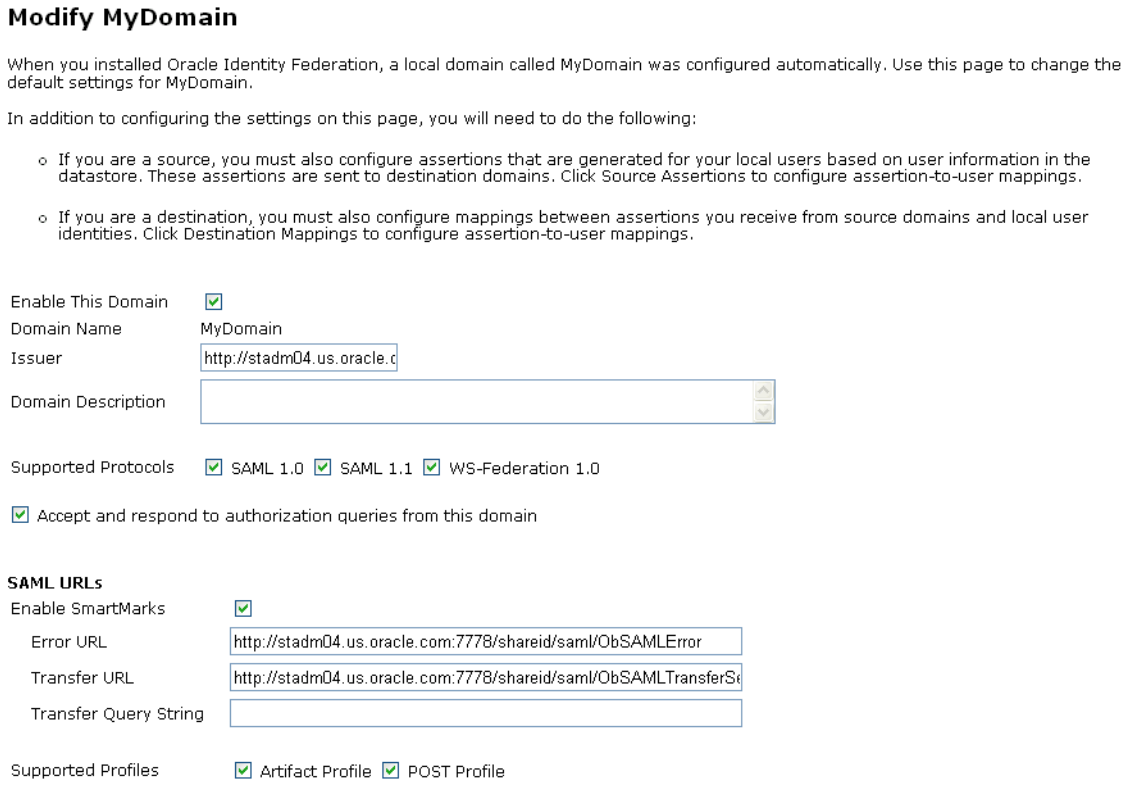
Buttons on this page provide the following domain maintenance features:

- **MyDomain** - allows you to configure information about the local Oracle Identity Federation instance.
- **Add Oracle Identity Federation Domain** - allows you to easily configure a partner domain that is also using Oracle Identity Federation.
- **Add Non-Oracle Identity Federation Domain** - allows you to configure a partner domain using any other SAML 1.x or WS-Federation products.
- **Delete** - deletes the domain record(s) indicated in the Delete checkbox.
- **Reset** - cancels and resets the fields.

Update MyDomain

Use this page to update MyDomain, which is a local domain configured automatically with every Oracle Identity Federation installation.

Basic MyDomain Information



You can update the following:

- **Enable this Domain** - Enables or disables the domain
- **Domain Name** - Protected field
- **Issuer**
- **Domain Description** - A brief description
- **Supported Protocols**

> **Note:** If you disable SAML 1.0, SAML 1.1, or WS-Federation, this takes precedence over disabling it in partner domains. Thus, if WS-federation is disabled in MyDomain, then it is disabled for all partners, even if its enabled for some partner domain configurations.

- Whether MyDomain responds to requests for authorization; if checked, this domain accepts and responds to authorization queries.
- **Enable SmartMarks** - Check the box to enable SP-initiated IdP discovery (formerly SmartMarks - details appear in the following paragraph).
- **Error URL** - This is the URL to which users are redirected in the event of an error.
- **Transfer URL** - This is the redirect URL.
- **Transfer Query String** - This is a query string, if any, to be utilized at the transfer URL.
- **Supported Profiles**

About SP-initiated IdP Discovery

SmartMarks, a proprietary Oracle technique for SP-initiated SSO with SAML 1.x and Oracle Access Manager, has been superseded in SAML 2.0 with the implementation of the SP-initiated IdP discovery using common domain cookies. While the feature is still applicable in the context of SAML 1.x, from now on it is referred to as SP-initiated IdP discovery..

When a resource - protected by an Oracle Access Manager policy using the "Fed SSO - SAML 1.x" authentication scheme - is accessed by a federated user, Oracle Access Manager redirects the access to Oracle Identity Federation. Oracle Identity Federation then redirects the access to a source site to initiate a SAML 1.x SSO for the resource. To do this, Oracle Identity Federation constructs a redirection URL from the Transfer URL and the Transfer Query String configured for the source site and the target URL. For a source site using Oracle Identity Federation, these are:

```
Transfer URL =
http(s)://source-host:port/shareid/saml/ObSAMLTransferService
```

```
Transfer Query String = DOMAIN=dest-domain&METHOD=method&TARGET=
```

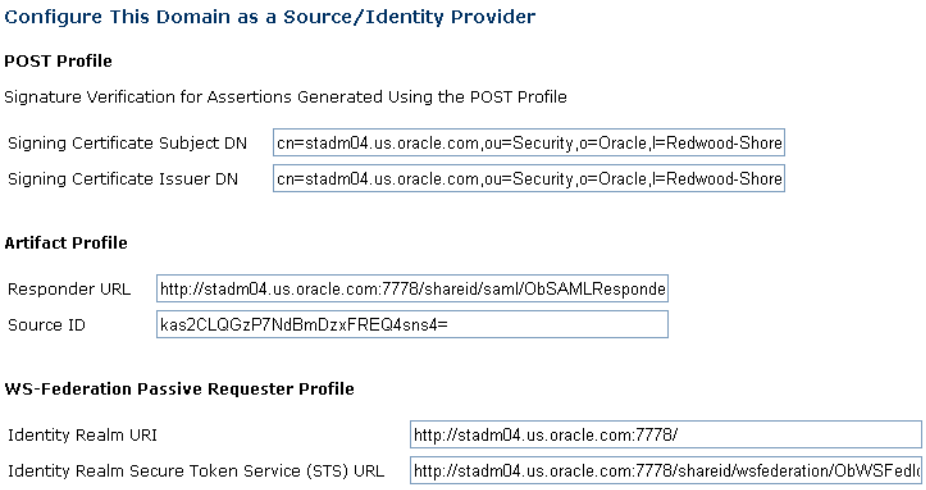
where `dest-domain` is the domain configured at the source site at the Oracle Identity Federation destination, and the method is post or artifact.

So the entire redirection URL is:

```
http(s)://source-host:port/shareid/saml/ObSAMLTransferService?DO
MAIN=source-domain&TARGET=targetURL
```

SP-initiated IdP Discovery can also be configured for source sites using other SAML 1.x products, as long as the targetURL is the last part of the redirection URL.

Configure This Domain as a Source/Identity Provider



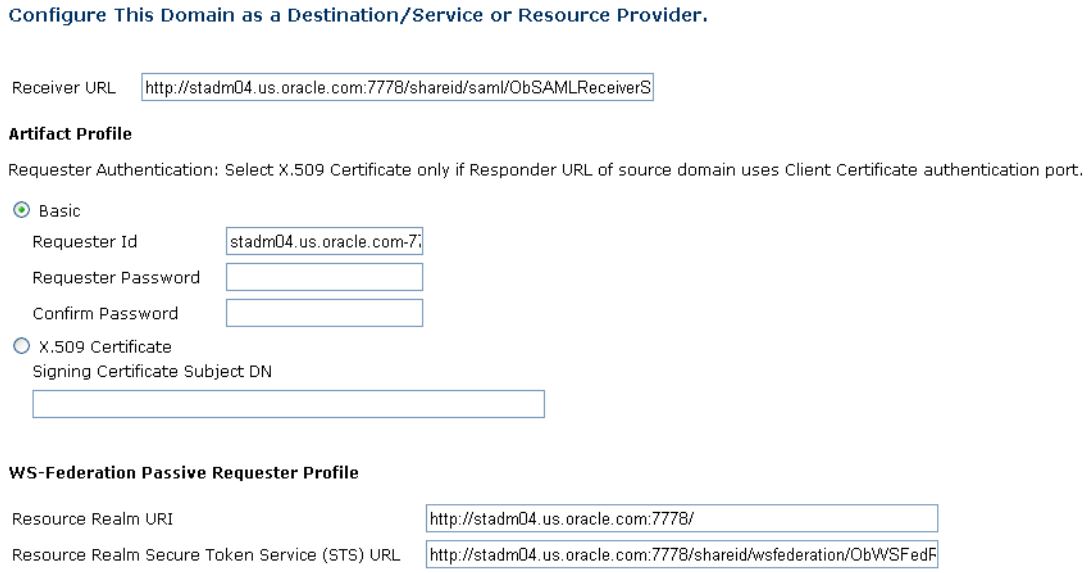
If users from your domain wish to access resources at other domains, configure MyDomain as an IdP.

You can update the following:

- Post profile **signing certificate subject DN**
- Post profile **signing certificate issuer DN**
- Artifact profile **responder URL**
- Artifact profile **source ID**

- The **Identity Realm URI** for the WS-Federation Passive Requester Profile
- The **Identity Realm STS URL** for the WS-Federation Passive Requester Profile

Configure This Domain as a Destination/Service or Resource Provider

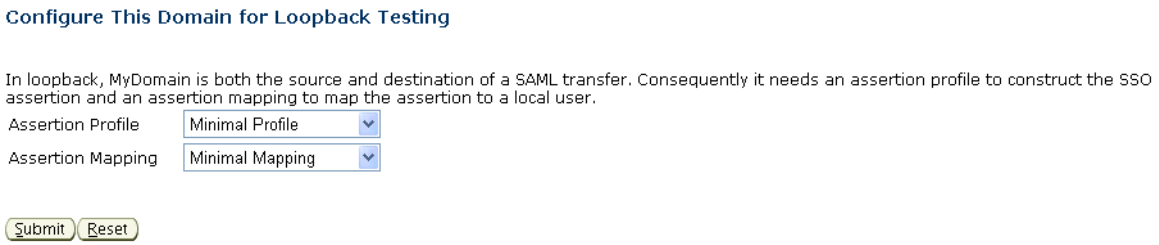


If users from other domains wish to access resources at MyDomain, configure MyDomain as an SP.

You can update the following:

- Receiver URL
- **Artifact Profile (Basic configuration)**
  - Requester ID
  - Requester Password
  - Password Re-entry
- **Artifact Profile (X509 configuration)**
  - X.509 Signing Certificate Subject DN
- **WS-Federation Passive Requester Profile**
  - The **Resource Realm URI**
  - The **Resource Realm STS URL**

Configure this Domain for Loopback Testing



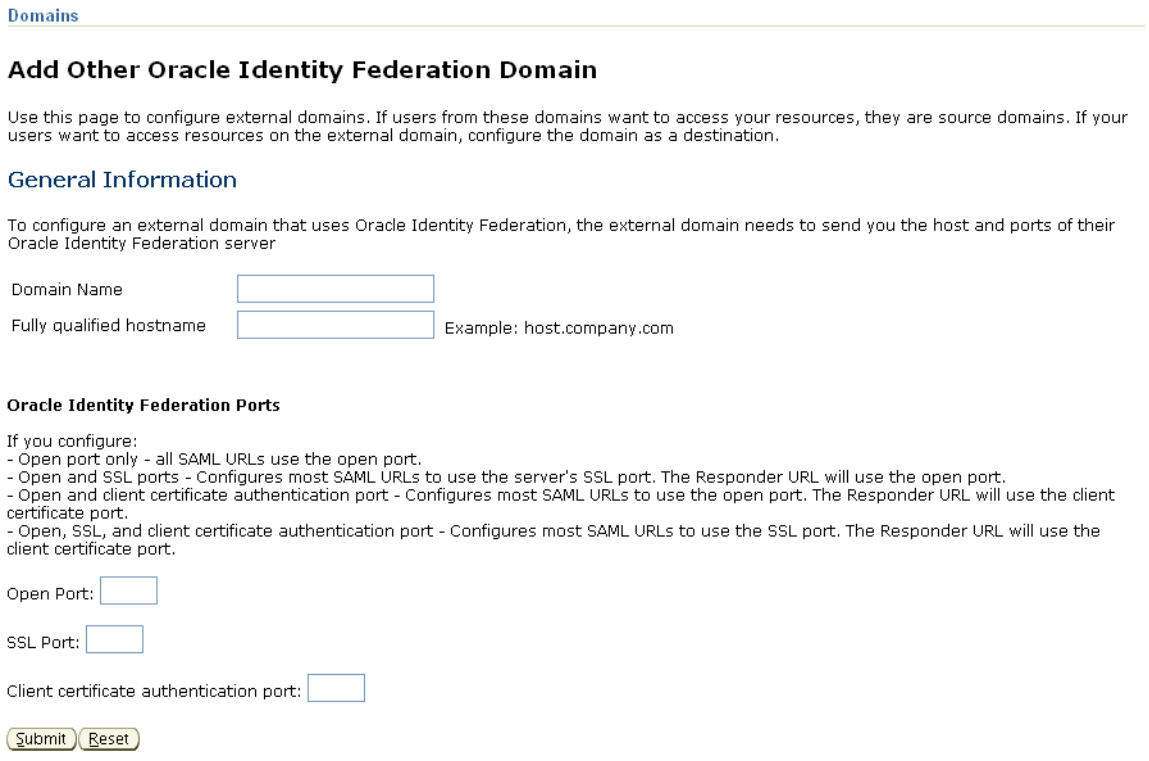
Use these fields to configure MyDomain in loopback mode for testing.

The URL to use for loopback testing is:

```
http(s)://OIF-host:port/shareid/saml/ObSAMLTransferService?DOMAIN=MyDomain&METHOD=method&TARGET=targetURL
```

Add Oracle Identity Federation Domain

Use this page to configure a partner domain that is also using Oracle Identity Federation. This page uses default values for most of the domain configuration items based on Oracle Identity Federation conventions. The administrator of the domain must provide you the host and port information. After you submit this page, the Modify Other Domain page will be returned to allow you to modify the domain configuration if needed. In particular, you need to set the Assertion Profile and Assertion Mapping for the domain.



You can update the following:

- **Domain Name**
- **Fully qualified hostname** (host and port)

- **Oracle Identity Federation Ports**:
  - Open Port
  - SSL Port
  - Client certificate authentication port

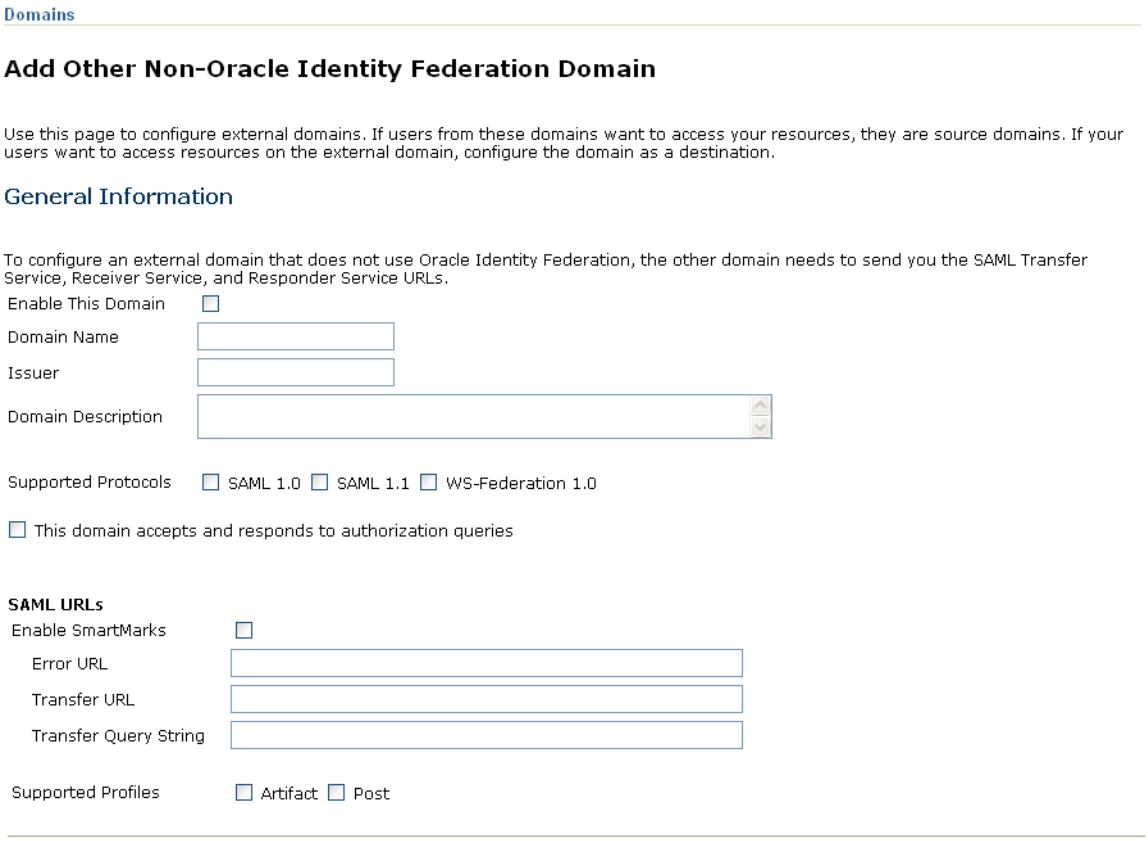
Buttons on this page provide the following domain maintenance features:

- **Submit** - adds the domain record to the SAML 1.x configuration.
- **Reset** - cancels the operation.

Add a Non-Oracle Identity Federation Domain

Use this page to configure an external domain as either an IdP or SP domain. The domain uses a federation server other than Oracle Identity Federation. Besides basic server information, the administrator of the domain must provide you the SAML service URL's.

Basic MyDomain Information

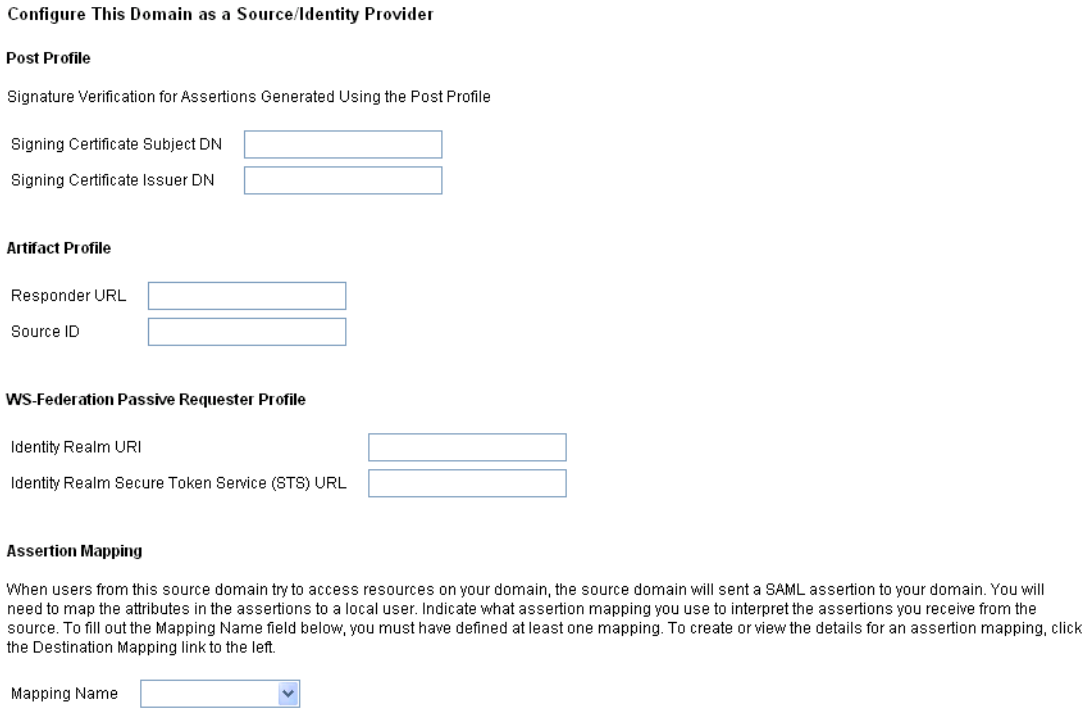


Provide the following:

- **Enable this Domain** - Enables or disables the domain.
- **Domain Name** - This is a protected field.
- **Issuer**
- **Domain Description** - This is a brief description.

- **Supported Protocols**
- Whether the domain responds to requests for authorization
- **Enable SmartMarks**
- **Error URL** - This is the URL to which users are redirected in the event of an error.
- **Transfer URL** - Tis is the redirect URL.
- **Transfer Query String** - This is the query string, if any, to be utilized at the transfer URL.
- **Supported Profiles**

Configure This Domain as a Source/Identity Provider

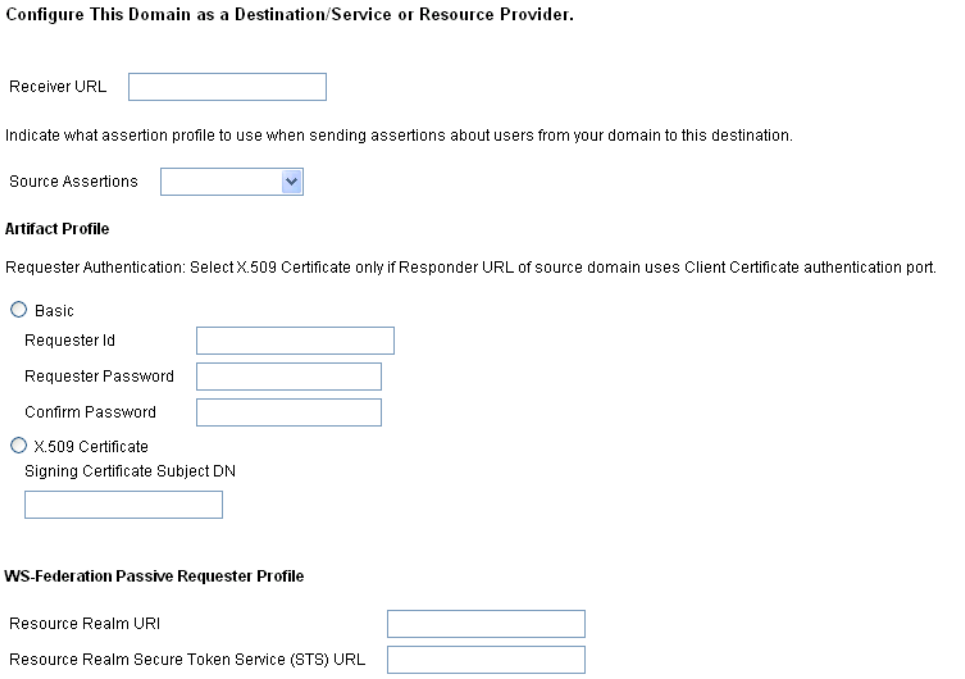


If users from this domain wish to access resources at your domain, configure the domain as an IdP.

You can update the following, depending on the chosen profile (post, artifact, or WS-Federation):

- Post profile **signing certificate subject DN**
- Post profile **signing certificate issuer DN**
- Artifact profile **responder URL**
- Artifact profile **source ID**
- The **Identity Realm URI** for the WS-Federation Passive Requester Profile
- The **Identity Realm STS URL** for the WS-Federation Passive Requester Profile

Configure This Domain as a Destination/Service or Resource Provider

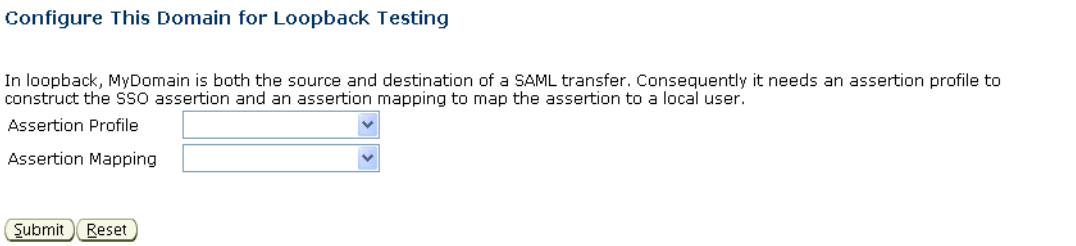


If users from this domain wish to access resources from other domains, configure the domain as an SP.

You can update the following:

- Receiver URL
- **Artifact Profile (Basic configuration)**
  - Requester ID
  - Requester Password
  - Password Re-entry
- **Artifact Profile (X509 configuration)**
  - X.509 Signing Certificate Subject DN
- **WS-Federation Passive Requester Profile**
  - The **Resource Realm URI**
  - The **Resource Realm STS URL**

Configure this Domain for Loopback Testing



Use these fields to configure the domain for loopback testing.

Exchanging SAML 1.x and WS-Federation Configuration Data with Peers

Service URLs and related information about Oracle Identity Federation will need to be shared with trusted peers using the SAML 1.x and WS-Federation protocols. The information described here is obtained from the MyDomain configuration page (**SAML 1.x/WS-Fed** - > **Domains** - > **MyDomain**). Use this information if your site's **circle of trust** contains one or more peer providers, and you need to enter the URL details for an Oracle Identity Federation server.

This section explains how to obtain the service URLs:

- When Oracle Identity Federation is an IdP
- When Oracle Identity Federation is an SP

When Oracle Identity Federation is an IdP

Provide the following configuration data about your Oracle Identity Federation, acting as an identity provider, to the service providers in the circle of trust using SAML 1.x or WS-Federation:

- Issuer: Oracle Identity Federation actually uses the Issuer defined in the assertion profile associated with the partner domain. This allows different issuers to be used for different partners if necessary, but in practice this flexibility is not needed. To avoid confusion, the best practice is to set the Issuer in all the assertion profiles to the issuer in MyDomain.
- If the SP is also an Oracle Identity Federation installation, use:
  - Error URL - to use the error redirection feature
  - Transfer URL - to use the SP-initiated SSO or SP-initiated IdP discovery (previously SmartMarks) feature
  - Transfer Query String - to use the SP-initiated SSO feature:
    * `DOMAIN=<SP-domain>&METHOD=<method>&TARGET=`
    * `SP-domain` is the name of the domain configured for the SP
    * `method` is artifact or post
- If the SAML 1.x Artifact Profile is used:
  - Responder URL - this is the SOAP service to which the SP sends the Artifact request.
  - SourceID - Identifies the IdP to the SP.
- If the SAML 1.x POST Profile is used:

  - If the SP is also an Oracle Identity Federation server:
    * Signing Certificate Subject DN and Issuer DN (used to verify the signed SAML response in the POST).
  - the SP is another SAML implementation:
    * the certificate used to verify the signed SAML response. The certificate can be exported from the keystore as follows:

      ```
      cd ORACLE_HOME/fed/shareid/oblix/config/keystore
      ```

      ```
      keytool -keystore <keystore> -storepass <ias-password>
      -export -alias shareid -rfc -file <file>
      ```

      `ias-password` is the administrator password selected when Oracle Identity Federation was installed.

      `file` is a file to which the certificate is written, in PEM format.

- If the WS-Federation Passive Requester Profile is used:
  - Identity Realm URI - identifies the IdP to the SP.
  - Identity Realm STS URL - this is the URL to which the SP redirects to obtain a security token.

When Oracle Identity Federation is an SP

Provide the following configuration data about your Oracle Identity Federation, acting as a service provider, to the identity providers in the circle of trust using SAML 1.x or WS-Federation:

- If the SAML 1.x Artifact Profile is used:
  - Receiver URL - this is the URL to which the IdP redirects with the artifact.
  - For IdP authentication of the SP's SOAP Artifact request:
    * If using HTTP basic - Requester ID and Password
    * If using SSL client certificate authentication:

      If the IdP is also Oracle Identity Federation - Signing Certificate Subject DN

      If the IdP is another SAML implementation - the certificate, exported as described above.
- If the SAML 1.x POST Profile is used:
  - Receiver URL - this is the URL to which the IdP POSTs the SAML response.
- If the WS-Federation Passive Requester Profile is used:
  - Resource Realm URI - this URI identifies the SP to the IdP.
  - Resource Realm STS URL - this is the URL to which the IdP POSTs the security token.

Configuring Attribute Sharing

Attribute sharing is a joint feature of Oracle Access Manager and Oracle Identity Federation that implements the SAML Attribute Sharing Profile for X.509 Authentication-Based Systems. In this profile, a user who requests a protected resource or service is authenticated via SSL client X.509 certificates, but authorization is

performed with user attributes retrieved from the user's home organization using the SAML protocol. The user's home organization is the identity provider (IdP), and the organization performing authentication and authorization is the service provider (SP).

This section explains how to configure Oracle Access Manager and Oracle Identity Federation for attribute sharing. It contains these topics:

Components Used for Attribute Sharing

Attribute sharing uses several Oracle Access Manager and Oracle Identity Federation components. The instructions assume that these components have been installed and configured for their normal operation.

Service Provider Components

- *Web Server with an Access Manager WebGate* - for HTTP requests for a protected URL, performs the SSL client certificate authentication and enforces the access decision from the Oracle Access Manager server.
- *Oracle Access Manager* - performs authentication and authorization for the WebGate. Uses these custom plugins for the attribute sharing feature:
  - *authz_attribute Authentication Plugin* - passes the certificate SubjectDN to the `authz_attribute` authorization plugin
  - *authz_attribute Authorization Plugin* - uses the Attribute Requester Service to retrieve attribute values for the user's SubjectDN and evaluates a rule expression with the attribute values to determine if access is allowed

  Note: The authentication and authorization plugins use the same `authz_attribute` library.

- *Oracle Identity Federation Attribute Requester Service* - sends a SAML 2.0 attribute query to the IdP Attribute Responder Service determined by the user's SubjectDN, and returns the retrieved attributes to the `authz_attribute` plugin.

Identity Provider Component

Oracle Identity Federation Attribute Responder Service or other SAML 2.0-compliant federation product - receives a SAML 2.0 attribute query from the SP Attribute Requester Service, retrieves the attributes for the specified user (subject to local policy controls), and returns a response with the attributes to the Attribute Requester Service.

Remote and Local Users

In addition to remote users authorized by SAML attribute retrieval, the protected resource may also be accessed by local users with attributes defined within the service provider Oracle Access Manager user directory. Local users, configured as discussed

here, are detected by the `authz_attribute` authentication plugin, which returns a `Failure` status. The authentication scheme described later uses this status to create a local session for the user, and authorization rules with local LDAP filters can be applied.

Configuring the Oracle Access Manager Plugins

Take these steps to configure the Oracle Access Manager plugins:

1. Log in to the Access Server host as the user who installed the Access Server.
2. Create the directory `INSTALLDIR/oblix/config/attributePlugin`, if it does not already exist.
3. Edit or create the `config.xml` file in the `INSTALLDIR/oblix/config/attributePlugin` directory, using the sample `config.xml` file shown here as a template.
4. Edit the attributes and elements of the `config.xml` file as required.
5. Restart the Access Server for changes to take effect.

Sample config.xml File

```
<Config LogLevel="audit" WaitTime="30" SizeLimit="0" MaxConnections="5"
  InitialConnections="2"
    Authn="basic" Username="coreid-as-ashost-6021"  Password="xyzzy"
      KeyPassword="abcde"
    CacheTimeout="3600" MaxCachedUsers="1000" HeaderKeyLength="128"
      RequestFormat="values">
    <Mapping Local="true">
        <DN>O=Company,C=US</DN>
    </Mapping>
    <Mapping URL="https://fed1.company.com:7777/fed/ar/soap">
        <DN>O=PeerA,C=US</DN>
        <DN>O=PeerB,C=US</DN>
    </Mapping>
    <Mapping URL="https://fed2.company.com:7777/fed/ar/soap"
      RequestFormat="all">
        <DN>O=PeerC,C=US</DN>
        <DN>O=PeerD,C=US</DN>
    </Mapping>
    <Mapping URL="https://fed3.company.com:7777/fed/ar/soap">
        <DN>C=US</DN>
    </Mapping>
</Config>
```

Configuration Parameters

The configuration parameters are:

- LogLevel - Controls the amount of information logged to `INSTALL_DIR/oblix/logs/authz_attribute_plugin_log.txt`.
  - `off` - Nothing is logged except errors (this is the default).
  - `audit` - One line is logged for each authentication request, showing the access decision, the user's certificate subject DN or local directory DN, and the HTTP operation and the local part of the requested URL.
  - `debug` - Logs extensive information useful in debugging problems.

- HTTP connection parameters (`authz_attribute` plugin to the Oracle Identity Federation Attribute Requester Service), consisting of:
  - WaitTime - This is the time in seconds to wait for a response; default is 30 seconds.
  - SizeLimit - This is the maximum size in bytes of HTTP messages sent and received (default is unlimited, 0 means unlimited).
  - MaxConnections - This is the maximum number of concurrent HTTP connections (default is 5).
  - InitialConnections - This is the number of current HTTP connections opened initially (default is 2).
- Parameters for authentication of the `authz_attribute` plugin to the Oracle Identity Federation Attribute Requester Service, including:
  - Authn - authentication method
    * none - no authentication
    * basic - use HTTP basic authentication with Username and Password (default)
    * cert - use SSL client certificate authentication using `key.pem`, `cert.pem`, and `KeyPassword`
  - Username - This is the username for basic authentication.
  - Password - This is the password for basic authentication.
  - KeyPassword - This is the password for `key.pem` for SSL client certificate authentication.
- Attribute value cache parameters, including:
  - CacheTimeout - This is the time, in seconds, that cached attribute values will be held before requiring updated values (default 3600 seconds - 1 hour; 0 disables caching).
  - MaxCachedUsers - This is the maximum number of users with cached attribute values; if the cache is full, the least recently used unexpired entries will be reclaimed (default is 1000).
- Mappings of subject DNs to Attribute Requester Service URLs. For each Attribute Requester Service, specify:
  - URL - the URL for the service, of the form `%HTTP_PROTOCOL%://%OIF_HOST%:%OIF_PORT%/fed/ar/soap`, where:
    * `%HTTP_PROTOCOL%` - `http` or `https`
    * `%OIF_HOST%:%OIF_PORT%` - This is the host and port of the Oracle Identity Federation server.

    For example: `https://fed1.company.com:7777/fed/ar/soap`
  - Local - if true, the matching users are local and an Attribute Requester Service is not used. If true, the URL parameter is ignored.
  - DN - one or more elements specifying a DN pattern to match against the user Subject DN; the pattern is simply the right most components of the DN. For example: `O=PeerA,C=US`.
- Attribute query properties - The RequestFormat parameter determines the attributes and values returned in an attribute response. RequestFormat overrides

authorization rules; for example, if an authorization rule specifies both attributes and values, but RequestFormat specifies names, the query omits values. RequestFormat can be specified with these options:

– **RequestFormat="values"**

  The AttributeQuery contains attribute names and values taken from the authorization rule's ruleExpression. The Attribute Responder will only return user attributes and values that are in the AttributeQuery. This is the default setting. This setting minimizes the amount of memory used for cached attribute values (values are only requested when needed for authorization), at the cost of more frequent attribute requests.

– **RequestFormat="names"**

  The AttributeQuery contains attribute names but not values taken from the ruleExpression. The Attribute Responder returns all the user's values for the named attributes, subject to any Responder policies controlling access to the attributes values. This setting provides a trade-off between cache memory usage and attribute requests that is somewhere between the "values" and "all" setttings. *Note*: With this setting, the AttributeQuery does not disclose to the IdP what attribute values are required for authorization; for security reasons, this might be preferred over the "values" setting.

– **RequestFormat="all"**

  The AttributeQuery does not contain any attribute names or values. The Attribute Responder returns all the attributes and values for the user subject to any Responder policies controlling access to the attributes values. This setting minimizes the number of attribute requests (only one request per user), at the cost of more memory used for caching attribute values before they are used (and may never be used) for authorization. This setting works best when the Attribute Responder policies have been reasonably configured to return only attributes that the SP might want. *Note*: With this setting, the AttributeQuery does not disclose to the IdP what attributes are required for authorization; for security reasons, you may prefer this over the "values" and "names" settings.

As illustrated in the sample config.xml file, the RequestFormat parameter can appear in the `<Config>` element, where it sets the default request format, and in the `<Mapping>` elements, where it sets the request format for subject DNs covered by the mappings.

Mapping Examples for the Sample Configuration

Here are some mapping examples for the sample `config.xml` configuration file shown earlier.

User Subject DN	Maps to URL
E=john.smith@company.com,CN=John Smith, OU=Development,O=Company,C=US	local
E=betty.jones@peera.com.CN=Betty Jones,OU=Marketing,O=PeerA,C=US	https://fed1.company.com:7777/osfs/sp/soap
E=sally.smith@peerd.com,CN=Sally Smith,OU=Marketing,O=PeerD,C=US	https://fed2.company.com:7777/osfs/sp/soap
E=bill.jones@peerx.com,CN=Bill Jones,OU=Finance,O=PeerX,C=US	https://fed3.company.com:7777/osfs/sp/soap

Configuring SSL and Client Certificate Authentication

Use these steps to configure HTTPS and SSL client certificate authentication:

1. If HTTPS is used between the `authz_attribute` plugin and at least one Attribute Requester Service, set up the trusted CA list in `INSTALL_DIR/oblix/config/attributePlugin/cacerts.pem`. For each CA that certifies an Attribute Requester service, add the `PEM` formatted certificate (including the -----BEGIN CERTIFICATE----- and -----END CERTIFICATE-----) to `cacerts.pem`.
2. If SSL client certificate authentication is used between the `authz_attribute` plugin and at least one Attribute Requester Service, set up the `key.pem` and `cert.pem` files:
   - Generate the private key and certificate request using the `openssl` utility included with Oracle Access Manager with these steps:
     - `cd INSTALL_DIR/oblix/tools/openssl`
     - `openssl req -config openssl.cnf -newkey rsa:1024 -keyout ../../config/attributePlugin/key.pem -out ../../config/attributePlugin/req.pem`
   - Send `INSTALL_DIR/oblix/config/attributePlugin/req.pem` to your CA to get a certificate.
   - Copy the generated certificate to `INSTALL_DIR/oblix/config/attributePlugin/cert.pem`.
3. Restart the Access Server to ensure that the plugin uses the PEM files.

Configuring Oracle Access Manager Schemes and Policies

This section explains how to configure Oracle Access Manager schemes and policies for Oracle Identity Federation. It contains these sections:

- Configuring the Attribute Sharing Authentication Scheme
- Configuring the Attribute Sharing Authorization Scheme
- Configuring an Oracle Access Manager Policy using Attribute Sharing

Configuring the Attribute Sharing Authentication Scheme

Take these steps:

1. Log in to the Oracle Access Manager System Console as a Master Access Administrator. Select the Access System Configuration panel and Authentication Management.
2. Click on Add and fill out the Define a New Authentication Scheme form.
   - Name: OIF Attribute Sharing
   - Description: Performs an SSL client certificate authentication for Oracle Identity Federation Attribute Sharing authorization
   - Level: set based on the requirements of the protected resources; should be higher than any password schemes
   - Challenge Method: X509Cert
   - Challenge Parameter: ssoCookie:Expires=Tue, 1 Nov 2005 00:00:00 GMT

Note: To ensure that this authentication scheme is run on every access to protected resources, this challenge parameter forces the browser to discard the ObSSOCookie which forces Oracle Access Manager to re-authenticate.

- SSL Required: yes
- Enabled: no (until the plugins are configured...)

  Click **Save** and commit the changes.

3. Select the **Plugins** tab and click **Modify**. Add the plugins and parameters shown in the table. To enter built-in plugins, select the plugin name from the drop-down list.To enter custom plugins, select Custom Plugin from the drop-down list and enter the plugin name in the text box. Click on **Save** when all plugins have been added.

Plugin Name	Type	Plugin Parameters
authz_attribute	custom	(none)
cert_decode	built-in	(none)
credential_mapping	built-in	obMappingBase="MAPPING_ BASE",obMappingFilter="(uid=O blixAnonymous)"
credential_mapping	built-in	obMappingBase="MAPPING_ BASE",obMappingFilter="(&(&(ob jectclass=PERSON_ OBJECTCLASS) (USER_ ATTRIBUTE=%certSubject.FIELD %))(|(!(obuseraccountcontrol=*))(obuseraccountcontrol=ACTIVATE D)))"

Here is an example:

Plugin Name	Plugin Parameters
authz_attribute	
cert_decode	
credential_mapping	obMappingBase="o=Company,c= US", obMappingFilter="(uid=OblixAno nymous)"
credential_mapping	obMappingBase="o=Company,c= US", obMappingFilter="(&(&(objectclas s=inetorgperson)(mail=%certSubje ct.E%)) (|(!(obuseraccountcontrol=*))(obu seraccountcontrol=ACTIVATED))) "

4. Select the **Steps** panel. Add the following steps:

Step Name	Active Plugins	Purpose
SubjectDN	authz_attribute	Extracts SubjectDN from certificate; determines if user is remote or local
RemoteUser	first credential_ mapping	Creates an anonymous session for a remote user
LocalUser	cert_decode, second credential_mapping	Creates a real session for a local user

5. Select the **Authentication** Flow panel, click **Modify** and set the flow shown in the table. *Note*: The `authz_attribute` plugin returns `Success` if the user is remote and `Failure` if the user is local.

Step Name	Initiating Step	On Success Next Step	On Failure Next Step
Default Step	no	Stop	Stop
SubjectDN	yes	RemoteUser	LocalUser
RemoteUser	no	Stop	Stop
LocalUser	no	Stop	Stop

6. Return to the **Steps** panel and remove the now-unused Default step.
7. Return to the **General** panel and enable the authentication scheme.

Configuring the Attribute Sharing Authorization Scheme

Take these steps to configure the attribute sharing authorization scheme:

1. Log in to the Oracle Access Manager System Console as a Master Access Administrator. Select the **Access System Configuration** panel and **Authorization Management**.
2. Click **Add** and fill out the **Define a new Authorization Scheme** form:
   - Name: OIF Attribute Sharing
   - Description: Uses Oracle Identity Federation to obtain attributes for remote users to evaluate the rule expression
   - Shared Library: oblix/lib/authz_attribute
   - Plugin is Managed Code: no
   - Managed Code Name Space: (none)
   - User Parameter: RA_SubjectDN (*Note*: This uses the "reverse action" feature to obtain the `SubjectDN` header set by the `authz_attribute` plugin.)
   - Required Parameter
     - Name: ruleExpression
     - Value: (none) (*Note*: Each access policy authorization rule will supply the rule expression.)
   - Click **Save** to commit the changes.

Configuring an Oracle Access Manager Policy using Attribute Sharing

Take these steps to configure an Oracle Access Manager policy using the Attribute Sharing profile:

1. Log in to Oracle Access Manager as a Master or Delegated Access Administrator. Select **Create Policy Domain**.
2. Fill out the **General** panel form:
   - Name: as appropriate (for example, Oracle Identity Federation Attribute Sharing Test)
   - Description: as appropriate

   Click **Save**.
3. Select the Resource panel and add one or more resource URL prefixes to protect (for example, /attribute-test).
4. Select the **Authorization Rules** panel and add an authorization rule for each set of attributes (represented as a rule expression) required for a remote user.
   - Select **Custom Authorization Scheme** and click **Add**.
   - Fill out the authorization rule form and click **Save**.
     - 1. Name: as appropriate (for example, Peer Marketing VP)
     - 2. Description: as appropriate
     - 3. Authorization Scheme: OIF Attribute Sharing
   - Select the **Plugin Parameters** panel, click **Modify**, and set the `ruleExpression` parameter as specified in the table. *Note*: White space is allowed around =, !=, &, and `|`.

Element	Syntax	Meaning
name	alphanumeric string including '-', '_', and '.'	Name of attribute to request from the user's identity provider
value	one of "string", any, or null	Required attribute value. With Oracle COREid Access 7.0.4 the string is restricted to Latin-1 characters. With Oracle Access Manager 10.1.4 and later, the string can contain any Unicode characters. The `any` value retrieves and matches all values for the attribute. The null value matches a SAML <Attribute> with the xsi:nil="true" attribute.
comparison	name = value, name != value, or (expression)	True if the user has/does not have the attribute value
and-clause	comparison & comparison	True if both comparisons are true.
or-clause	comparison | comparison	True if either comparison is true. & has higher precedence than !.

Name examples:

- title = "VP" & function = "Marketing"
- title = "VP" | title = "Director"

  - title = "VP" & (function = "Marketing" | function = "Finance")
  - title = any & function = any
- Set any timing conditions or actions as desired for the authorization rule.
- Return to the **General** panel and enable the rule.

5. Select the **Authorization Rules** panel and add an authorization rule for any local user attributes.
   - Select **Oracle Authorization Scheme** and click **Add**.
   - Fill out the authorization rule form:
     - Name: as appropriate (for example, Company Marketing VP).
     - Description: as appropriate
     - Enabled: yes
     - Allow Takes Precedence: no

     Click **Save**.
   - Select the **Allow Access** panel, click **Modify**, and add an LDAP filter for the local attributes. You can use the Query Builder in the Oracle Access Manager Identity User Manager (Configuration -> Delegate Administration -> Build Filter). For example:

     ldap:///o=Company,c=US??sub?(&(title=VP)(function=Marketing))
   - Set any timing conditions or actions as desired for the authorization rule.
   - Return to the **General** panel and enable the rule.
6. Select the **Default Rules** panel and add the default authentication rule:
   - Name: as appropriate
   - Description: as appropriate
   - Authentication Scheme: OIF Attribute Sharing

   Click **Save**.
7. Select the **Authorization Expression** panel and add the default authorization rule:
   - Select the applicable remote authorization rule as defined above and click **Add** (for example, Peer Marketing VP).
   - If there is a corresponding local authorization rule, select `OR` and **Add** the local authorization rule. (for example, Peer Marketing VP | Company Marketing VP).

   Click **Save**.
8. Alternatively, you can add policies to the policy domain with authorization expressions for subsets of the protected URLs.

Configuring Oracle Identity Federation as an SP Attribute Requester

Take these steps to configure Oracle Identity Federation as an attribute requester in service provider mode:

1. Access the Oracle Identity Federation Administration Console at `http://sp-hostname:port/fedadmin`, with username `oif_admin` and the password that was set during installation.

2. Enable the Attribute Requester functionality:
   - Go to the **Server Configuration** tab and select the Service Provider section
   - Click on **SAML 2.0** to edit SAML 2.0 Service Provider Properties
   - Check **Attribute Requester Enabled**
   - Click **Save**
3. Add the SAML 2.0 IdP metadata to the Circle Of Trust:
   - Go to the Server Configuration section.
   - Select the Circle of Trust section
   - Enter the location of the SAML 2.0 IdP metadata in the Add Trusted Provider section and an additional description
   - Click **Add**
4. Configure the DN to IdP mapping:
   - Click on the **Server Configuration** tab and go to the **Service Provider** section

     Go to **Attribute Requester**.
   - To add a mapping:
     - Click **Add Mapping**
     - Enter the DN or sub-DN (for example, c=us)
     - Map this DN or sub-DN to an existing IdP
     - Repeat the operation if necessary
   - Click **Apply**
   - Click **Refresh Server**
5. Enable Certificate Validation:
   - Go to the **Server Configuration** section
   - Click **Certificate Validation Enabled**. (*Note*: Setting a property on this page requires a server restart)
   - Go to the Enterprise Manager Console, with the username `ias_admin` and the password that was set up during installation
   - Select OC4J_FED in the **System Components**
   - Click **Restart**

   Note: Certificate validation is optional.
6. Configure Certificate Validation:
   - Go to the **Server Configuration** section.
   - Select the Certificate Validation section.
   - Add Trusted CAs or CRLs. (*Note*: if Certificate Validation is enabled, a Trusted CA is required to validate signatures).
   - Click **Done**.
   - Click **Refresh Server**.

Note: Certificate validation is optional.

7. Enable Encryption:
   - Go to the Server Configuration section.
   - Select the Service Provider section and select the SAML 2.0 section.
   - Check **Send Encrypted NameIDs** to encrypt the Name Identifiers in the `AttributeQuery` to the Attribute Responder.

     Note: Encryption is optional.

   - Check **Send Encrypted Attributes** to encrypt the Attributes in the `AttributeQuery` to the Attribute Responder.
   - Click **Save**.
   - Click **Refresh Server**.
8. The Attribute Requester service is available at `http://sp-hostname:port/fed/ar/soap`.

If Using Basic Authentication

If using basic authentication between the plug-in and Oracle Identity Federation, you need to add the following to the `httpd.conf` file of the OHS server for your Oracle Identity Federation instance:

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

A user passwords file must also be created using the `htpasswd` utility. In the above example, the `AuthUserFile` containing the users and their passwords points to the `/private/oifpassword` file, in which the user **alice** is defined.

This example creates such a file by adding the user **alice**:

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

If Using Client Certificate Authentication

If using client certificate authentication, see "Configuring SSL Server on Oracle Identity Federation" on page 6-135.

Configuring Oracle Identity Federation as an IdP Attribute Responder

Use these steps to configure Oracle Identity Federation as an attribute responder in IdP mode:

1. Save the SAML 2.0 SP Metadata:

- Go to URL `http://sp-hostname:port/fed/sp/metadatav20`.
- Save the XML file to the local disk.

Note: The metadata URL applies only if the SP is using Oracle Identity Federation. If the SP is using some other SAML implementation, the process of obtaining the metadata will be determined by that implementation.

---

2. Access the Oracle Identity Federation Administration Console at `http://sp-hostname:port/fedadmin`, with username `oif_admin` and the password that was set during installation.
3. Enable the Attribute Responder functionality:
   - Go to the Server Configuration tab, select the Identity Provider section and select the SAML 2.0 section to edit SAML 2.0 IdP properties.
   - Check **Attribute Responder Enabled**.
   - Click **Save**.
4. Map the X.509 Name ID:
   - Go to the **Server Configuration** section.
   - Select the Identity Provider section and select the SAML 2.0 section.
   - Click on **Assertion NameID Formats**.
   - Check that the X.509 Subject Name is enabled and mapped to the correct user entry attribute from the user repository:
     - Use `dn` to map the X.509 Subject Name to an entry's Distinguished Name, or
     - use any attribute from a user entry

Note: The attribute selected for the X509 Subject Name must exactly match the client certificate subject DN, following the format specified in RFC 2253. If unsure of the format, you can perform a test with the SP and look at the Subject NameIdentifier value sent from the SP, which is logged in `ORACLE_HOME/fed/log/federation-msg.log`.

   - Click **Apply**.
   - Click **Save** on the Edit SAML 2.0 Identity Provider Properties page.
   - Click **Refresh Server**.
5. Add the SAML 2.0 SP metadata to the Circle Of Trust:
   - Go to the **Server Configuration** tab.
   - Select the Circle of Trust section.
   - Enter the location of the SAML 2.0 SP metadata in the Add Trusted Provider section and enter an additional description.
   - Click **Add**.
6. Configure the Attribute Mappings for the SP Attribute Requester:

- Go to the Server Configuration section.
- Select the Circle of Trust section.
- Select the SP Attribute Requester entry and click **Update**.
- Click on Attribute Mappings.
- To add attribute mapping:
  - Click **Add Another Row**.
  - Enter the user repository attribute name in the **User Attr Name** column.
  - In **Assertion Attr Name**, enter the identifier used in the `AttributeQuery` or Assertion to reference the attribute.
  - Leave the **Format or Namespace** column empty for SAML 2.0.
  - Repeat the operation to add other attribute mappings.
- Click **Apply**.
- Click **Apply** on the Edit Trusted Provider page.
- Click **Refresh Server**.

Note: For an SP using Oracle Identity Federation, the Assertion Attr Name is determined by the attribute name in a ruleExpression as set in "Configuring an Oracle Access Manager Policy using Attribute Sharing" on page 6-106. The attribute names must be agreed upon between the IdP and SP.

---

7. Enable Certificate Validation:
   - Go to the Server Configuration section.
   - Click **Certificate Validation Enabled**. (*Note*: Setting a property on this page requires a server restart).
   - Go to the Enterprise Manager Console, with the username `ias_admin` and the password that was set up during installation.
   - Select OC4J_FED in the System Components.
   - Click **Restart**.

Note: Certificate validation is optional.

---

8. Enable Encryption:
   - Go to the Server Configuration section.
   - Select the Identity Provider section and select the SAML 2.0 section.
   - Check **Send Encrypted NameIDs** to encrypt the name identifiers in the response to the attribute requester.
   - Check **Send Encrypted Attributes** to encrypt the attributes in the response to the attribute requester.
   - Check **Send Encrypted Assertions** to encrypt the assertion in the response to the attribute requester.

- Click **Save**.
- Click **Refresh Server**.

Note: Encryption is optional.

Configuring Oracle Identity Federation for SSL

To configure SSL on the Oracle Application Server, refer to "Using SSL with Oracle Identity Federation" on page 6-133

Web Services Interface for Attribute Sharing

This section describes the Oracle Identity Federation's Attribute Requester Service Interface. It contains these topics:

Overview of the Service Interface

The Attribute Requester Service provides a request/response interface using the SOAP POST protocol. The service supports the X.509 authn-based attribute sharing profile and follows the SAML 2.0 `<AttributeQuery>` convention.

The service can be invoked to send `samlp:AttributeQuery` messages to a remote identity provider.

Here are the steps that are exercised when the web service client sends an `AttributeRequest` to the Oracle Identity Federation/Attribute Requester server:

1. The web service client sends an `AttributeRequest` message using the SOAP protocol.
2. Oracle Identity Federation processes the incoming `AttributeRequest` message, and selects the IdP to which to send the SAML 2 `AttributeQuery`, based on the `Subject` contained in the `AttributeRequest`.
3. Oracle Identity Federation applies, for the specific remote IdP, the atttribute value mapping for the optional attribute values listed in the `AttributeRequest`.
4. Oracle Identity Federation applies, for the specific remote IdP, the atttribute name mapping for the optional attribute listed in the `AttributeRequest`.
5. Oracle Identity Federation sends the `AttributeQuery` to the remote IdP.
6. Oracle Identity Federation receives the response containing the assertion, along with the attributes sent by the IdP.
7. Oracle Identity Federation applies, for the specific remote IdP, the attribute name mapping for the attribute names listed in the assertion's `AttributeStatement`.
8. Oracle Identity Federation applies, for the specific remote IdP, the attribute value mapping for the attribute values listed in the assertion's `AttributeStatement`.

9. Oracle Identity Federation builds the `AttributeResponse` message, and returns it to the web service client in a SOAP response message.

Attribute Request Message

The `AttributeRequest` message issues a request for attribute data about a user.

The `AttributeRequest` specifies these inputs:

- The user's `SubjectDN`. This is a required input.
- Zero or more names of attributes to be retrieved for the user.

The AttributeRequest message is wrapped in a SOAP `Envelope` and `Body` and sent in an HTTP POST request. The message structure is:

```
<SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
   <SOAP-ENV:Body>
      <orafed-arxs:AttributeRequest
xmlns:orafed-arxs="http://www.oracle.com/fed/ar/10gR3">
         <orafed-arxs:Subject>cn=alice,cn=users,dc=us,dc=oracle,dc=com
         </orafed-arxs:Subject>
         <orafed-arxs:Attribute Name="mail">
            <orafed-arxs:Value>alice@oracle.com</orafed-arxs:Value>
            <orafed-arxs:Value>bob@oracle.com</orafed-arxs:Value>
         </orafed-arxs:Attribute>
         <orafed-arxs:Attribute Name="firstname">
            <orafed-arxs:Value>Bobby</orafed-arxs:Value>
            <orafed-arxs:Value>Charles</orafed-arxs:Value>
         </orafed-arxs:Attribute>
         <orafed-arxs:Attribute Name="lastname">
         </orafed-arxs:Attribute>
      </orafed-arxs:AttributeRequest>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

The output rules are as follows:

- Following the SAML 2.0 `<AttributeQuery>` convention, if no attributes are named, all of the user's attributes are returned.
- If one or more attributes are named in the request, only these are returned.
- Attributes are returned subject to the responder's local policy.

Attribute Response Message

The Attribute Requester service returns the `AttributeResponse` message to a SOAP client following an attribute request.

Outputs of `AttributeResponse` include:

- the status of the SAML 2.0 query (Success or Failure, with the reason)

  The client can use this information for logging.
- the `SubjectDN` of the user
- zero or more `<Attribute>` elements, with each element supplying an attribute name and zero or more values

Note the following about returned attribute values:

- All values are UTF-8 strings.

- Following the SAML 2.0 `AttributeQuery` convention, if the requestor is not allowed to see any values for an attribute, the `Attribute` element will be returned with no `Value` elements.
- An attribute value of NULL is represented by `<Value Null="true"/>`.
- The `CacheFor` attribute in the `AttributeResponse` message specifies how long the attribute values can be cached.

The `AttributeResponse` message is wrapped in a SOAP `Envelope` and `Body` and returned in an HTTP 200 OK response. The message structure is:

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Body>
      <orafed-arxs:AttributeResponse
      xmlns:orafed-arxs="http://www.oracle.com/fed/ar/10gR3" CacheFor="1199">
         <orafed-arxs:Status>Success</orafed-arxs:Status>
         <orafed-arxs:Subject>cn=alice,cn=users,dc=us,dc=oracle,dc=com
         </orafed-arxs:Subject>
         <orafed-arxs:Attribute Name="sn">
            <orafed-arxs:Value>Appleton</orafed-arxs:Value>
         </orafed-arxs:Attribute>
         <orafed-arxs:Attribute Name="givenname"></orafed-arxs:Attribute>
         <orafed-arxs:Attribute Name="mail">
            <orafed-arxs:Value>alice@oracle.com</orafed-arxs:Value>
         </orafed-arxs:Attribute>
      </orafed-arxs:AttributeResponse>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Interface WSDL

The WSDL that formally defines the Attribute Requester Service interface is as follows:

```
<?xml version ="1.0" encoding="US-ASCII" ?>
<wsdl:definitions name="AttributeRequesterFed"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
                   xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
                   xmlns:xml="http://www.w3.org/XML/1998/namespace"
                   xmlns:orafed-arxs="http://www.oracle.com/fed/ar/10gR3"
                   xmlns:orafed-arwsdl="http://www.oracle.com/fed/ar/wsdl"
                   targetNamespace="http://www.oracle.com/fed/ar/wsdl">
       <wsdl:types>
              <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
              targetNamespace="http://www.oracle.com/fed/ar/10gR3"
              elementFormDefault="qualified"
       attributeFormDefault="unqualified">
                     <xs:element name="Subject" type="xs:string"/>

                     <xs:complexType name="ValueType">
                         <xs:simpleContent>
                             <xs:extension base="xs:string">
                                 <xs:attribute name="Null" type="xs:boolean"/>
                             </xs:extension>
                         </xs:simpleContent>
                     </xs:complexType>
                     <xs:element name="Value" type="orafed-arxs:ValueType"/>

                     <xs:complexType name="AttributeType">
                             <xs:sequence>
```

```
                                        <xs:element ref="orafed-arxs:Value"
minOccurs="0" maxOccurs="unbounded"/>
                                </xs:sequence>
                                <xs:attribute name="Name" type="xs:ID"/>
                        </xs:complexType>
                        <xs:element name="Attribute"
type="orafed-arxs:AttributeType"/>

                        <xs:complexType name="AttributeRequestType">
                                <xs:sequence>
                                        <xs:element ref="orafed-arxs:Subject"/>
                                        <xs:element ref="orafed-arxs:Attribute"
minOccurs="0" maxOccurs="unbounded"/>
                                </xs:sequence>
                        </xs:complexType>
                        <xs:element name="AttributeRequest"
type="orafed-arxs:AttributeRequestType"/>

                        <xs:complexType name="AttributeResponseType">
                                <xs:sequence>
                                        <xs:element name="Status"
type="xs:string"/>
                                        <xs:element ref="orafed-arxs:Subject"/>
                                        <xs:element ref="orafed-arxs:Attribute"
minOccurs="0" maxOccurs="unbounded"/>
                                </xs:sequence>
                                <xs:attribute name="CacheFor"
type="xs:unsignedInt"/>
                        </xs:complexType>
                        <xs:element name="AttributeResponse"
type="orafed-arxs:AttributeResponseType"/>
                </xs:schema>
        </wsdl:types>

        <wsdl:message name="AttributeRequestMessage">
                <wsdl:part name="body" element="orafed-arxs:AttributeRequest"/>
        </wsdl:message>

        <wsdl:message name="AttributeResponseMessage">
                <wsdl:part name="body" element="orafed-arxs:AttributeResponse"/>
        </wsdl:message>

        <wsdl:portType name="AttributeRequesterServicePortType">
                <wsdl:operation name="AttributeRequestOp">
                        <wsdl:input
message="orafed-arwsdl:AttributeRequestMessage"/>
                        <wsdl:output
message="orafed-arwsdl:AttributeResponseMessage"/>
                </wsdl:operation>
        </wsdl:portType>

        <wsdl:binding name="AttributeRequesterServiceBinding"
type="orafed-arwsdl:AttributeRequesterServicePortType">
                <soap:binding style="document"
transport="http://schemas.xmlsoap.org/soap/http"/>
                <wsdl:operation name="AttributeRequestOp">
                        <soap:operation
soapAction="http://www.oracle.com/fed/AttributeRequestOp" />
                        <wsdl:input>
                                <soap:body use="literal"/>
```

```
                        </wsdl:input>
                        <wsdl:output>
                                <soap:body use="literal"/>
                        </wsdl:output>
                </wsdl:operation>
        </wsdl:binding>

        <wsdl:service name="AttributeRequesterService">
                <wsdl:port name="AttributeRequesterServicePort"

binding="orafed-arwsdl:AttributeRequesterServiceBinding">
                        <soap:address
location="http://stadm04.us.oracle.com:7778/fed/ar/soap"/>
                </wsdl:port>
        </wsdl:service>
</wsdl:definitions>
```

The `types` and `message` sections define the contents of the `AttributeRequest` and `AttributeResponse` messages.

The built-in XML Scheme type ID is used for the Name attribute of the `Attribute` elements; this type approximates the desired syntax for attribute names (letters, numbers, "_", "-", and ".") However, ID (which is derived from the XML NCName type) also includes a number of Unicode combining characters and extenders.

> **See Also:** The W3C specification, Namespaces in XML, at `http://www.w3.org/TR/1999/REC-xml-names-19990114/#NT-NCName`.

The `binding` and `service` sections specify how the messages are to be sent over SOAP and HTTP(S).

References

See `http://java.sun.com/j2se/1.4.2/docs/api/java/util/regex/Pattern.html` for a summary of regular-expression constructs.

Configuring Attribute Mapping

This section explains how to configure the attribute mapping functionality in Oracle Identity Federation. It contains these topics:

- Introduction to Attribute Mapping
- Mapping Configuration
- Sample attr-config.xml File
- Examples

> **See Also:** See `http://java.sun.com/j2se/1.4.2/docs/api/java/util/regex/Pattern.html` for a summary of regular-expression constructs.

Introduction to Attribute Mapping

Oracle Identity Federation supports attribute mapping for the following:

- SAML 2.0 Attribute Authority

- SAML 2.0 Attribute Requester
- Liberty 1.x/ SAML 2.0 Identity Provider, when sending attributes in SSO assertions

Oracle Identity Federation provides the following attribute mapping capabilities:

- Attribute Name Mapping: maps local attribute names to external attribute names used in Liberty/SAML messages
- Attribute Value Mapping: maps local attribute values to external attribute values used in Liberty/SAML messages
- Attribute Value Filtering: filters local attribute values by sending only allowed values in assertion messages

> **Note:** In this release, all attribute mapping and filtering is available only as per-peer-provider configuration, not at the global level.

Attribute Name Mapping

Attribute name mapping allows the administrator to specify the name with which a local attribute should be defined in the Liberty/SAML messages when sending or receiving messages.

On the IdP/Attribute Authority side, when a mapping is defined, Oracle Identity Federation can also be configured to send the attribute to a specific peer provider. Thus, when no name mappings are defined, Oracle Identity Federation is configured to send no attributes to peer providers.

Oracle Identity Federation exercises attribute name mapping when acting as a:

- SAML 2.0 Attribute Authority
- SAML 2.0 Attribute Requester
- Liberty 1.x/SAML 2.0 Identity Provider, when sending attributes in SSO Assertions

Attribute name mapping is configured through the Oracle Identity Federation Administration Console. See "Attribute Name Mapping" on page 6-123 for details.

Attribute Value Mapping

Attribute value mapping allows the administrator to specify the value that a local attribute should be assigned in a Liberty/SAML message when sending or receiving messages.

Attribute value mapping has these characteristics:

- A value mapping consists of a combination, or duet, of a local value and the corresponding external value.
- Value mappings can be defined for any local attributes. Multiple value mappings can be defined for each local attribute.
- Different external values can be mapped to the same local value using value mappings. A default attribute is used to determine which external value will be used in outgoing mode.
- Different local values can be mapped to the same external value by means of value mappings. A default attribute is used to determine which local value to use in incoming mode when mapping external values into local values.

Oracle Identity Federation exercises attribute value mapping when acting as a:

- SAML 2.0 Attribute Authority
- SAML 2.0 Attribute Requester
- Liberty 1.x/SAML 2.0 Identity Provider, when sending attributes in SSO Assertions

Configuration involves manually editing an XML file. See "Attribute Value Mapping" on page 6-127 for more details.

Attribute Value Filtering

Attribute value filtering allows the administrator to specify which local value is allowed when sending a Liberty/SAML message.

Attribute value filtering has these characteristics:

- Filter rules can be defined for any local attributes. A filter rule evaluates each attribute value to determine if it can be sent. If the evaluation is positive, the value is sent; otherwise, it is removed from the list of attribute values to be sent.
- Multiple filter rules can be defined for each local attribute. When sending a value, Oracle Identity Federation can be set up to either:
  - send only after all filters evaluate successfully
  - send if at least one filter evaluates successfully
- The administrator defines a filtering rule by specifying the type of comparison, and the string value to compare (see "Attribute Value Filtering" on page 6-127).
- Oracle Identity Federation supports these comparison types when comparing the attribute value to a string:
  - equals
  - not equals
  - starts with
  - ends with
  - contains
  - does not contain
  - equals null
  - not equals null
- In addition to these comparison types, filtering supports regular expressions, allowing the user to match the attribute value against a regular expression. See "Filtering Conditions" in the section "Attribute Value Filtering" on page 6-127 for details.
- The filtering rules allow you to specify whether the comparison will be case-sensitive.

Oracle Identity Federation exercises attribute value filtering when acting as a:

- SAML 2.0 Attribute Authority
- Liberty 1.x/SAML 2.0 Identity Provider, when sending attributes in SSO Assertions

Configuration involves manually editing an XML file. See "Attribute Value Filtering" on page 6-127 for details.

Mapping Configuration

This section provides details about configuring the attribute mapping features of Oracle Identity Federation. It contains these topics:

- **Configuration Files**
- Server Configuration
- Mapping & Filtering Configuration

Configuration Files

Configuration information for Oracle Identity Federation attribute sharing features is stored in two different files:

- The `$ORACLE_HOME/fed/conf/cot.xml` file contains configuration data for attribute name mapping.
- The `$ORACLE_HOME/fed/conf/attr-config.xml` file contains configuration data for attribute value mapping and filtering.

cot.xml The `cot.xml` file contains the Circle Of Trust configuration information that Oracle Identity Federation uses to interact with peer providers.

This file is managed by the Oracle Identity Federation Administration Console and **must not** be manually edited.

attr-config.xml The `attr-config.xml` file was specifically created to hold configuration data for the attribute value mapping and filtering support in Oracle Identity Federation.

This file is not managed by the Oracle Identity Federation Administration Console. The administrator must edit it manually to configure attribute value mapping and filtering. For details on how to configure the Attribute Sharing framework in Oracle Identity Federation, see "Server Configuration" on page 6-122.

Note: The `$ORACLE_HOME/fed/conf/attr-config.xml` file is not present by default in the Oracle Identity Federation installation. The administrator needs to create it manually. See "XML File Structure" and "Server Configuration" below for details about the structure of this file.

The procedure for using this file depends on whether the transient data is stored in an RDBMS or in memory.

RDBMS Transient Store

When the Oracle Identity Federation transient data store is of RDBMS type (that is, user session information is stored in a database), Oracle Identity Federation uses the RDBMS to store its configuration data. Thus, in a clustered environment, Oracle Identity Federation servers using a common database will share the same configuration without the need to manually copy configuration files to propagate any changes.

All changes made through the Oracle Identity Federation Administration Console are persisted to the RDBMS, and will in turn be picked up by any Oracle Identity Federation server using the RDBMS as its configuration repository.

When manual changes are made to the `attr-config.xml` file, the data is not automatically persisted to the database. Take these steps to ensure the changes are reflected in the database:

1. Open the `$ORACLE_HOME/fed/config/attr-config.xml` file.
2. Locate the top root XML element, which is called `AttributeConfig`.
3. Set the value of the `useLocalConfig` attribute of the `AttributeConfig` XML element to true. That forces Oracle Identity Federation to persist the local `attr-config.xml` file to the database during the next restart, and the boolean flag is reset to `false`.
4. Save and exit the file.
5. Restart the OC4J_FED container of the machine where the `attr-config.xml` file was modified.

After these steps are performed, the local content of the `attr-config.xml` file are propagated to the RDBMS, and to all Oracle Identity Federation servers using this database to store their configuration information.

In-memory Transient Store

When the Oracle Identity Federation transient data store is of type memory, the configuration data is stored entirely in local files.

When performing manual changes to the `attr-config.xml` file, it is only necessary to force a refresh of the Oracle Identity Federation server to pick up the new changes.

To refresh the server:

1. Perform changes to the `attr-config.xml` file, and save the file.
2. Go to the Oracle Identity Federation Administration Console.
3. Click on the **Refresh Server** button. This forces the standalone Oracle Identity Federation server to reload its configuration based on the files.

XML File Structure

To set up attribute value mapping, the administrator manually edits the `$ORACLE_HOME/fed/conf/attr-config.xml` file. This XML file has a generic structure, as it defines the configuration information for attribute value mapping and attribute value filtering on a per-provider basis.

> **Note:** See "Sample attr-config.xml File" below for an example of the `attr-config.xml` file.

The structure of the `attr-config.xml` file is as follows:

`AttributeConfig` is the top XML root element. It contains:

- `useLocalConfig` attribute: Boolean to indicate whether the local file's content should overwrite the RDBMS configuration entry when Oracle Identity Federation uses RDBMS as its transient data store. See "RDBMS Transient Store" above for more information on this attribute.

- `PeerProvider` child elements: These XML elements appear as children of `AttributeConfig` and represent configuration information for a specific provider.

`PeerProvider` XML element represents the attribute value mapping and filtering for a specific peer provider. It contains:

- `providerID` attribute: Defines the identifier used to reference the Peer Provider. This value must equal the value used in the Circle Of Trust to reference the Peer Provider.
- `PropertyGroup` child elements: Represent a list of specific configuration entries. These are the only children of the `PeerProvider` XML element.

`PropertyGroup` XML element is a list of specific configuration entries, such as configuration data for attribute value mapping, and configuration information for attribute value filtering. It contains:

- `name` attribute: The name of the `PropertyGroup` XML element configuration group.
- `PropertyGroupItem` child elements: Represent configuration entries in the list defined by the `PropertyGroup` parent. These are the only children of the `PropertyGroup` XML element.

`PropertyGroupItem` XML element: Represents a configuration entry in the list defined by the `PropertyGroup` parent. For example, when the `PropertyGroup` parent represents configuration for attribute value filtering, the `PropertyGroupItem` XML element will contain filtering configuration information for one specific local attribute. It contains:

- `Property` child elements: Simple name-value pair properties attached to the `PropertyGroupItem` parent.
- `PropertiesList` child elements: Represent groups of name-value pair properties.

`Property` XML element: A simple name-value pair property. It contains:

- `name` attribute: The name of the property
- XML text child: The value of the property

`PropertiesList` XML element: Represents multiple groups of name-value pair properties. It contains:

- `name` attribute: The name of the `PropertiesList` element.
- `PropertiesListItem` child elements: Each child represents a group of name-value pair properties.

`PropertiesListItem` XML element: A group of name-value pair properties. For example, when the `PropertyGroup` parent represents configuration for attribute value filtering, the `PropertyGroupItem` XML element will contain filtering configuration information for one specific local attribute, the `PropertiesList` XML element will contain the mapping configuration information for that specific attribute, and a `PropertiesListItem` XML element will contain information for one mapping for that specific attribute.

The element consists of:

- Different XML attributes, based on the context. See "Attribute Value Mapping" on page 6-124 and "Attribute Value Filtering" on page 6-127 for a list of the defined attributes.

- `Property` child elements: Simple name-value pair properties.

Server Configuration

We now describe how to configure Oracle Identity Federation to:

- act as an Attribute Authority
- act as an Attribute Requester
- send attributes in SSO Assertions

Configuring Oracle Identity Federation as Attribute Authority Take these steps to configure Oracle Identity Federation to act as an attribute authority:

1. Go to the Oracle Identity Federation Administration Console.
2. Go to Server Configuration -> Identity Provider -> SAML 2.0.
3. Check the **Attribute Responder Enabled** box.
4. Save and refresh the server.

Checking the **Attribute Responder Enabled** box enables the attribute authority feature. It also modifies the IdP's metadata to include information about the attribute authority service. Note that the metadata at the peer providers' sites must be updated with the new version.

After enabling the attribute responder capability, you must configure:

- which attributes to send
- the attribute name mapping
- the attribute value mappings
- the attribute value filters

See"Mapping & Filtering Configuration" on page 6-123 for more information.

Configuring Oracle Identity Federation as Attribute Requester Take these steps to configure Oracle Identity Federation to act as an attribute requester:

1. Go to the Oracle Identity Federation Administration Console.
2. Go to Server Configuration -> Service Provider -> SAML 2.0.
3. Check the **Attribute Requester Enabled** box.
4. Save and refresh the server.

When Oracle Identity Federation acting as an attribute requester receives a request that will trigger an attribute exchange flow (by means of an attribute query from a Service Provider to attribute authority, and assertion from attribute authority to SP), the request will contain the DN identifying the user for which the attributes need to be retrieved.

Oracle Identity Federation can query a specific attribute authority based on the user's DN. Take these steps to configure this feature:

1. Go to the Oracle Identity Federation Administration Console.
2. Go to Server Configuration -> Service Provider -> Attribute Requester.
3. Enter a sub-DN, and select the attribute authority where a request will be sent with a DN matching the sub-DN.
4. Save and refresh the server.

See "Service Provider - Attribute Requester" on page 6-46 for more details.

If no matches can be found for the user's DN, the Default SSO IdP is used.

To configure the default SSO IdP:

1. Go to the Oracle Identity Federation Administration Console.
2. Go to Server Configuration -> Service Provider.
3. Select the Default SSO IdP from the drop down list.
4. Save and refresh the server.

After enabling the attribute requester capabilities and setting up the Default SSO IdP and/or the DN Mappings, you must configure the attribute name mapping and the attribute value mappings. See "Mapping & Filtering Configuration" on page 6-123 for more information.

Sending Attributes in SSO Assertion During a Single Sign-On operation, the IdP can optionally include attributes in the authentication assertion to be consumed by the Service Provider.

Take these steps to enable attributes to be sent in an assertion:

1. Go to the Oracle Identity Federation Administration Console.
2. Go to Server Configuration -> Circle Of Trust.
3. Select the peer provider with which you want to configure attribute sharing, and click the **Update** button.
4. Check the **Enable Attributes in SSO** box.
5. Click **Apply**.

After checking the **Enable Attributes in SSO** box, you need to configure

- the attributes to send
- the attribute name mapping
- the attribute value mappings
- the attribute value filters

See "Mapping & Filtering Configuration" on page 6-123 for more information.

Mapping & Filtering Configuration

This section explains how to configure mapping and filtering.

Attribute Name Mapping Attribute name mapping is configured at the Circle Of Trust page of the Oracle Identity Federation Administration Console. This configuration serves two purposes:

- to map attributes names contained in an assertion to local attribute names on the IdP
- to define which local attributes can be sent to the peer provider. This means that defining an attribute name mapping for a peer provider will authorize Oracle Identity Federation to send this attribute to the remote server.

Take these steps to define an attribute name mapping, or to enable an attribute to be sent, on the IdP side:

1. Go to the Oracle Identity Federation Administration Console.

2. Go to Server Configuration -> Circle Of Trust.
3. Select the peer provider with which you want to configure attribute sharing, and click the **Update** button.
4. Click the **Attribute Mappings** button.

   Note: Note [1] no longer applies to the field name.

   The first table contains the attribute name mappings configuration, with the following fields for each attribute definition:

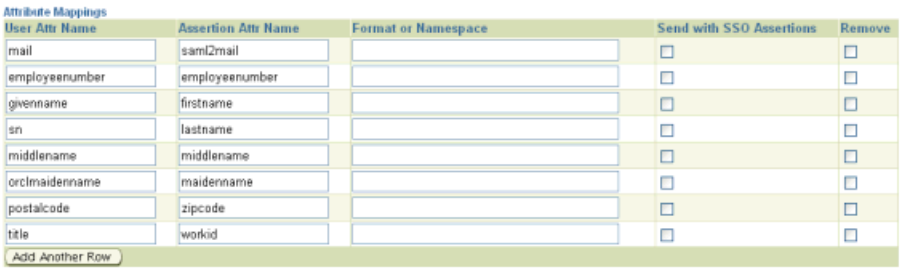
   - User Attr Name: The name of the local attribute in the user repository
   - Assertion Attr Name: The name that will be used to identify the attribute in the Assertion
   - Format or Namespace: An optional field used to specify the format of the namespace of the SAML attribute, depending on the version.
     - For SAML 1.x/Liberty 1.x, this field's value is used to set the SAML attribute's namespace.
     - For SAML 2.0, this value is used to set the SAML attribute's NameFormat; if this field is empty, the NameFormat of the SAML attribute will be set to `urn:oasis:names:tc:SAML:2.0:attrname-format:basic`; otherwise the NameFormat will hold the value specified in this field.
   - Send with SSO Assertions: Indicates whether the attribute should be sent during an SSO operation, along with the authentication assertion.
5. After defining the necessary attributes, click the **Apply** button.

   Note: if you want to send any attributes with the SSO assertions, define the name formats for which attributes will be sent in the second table of the Attribute Mappings page. Also check the **Enable Attributes in SSO** box at the Edit Trusted Provider page.

6. To make the changes effective for use by the server, click the **Refresh Server** button after saving all the data.

Here is an example of an Attribute Mappings configuration page:

Figure 6–1 Configuring Attribute Mappings



Attribute Value Mapping This section describes how to configure attribute value mapping.

Note: See "XML File Structure" above for a description of the `attr-config.xml` file. An understanding of the file structure is necessary to follow the subsequent discussion.

PeerProvider

Attribute value mapping is configured on a per-provider basis, which means that a specific mapping will apply only to one provider. The configuration for attribute value mapping will be stored in a `PeerProvider` element:

```
<PeerProvider providerID="idpID">
…  <!-- contains configuration for provider idpID -->
</PeerProvider>
```

PropertyGroup

The attribute value mapping configuration is contained in a `PropertyGroup` XML element whose name attribute is equal to `attr-value-mappings`. This element is defined under `PeerProvider`:

```
<PeerProvider providerID="idpID">
   <PropertyGroup name="attr-value-mappings">
   <!-- contains value mapping configuration for provider idpID -->
   </PropertyGroup>
</PeerProvider>
```

PropertyGroupItem

Each `PropertyGroupItem` defined under the `PropertyGroup` element represents mapping information for a local attribute:

```
<PropertyGroup name="attr-value-mappings">
   <PropertyGroupItem>
      …    <!-- contains mapping configuration for mail attribute -->
   </PropertyGroupItem>
   <PropertyGroupItem>
      …    <!-- contains mapping configuration for title attribute -->
   </PropertyGroupItem>
…
</PropertyGroup>
```

`PropertyGroupItem` contains two kinds of parameters:

- simple name-value pair properties contained in `Property` elements
- multiple groups of name-value pair properties contained in `PropertiesList`

Properties of PropertyGroupItem

The possible name-value pair properties that can be defined under the `PropertyGroupItem` for a specific attribute are:

- `attr-name`, which defines the name of the local attribute for which the mapping configuration will apply:

  ```
  <Property name="attr-name">title</Property>
  ```

- `send-unmapped-values`, which indicates whether Oracle Identity Federation will allow unmapped values to be sent out (`true` or `false`):

  ```
  <Property name="send-unmapped-values">true</Property>
  ```

- `receive-unmapped-values`, which indicates whether Oracle Identity Federation will allow receipt of unmapped values (`true` or `false`):

```
<Property name="receive-unmapped-values">false</Property>
```

PropertiesList

The `PropertiesList` element with the attribute name of `mappings` contains multiple groups of name-value pair properties. Each group of name-value pair properties defines a value mapping rule for the attribute defined in the `PropertyGroupItem` element.

A group of name-value pair properties is defined by a `PropertiesListItem` element.

```
<PropertiesList name="mappings">
   <PropertiesListItem>
      …  <!-- mapping rule for attribute title, for value local
       value "Senior Member of Technical Staff" and external value "smts" -->
   </PropertiesListItem>
   <PropertiesListItem>
      …  <!-- mapping rule for attribute title, for value local value
      "Principal Member of Technical Staff" and external value "pmts" -->
   </PropertiesListItem>
</PropertiesList>
```

A `PropertiesListItem` element can be made up of XML attributes and `Property` children.

Properties of PropertiesListItem

The possible name-value pair properties that can be defined under the `PropertiesListItem` for a value mapping rule are:

- `local-value`, which holds the local value for this mapping rule.

```
<Property name="local-value">Senior Member of Technical Staff
</Property>
```

- `external-value`, which holds the external mapped value for this mapping rule.

```
<Property name="external-value">smts
</Property>
```

Attributes of PropertiesListItem

The possible parameters that can be defined as attributes of the `PropertesListItem` XML element are:

- `ignoreCase`, which indicates whether the string comparison will be case-sensitive when matching attribute values (`true` or `false`):

```
<PropertiesListItem ignoreCase="true">
```

- `isLocalNull`, which indicates that the local value equals a null string (different from an empty string ""):

```
<PropertiesListItem default="true" isLocalNull="true">
<Property name="external-value">foo</Property>
</PropertiesListItem>
```

- `isExternalNull`, which indicates that the external value equals a null string:

```
<PropertiesListItem default="true" isExternalNull="true">
<Property name="local-value">foo</Property>
</PropertiesListItem>
```

- `default`, which indicates whether the local value will be used as the local value in case an incoming external value can be mapped to several local values. Note: There can only be one `PropertiesListItem` element defined with the default value set to `true` in a `PropertiesList` group (`true` or `false`).

```
<PropertiesListItem ignoreCase="true" default="true">
```

Attribute Value Filtering This section describes how attribute value filtering is applied in Oracle Identity Federation.

> **Note:** The structure of the `attr-config.xml` file was explained in "XML File Structure" above. An understanding of this file is necessary in order to follow the subsequent discussion.

PeerProvider

Attribute value filtering is configured on a per-provider basis, which means that a specific filter will only apply to one provider. So the configuration for attribute value filtering will be stored in a `PeerProvider` element:

```
<PeerProvider providerID="spID">
…  <!-- contains configuration for provider spID -->
</PeerProvider>
```

PropertyGroup

The attribute value filtering configuration information is contained in a `PropertyGroup` XML element whose name attribute is equal to `attr-value-filters`. This element is defined under `PeerProvider`:

```
<PeerProvider providerID="spID">
   <PropertyGroup name="attr-value-filters">
   …  <!-- contains value filtering configuration for provider spID -->
   </PropertyGroup>
</PeerProvider>
```

PropertyGroupItem

Each `PropertyGroupItem` defined under the `PropertyGroup` element represents filtering information for a local attribute:

```
<PropertyGroup name="attr-value-filters">
   <PropertyGroupItem>
      …    <!-- contains filtering configuration for mail attribute -->
   </PropertyGroupItem>
   <PropertyGroupItem>
      …    <!-- contains filtering configuration for title attribute -->
   </PropertyGroupItem>
…
</PropertyGroup>
```

The `PropertyGroupItem` contains two kinds of parameters:

- simple name-value pair properties contained in `Property` elements
- multiple groups of name-value pair properties contained in `PropertiesList`

Properties of PropertyGroupItem

The possible name-value pair properties that can be defined under the `PropertyGroupItem` for a specific attribute are:

- attr-name, which defines the name of the local attribute for which the filtering configuration will apply:

```
<Property name="attr-name">title</Property>
```

- condition-operator, which indicates whether all conditions need to be met for an attribute to be sent (and operator), or if only one filter is enough to send an attribute (or operator):

```
<Property name="condition-operator">and</Property>
```

PropertiesList

The PropertiesList element with the attribute name of filters contains multiple groups of name-value pair properties. Each group of name-value pair properties defines a value filtering rule for the attribute defined in the PropertyGroupItem element.

A group of name-value pair properties is defined by a PropertiesListItem element.

```
<PropertiesList name="filters">
   <PropertiesListItem>
   …  <!-- filtering rule for attribute title, for condition "not-equals" and
expression "Manager" -->
   </PropertiesListItem>
   <PropertiesListItem>
   …  <!-- filtering rule for attribute title, for condition "not-equals" and
expression "Director" -->
   </PropertiesListItem>
</PropertiesList>
```

A PropertiesListItem element can be made up of XML attributes and Property children.

Properties of PropertiesListItem

The possible name-value pair properties that can be defined under the PropertiesListItem for a value filtering rule are:

- condition, which holds the type of condition of the filtering rule.

```
<Property name="condition">not-equals
</Property>
```

- expression, which holds the expression that will be used to evaluate the condition with the outgoing attribute value.

```
<Property name="expression">Manager
</Property>
```

Attributes of PropertiesListItem

The following parameter can be defined as an attribute of the PropertiesListItem XML element:

- ignoreCase, which indicates whether the string comparison will be case sensitive when matching attribute values.

```
<PropertiesListItem ignoreCase="true">
```

Filtering Conditions

Oracle Identity Federation provides several filtering conditions (to be used in elements defined in "Properties of PropertiesListItem" above):

- `equals`: the filtering rule will return `true` if the expression value is equal to the outgoing attribute value.
- `not-equals`: the filtering rule will return `true` if the expression value is different from the outgoing attribute value.
- `startswith`: the filtering rule will return `true` if the outgoing attribute value begins with the expression value.
- `endswith`: the filtering rule will return `true` if the outgoing attribute value ends with the expression value.
- `contains`: the filtering rule will return `true` if the outgoing attribute value contains the expression value.
- `not-contains`: the filtering rule will return `true` if the outgoing attribute value does not contain the expression value.
- `equals-null`: the filtering rule will return `true` if the outgoing attribute value is null.
- `not-equals-null`: the filtering rule will return `true` if the outgoing attribute value is not null.
- `regexp`: the filtering rule will return `true` if the outgoing attribute value matches the regular expression, which is defined in the expression value.

Note: The rules are used to determine the *allowed* values. Consequently, if a rule evaluates to `true`, this means that it is permissible to send the value.

When the filtering rule is set to `regexp`, the expression value must be a standard Unix regular expression. See `http://java.sun.com/j2se/1.4.2/docs/api/java/util/regex/Pattern.html` for details about regular expression constructs.

Note: The `ignoreCase` flag is disregarded during attribute value processing because regular expressions already support case-insensitivity.

Table 6–9 contains some examples illustrating the use of the `regexp` filtering condition:

Table 6–9 Using the regexp Filtering Condition

Regular Expression	Meaning
.*rector	any string which ends with "rector"
[^abc]	any character except a, b, or c (negation)
user\d	user0, user1, ..., user9
a*b	any string which begins with 0+ "a" characters and ends with a letter b (for example, aaaaab)

Sample attr-config.xml File

A sample `attr-config.xml` file is provided here for reference.

```
<AttributeConfig useLocalConfig="false">
   <PeerProvider providerID="http://stadm14.us.oracle.com:7779/fed/idp">
      <PropertyGroup name="attr-value-filters">
         <PropertyGroupItem>
            <Property name="attr-name">title</Property>
            <Property name="condition-operator">and</Property>
            <PropertiesList name="filters">
               <PropertiesListItem ignoreCase="true">
                  <Property name="condition">not-equals</Property>
                  <Property name="expression">Manager</Property>
               </PropertiesListItem>
               <PropertiesListItem ignoreCase="true">
                  <Property name="condition">not-equals</Property>
                  <Property name="expression">Director</Property>
               </PropertiesListItem>
            </PropertiesList>
         </PropertyGroupItem>
      </PropertyGroup>
      <PropertyGroup name="attr-value-mappings">
         <PropertyGroupItem>
            <Property name="attr-name">title</Property>
            <Property name="send-unmapped-values">true</Property>
            <Property name="receive-unmapped-values">true</Property>
            <PropertiesList name="mappings">
               <PropertiesListItem ignoreCase="true">
                  <Property name="local-value">Senior Member of
                  Technical Staff</Property>
                  <Property name="external-value">smts</Property>
               </PropertiesListItem>
               <PropertiesListItem ignoreCase="true">
                  <Property name="local-value">Principal Member of
                  Technical Staff</Property>
                  <Property name="external-value">pmts</Property>
               </PropertiesListItem>
               <PropertiesListItem ignoreCase="true">
                  <Property name="local-value">Consulting Member of
                  Technical Staff</Property>
                  <Property name="external-value">cmts</Property>
               </PropertiesListItem>
            </PropertiesList>
         </PropertyGroupItem>
      </PropertyGroup>
   </PeerProvider>
</AttributeConfig>
```

Examples

This section provides some examples showing how value mapping/filtering rules apply to SAML 2.0 outgoing attributes.

Example 1

Attribute Name = "title"

Old Attribute Values = ["Consulting Member of Technical Staff", "PRINCIPAL MEMBER OF TECHNICAL STAFF", "Principal Member of Technical Staff", "Vice President"]

New Attribute Values = ["cmts", "pmts", "Vice President"]

Note:

1. Since we defined value mappings to be case-insensitive, both "PRINCIPAL MEMBER OF TECHNICAL STAFF" and "Principal Member of Technical Staff" get mapped to "pmts".
2. Since `"send-unmapped-values = true"` and there is no rule defined for value "Vice President", we map it to itself.
3. No filtering rules apply.

Example 2

Attribute Name = "title"

Old Attribute Values = ["Manager", "Consulting Member of Technical Staff"]

New Attribute Values = ["cmts"]

Note: "Manager" is one of the forbidden values, so it never gets sent.

Example 3

The old filtering rules were:

```
<PropertyGroupItem>
   <Property name="attr-name">title</Property>
   <Property name="condition-operator">and</Property>
   <PropertiesList name="filters">
      <PropertiesListItem ignoreCase="true">
         <Property name="condition">not-equals</Property>
         <Property name="expression">Manager</Property>
      </PropertiesListItem>
      <PropertiesListItem ignoreCase="true">
         <Property name="condition">not-equals</Property>
         <Property name="expression">Director</Property>
      </PropertiesListItem>
   </PropertiesList>
</PropertyGroupItem>
```

The interpretation is that we allow any attribute value which is not equal to "Manager" and not equal to "Director".

Now suppose we change the filtering rules (changed items are in **bold**):

```
<PropertyGroupItem>
   <Property name="attr-name">title</Property>
   <Property name="condition-operator">or</Property>
   <PropertiesList name="filters">
      <PropertiesListItem ignoreCase="true">
         <Property name="condition"> equals</Property>
         <Property name="expression">Manager</Property>
      </PropertiesListItem>
      <PropertiesListItem ignoreCase="true">
         <Property name="condition"> regexp</Property>
         <Property name="expression">.*rector</Property>
      </PropertiesListItem>
   </PropertiesList>
</PropertyGroupItem>
```

The interpretation changes, in that we now only allow an attribute value which is either "Manager" or ends in "rector". Thus, we get:

Attribute Name = "title"

Old Attribute Values = ["Manager", "Consulting Member of Technical Staff"]

New Attribute Values = ["Manager"]

Configuring the Logout Service

Oracle Identity Federation provides a logout service that can be invoked to log the user out from all peer providers and from the Oracle Identity Federation, OracleAS Single Sign-On, Oracle Access Manager, and eTrust SiteMinder domains (clearing cookies for the last two servers).

The logout flow is invoked when logging out of the OracleAS Single Sign-On domain, but it may not be triggered when logging out from Oracle Access Manager or eTrust SiteMinder.

Administrators can use this service to trigger the logout flow from Oracle Identity Federation or use it as a link in a portal.

The logout process can be launched by accessing a URL of the form:

```
http://hostname:port/fed/user/logout?returnurl=http://anotherhos
tname/path
```

The logout service takes a required `returnurl` parameter, which is necessary for correct operation; the user will be redirected to this URL after the logout process completes.

If no `returnurl` parameter is specified when invoking the Oracle Identity Federation logout URL, the sign-off operation is performed, but the server does not display a result page; instead, it displays the last page visited by the browser, and the operation appears to have aborted even though logout was successful. To avoid the problem, specify the `returnurl` parameter to point to the result page.

Note: The logout is performed for all peer providers for the following protocols:

- Liberty 1.1
- Liberty 1.2
- SAML 2.0
- WS-Federation

WS-Federation Logout

WS-Federation can be used to sign into one or more service providers using an identity provider that performs the actual authentication.

The logout mechanism and URLs depend on whether the action is initiated at the IdP or the SP.

Note: The IdP and SP logout URLs listed here are for use by remote WS-Federation providers when these providers need to set up their implementation for logout.

IdP-initiated Logout

The user clicks on a link at the IdP site that initiates a WS-Federation signout. This URL is:

```
http(s)://IP-HOST:IP-PORT/shareid/wsfederation/
ObWSFedIdentitySTS?wa=wsignout1.0
```

The Oracle Identity Federation IdP keeps track of each service provider that the user signed into. The server constructs and returns to the user's browser an HTML signout page, with iframes for each SP. Each iframe issues a request to the target SP for the WS-Federation signout cleanup:

```
http(s)://RP-HOST:RP-PORT/shareid/wsfederation/
ObWSFedResourceSTS?wa=wsignoutcleanup1.0
```

Each SP processes the signout cleanup to signout the SP session created for WS-Federation, and returns an appropriate logout message to be displayed in the iframe in the IdP's signout page.

When Oracle Identity Federation is the SP, it performs an internal redirection back to the `ObWSFedResourceSTS` servlet, and the servlet returns the following logout message:

```
Successful logout for Realm Resource Realm URI
```

This message and its HTML declaration is defined in the `shareid-messages.properties` file, and can be translated and customized.

To sign out the IdP session, the Oracle Identity Federation IdP adds an iframe with a signout cleanup request for the IP host and port. The logout page will display logout messages for each SP and for the IdP.

SP-initiated Logout

The user clicks on a link at an SP site that initiates the WS-Federation signout. This URL is:

```
http(s)://RP-HOST:RP-PORT/shareid/wsfederation/
ObWSFedResourceSTS?wa=wsignout1.0
```

The SP records the IP address that was used to sign in, and redirects the user's browser to the IdP's signout URL to perform the signout. From this point on, the signout is the same as for the IP-initiated signout.

Using SSL with Oracle Identity Federation

This section explains how to set up SSL connections between Oracle Identity Federation servers and peer providers. It contains these topics:

- Connecting to SSL Servers
- Authenticating to SSL Servers
- Configuring SSL Server on Oracle Identity Federation
- Requiring a Client SSL Certificate for SOAP Requests

To configure SSL on the Oracle Application Server where Oracle Identity Federation is running, refer to:

Oracle HTTP Server Administrator's Guide

Connecting to SSL Servers

To configure Oracle Application Server to allow Oracle Identity Federation to make SSL connections to remote Liberty 1.x/SAML providers, you will need to import the trusted CA certificates in the JVM's keystore. To do so, use the keytool application located in the `$ORACLE_HOME/jdk/bin` directory to add trusted certificates to the keystore in the `$ORACLE_HOME/fed/shareid/oblix/config/keystore` file.

Restart Oracle Application Server for the changes to take effect.

Note: If you had previously inserted SSL certificates into the `$ORACLE_HOME/jdk/jre/lib/security/cacerts` file, you will need to re-import them to `$ORACLE_HOME/fed/shareid/oblix/config/keystore`.

Authenticating to SSL Servers

Some SSL Servers might require authentication of the client performed during the SSL handshake. This operation is typically done by having the SSL Client presenting an SSL Client Certificate to the SSL Server.

This section describes how to set up SSL with client certificate authentication. In brief, the steps are:

1. Set up trust for the CA that issued the IdP SSL server certificates.
2. Obtain a certificate for the Oracle Identity Federation SSL client.

The detailed step-by-step procedure follows.

In the syntax shown in these steps, KEYTOOL, KEYSTORE, and STOREPASS represent these values:

```
KEYTOOL=ORACLE_HOME/jdk/bin/keytool
KEYSTORE=ORACLE_HOME/fed/shareid/oblix/config/keystore
STOREPASS=ias_admin password
```

Take these steps to configure certificate authentication on the server where the SSL client resides:

1. Set up trust for the CA(s) that issued the SSL server certificates for the remote SSL Server(s) to which Oracle Identity Federation will connect:
   - Obtain the PEM-formatted certificates for each CA that issued a certificate for the SSL servers.
   - Import each CA certificate into the shareid keystore.

     ```
     KEYTOOL -keystore ORACLE_HOME/fed/shareid/oblix/config/keystore
        -storepass OIF_ADMIN_PASSWORD -import -alias CA-ALIAS
        -file CA-CERT.pem
     ```

   - Restart OC4J_FED.
2. For client certificate authentication, get a certificate for the Oracle Identity Federation SSL client.

- Generate a certificate request for the existing key:

```
KEYTOOL -keystore KEYSTORE -storepass STOREPASS
   -certreq -alias shareid -file CLIENT-REQ.pem
```

- Submit the certificate request in `client-req.pem` to your CA. Request a certificate without a `keyUsage` extension.
- Obtain the certificate from the CA, and also the CA's certificate, in `PEM` format. Paste the certificates into files and ensure there are no extraneous newlines in the files.
- Import the CA certificate into the keystore as a trusted certificate:

```
KEYTOOL -keystore KEYSTORE -storepass STOREPASS
   -import -alias your-CA -file CA-CERT.pem

Trust this certificate? yes
```

- Import the certificate into the keystore:

```
KEYTOOL -keystore KEYSTORE -storepass STOREPASS
   -import -alias shareid -file CLIENT-CERT.pem
```

Note: If you see the exception "`java.security.cert.CertificateException: DerInputStream.getLength(): lengthTag=127, too big`", there is an extraneous newline in the certificate file after the last line. Delete it and try again.

- Restart OC4J_FED:

```
ORACLE_HOME/opmn/bin/opmnctl stopproc ias-component=OC4J_FED
ORACLE_HOME/opmn/bin/opmnctl startproc ias-component=OC4J_FED
```

Configuring SSL Server on Oracle Identity Federation

This section describes how to set up SSL with optional client certificate authentication. In brief, the steps are:

1. Create a wallet with the HTTP server key and certificate.
2. Enable SSL on the HTTP server.
3. Optionally configure an additional SSL port on the HTTP server with client certificate authentication enabled.

The detailed step-by-step procedure follows. Take these steps to configure certificate authentication on the Oracle Identity Federation site where the SSL server resides:

1. Create a wallet with the HTTP server key and certificate.
   - Set the `ORACLE_HOME` environment variable, if it is not already set.
   - Run the Oracle Wallet Manager:

     Unix: `$ORACLE_HOME/bin/owm`

     Windows: Start - > All Programs - > Oracle - ORACLE_HOME_NAME - > Integrated Management Tools - > Wallet Manager

- Create a new wallet:
  - In the Manager window, select **Wallet** - > **New**.
  - Answer **Yes** to "Your default wallet does not exist".
  - Answer **Yes** to "Cannot create system default wallet directory... continue anyway?"
  - In the **New Wallet** window, enter a password for the wallet. Use the default Standard Wallet Type. Click **OK**.
  - Answer **Yes** to "Do you want to create a certificate request at this time?"
  - Fill in the **Create Certificate Request** window with the fields you want in the server's certificate DN. Note that the Common Name *must* be the server's DNS name. Click **OK**.
  - Click **OK** on the "A certificate request has been created" confirmation.
- Save the wallet as the default wallet for Oracle HTTP Server.
  - Select **Wallet** - > **Save As**.
  - In the **Select Directory** window, enter `ORACLE_HOME/Apache/Apache/conf/ssl.wlt/default`.
  - Click **OK** twice.
  - Answer **Yes** in response to "A wallet already exists in the selected location."
- Obtain a certificate from your CA and import it into the wallet.
  - In the **Wallet Manager** window, left-click on **Certificate:[Requested]**.
  - Submit the certificate request shown in the **Certificate Request** panel to your CA. Request an SSL server certificate, for OpenSSL.

    ```
    - keyUsage =
    digitalSignature,nonRepudiation,keyEncipherment,dataEncipherment

    - nsCertType = server
    ```
  - Obtain the base64 format certificate from the CA, and obtain the base64 format certificate for the CA.
  - In the **Wallet Manager** window, right-click on **Trusted Certificate** and select **Import Trusted Certificate**.
  - In the **Import Trusted Certificate** window, select **Paste Certificate**. Click **OK** and paste the CA certificate in the text box. Click **OK**. The CA certificate appears in the list of Trusted Certificates in the **Wallet Manager** window.
  - In the **Wallet Manager** window, right-click on **Certificate:[Requested]** and select **Import User Certificate**.
  - In the **Import Certificate** window, select **Paste Certificate**. Click **OK** and paste the certificate in the text box. In the **Wallet Manager** window, the certificate status turns to **Certificate:[Ready]**.
- Select **Wallet->Save**.
- Select **Wallet->Exit**.
- Also save the wallet to `ORACLE_HOME/opmn/conf/ssl.wlt/default`.

2. Enable SSL on Oracle HTTP Server.

- Edit ORACLE_HOME/opmn/conf/opmn.xml. In the element shown here, change "ssl-disabled" to "ssl-enabled".

```
<ias-component id="HTTP_Server">
   <process-type id="HTTP_Server" module-id="OHS">
      <module-data>
         <category id="start-parameters">
            <data id="start-mode" value="ssl-enabled/>
         </category>
      </module-data>
      <process-set id="HTTP_Server" numprocs="1"/>
   </process-type>
</ias-component>
```

  Save the edited file.

- Update the Distributed Cluster Management database using this command:

```
ORACLE_HOME/dcm/bin/dcmctl updateconfig -ct opmn
```

- Reload OPMN using this command:

```
ORACLE_HOME/opmn/bin/opmnctl reload
```

Note: If you see the "globalInitNLS: NLS boot file not found or invalid" error, unset any environment variables with names starting in ORA_NLS.

- Restart Oracle HTTP Server.

```
ORACLE_HOME/opmn/bin/opmnctl stopproc
ias-component=HTTP_Server
```

```
ORACLE_HOME/opmn/bin/opmnctl startproc
ias-component=HTTP_Server
```

Note: You *must* use opmnctl to restart Oracle HTTP Server; do not restart from the Enterprise Manager console.

- Use the Enterprise Manager console to verify that the SSL port is active. Select the **Ports** tab from the EM home page. You should see an entry in the ports table for the Oracle HTTP Server component with type **Listen (SSL)** and the port number. (The default port number for the first install on the host is 4443, for the second install 4444, and so on.)

- Use a browser to verify the SSL port is active.

```
https://hostname:ssl-port/
```

Note: If you see an alert indicating that the server certificate does not match the server name, check the certificate subject. If the subject is GET A REAL CERTIFICATE, then Oracle HTTP Server is still using the original dummy certificate. Try restarting the server again.

---

3. For client certificate authentication, configure an additional SSL port on Oracle HTTP Server with client certificate authentication enabled.

   - Edit the ORACLE_HOME/Apache/Apache/conf/ssl.conf file.

- Copy the existing `Listen` and `VirtualHost` definitions and paste them at the end of the file, before the closing `</IfDefine>`.
- Change the port in the copied definitions to an unused port for SSL client certificate authentication.
- In the copied `VirtualHost` definition, uncommment `SSLVerifyClient require`. This directs Oracle HTTP Server to perform the certificate authentication with the client.
- In the copied `VirtualHost` definition, replace the commented `SSLOptions` with:

  ```
  SSLOptions +ExportCertData +CompatEnvVars
  ```

  This directs Oracle HTTP Server to pass the client certificate on to the mod_oc4j module.

  Here is an example (comments omitted):

  ```
  Listen CLIENT-CERT-PORT
  <VirtualHost _default_:CLIENT-CERT-PORT>
  DocumentRoot "ORACLE_HOME/Apache/Apache/htdocs"
  ServerName HOST-DNS-NAME
  ServerAdmin you@your.address
  ErrorLog "|ORACLE_HOME/Apache/Apache/bin/rotatelogs
     ORACLE_HOME/Apache/Apache/logs/error_log 43200"
  TransferLog "|ORACLE_HOME/Apache/Apache/bin/rotatelogs
     ORACLE_HOME/Apache/Apache/logs/access_log 43200"
  Port CLIENT-CERT-PORT
  SSLEngine on
  SSLCipherSuite ALL:!ADH:!EXPORT56:+HIGH:+MEDIUM:+LOW:+SSLv2:+EX
     SSLWallet file:ORACLE_HOME/Apache/Apache/conf/ssl.wlt/default
     SSLVerifyClient require
     SSLOptions +ExportCertData +CompatEnvVars
     . . .
  </VirtualHost>
  ```

- Save the `ssl.conf` file.

Notes:

When setting up the SSL endpoint with client certificate, the Administrator can only specify the Certificate Authority from which Oracle Identity Federation/Oracle HTTP Server will accept SSL Certificates when requiring the Client Certificate: all the certificates issued by the CA will be accepted.

To be able to do some fine-grained filtering of SSL certificates (that is, only a few certificates out of the whole issued by the CA), the Oracle Identity Federation Administrator can build a filter based on the certificate's subject and issuer names. For example, in the `$Oracle_Home/Apache/Apache/conf/ssl.conf` file, in the VirtualHost element defining the SSL endpoint, one could add an `SSLRequire` command defined in a Location element for the `/fed` relative URL:

```
<Location /fed>
   SSLRequire (%{SSL_CLIENT_S_DN_CN} eq "john doe")
</Location>
```

This statement will only accept certificate with a Subject's Common Name equals "john doe".

Other filters are available. Please check the `mod_ssl SSLRequire` command in the *Oracle HTTP Server Administrator's Guide*.

- Edit the `ORACLE_HOME/Apache/Apache/conf/mod_oc4j.conf` file.
  - Add `Oc4jExtractSSL On` to the end of the file, before the `</IfModule>` tag. This directs mod_oc4j in Oracle HTTP Server to pass the client certificate on to OC4J in the `javax.servlet.request.X509Certificate` attribute, as expected by the `ObSAMLResponderService` servlet, as well as the Oracle Identity Federation Server when the `soaprequiresslcert` property is set to `true`, indicating that the Liberty 1.x/SAML 2.0 SOAP stack will require an SSL Client certificate when processing an incoming SOAP connection from a remote provider..
  - Save the `mod_oc4j.conf` file.
- Restart Oracle HTTP Server.

```
ORACLE_HOME/opmn/bin/opmnctl stopproc
ias-component=HTTP_Server
```

```
ORACLE_HOME/opmn/bin/opmnctl startproc
ias-component=HTTP_Server
```

Note: You *must* use `opmnctl` to restart Oracle HTTP Server; do not restart from the Enterprise Manager console.

- If you have a browser with one or more client certificates issued by your CA, use that browser to verify that the `client-cert-port` is active.

```
https://hostname:client-cert-port/
```

Requiring a Client SSL Certificate for SOAP Requests

Oracle Identity Federation version 10.1.4.2 introduces a new configuration parameter, `requireSSLCert`, applicable only to Liberty 1.x and SAML 2.0 protocols, which allows the Oracle Identity Federation administrator to require a client SSL certificate for all SOAP requests. This protects the SOAP port, if so desired, by setting up an SSL port with SSL client certificate turned on, and instructing Oracle Identity Federation to check if an SSL Client certificate was sent.

When a requester hits the SOAP URL using any opened Oracle HTTP Server ports, Oracle Identity Federation will make sure that the requester presented a certificate; an unknown requester using a non-SSL port or an SSL port without SSL client certificate will be denied access.

Take these steps to configure Oracle Identity Federation to require an SSL client certificate on the SOAP endpoint:

1. Perform the steps described in "Configuring SSL Server on Oracle Identity Federation" on page 6-135.
2. Open the `$ORACLE_HOME/fed/conf/config.xml` file.
3. Locate the `FederationConfig` XML element, and set its `useLocalConfig` attribute to `true`:

   ```
   <FederationConfig useLocalConfig="true">
   ```

4. Locate the XML `Config` element named `serverconfig`, and find the `soaprequiresslcert` property:

   ```
   <Config name="serverconfig">
   ...
      <property name="soaprequiresslcert">false</property>
   ...
   </Config>
   ```

   To force Oracle Identity Federation to require SSL certificates on the SOAP endpoint, change the value of the property to `true`. To accept SOAP requests without SSL client certificates, change the value of the property to `false`.
5. Save the file and exit.
6. Restart the OC4J_FED instance.

Protecting the Liberty 1.x / SAML 2.0 SOAP Endpoint

Oracle Identity Federation provides two methods to protect the SOAP endpoint used in the Liberty 1.x / SAML 2.0 protocols:

- SSL with Client Authentication via SSL Certificate: the SOAP Endpoint is protected with SSL, and by requiring an SSL Client Certificate
- HTTP Basic Authentication: with this method, the SOAP Endpoint is protected using the HTTP Basic Authentication mechanism.

This section explains how to implement these techniques.

SSL Client Authentication

Refer to "Using SSL with Oracle Identity Federation" on page 6-133 for details on how to:

- configure SSL to protect the SOAP URL

- configure Oracle Identity Federation to connect to SOAP endpoints protected by SSL

HTTP Basic Authentication

This section describes:

- how to configure HTTP Basic Authentication on the server to protect the SOAP URL
- how to configure the credentials that will be used when connecting to a remote server protected by HTTP Basic Authentication using the SOAP protocol

> **Note:** When it is integrated with Oracle Single Sign-On, the Oracle Identity Federation server cannot be protected using HTTP Basic Authentication.

Configuring HTTP Basic Authentication to protect the SOAP URL

This section lists the steps needed to protect the Liberty 1.x / SAML 2.0 SOAP endpoint. With these steps, you create a file defining the user credentials, and modify a configuration file to indicate the URL that needs to be protected.

The configuration changes will be made on the Oracle HTTP Server, using the basic `mod_auth` module.

The steps are as follows:

1. Create the user credentials file to hold the username and password entries, using the `htpasswd` utility:

   ```
   $ORACLE_HOME/Apache/Apache/bin/htpasswd -c $ORACLE_
   HOME/Apache/Apache/conf/.htpasswd-fed SOAP_USERNAME
   ```

   where *`SOAP_USERNAME`* is the username of the first account to be added to the newly created file. You will be prompted for the password.

   To add other entries to the file, use the following command:

   ```
   $ORACLE_HOME/Apache/Apache/bin/htpasswd $ORACLE_
   HOME/Apache/Apache/conf/.htpasswd-fed SOAP_USERNAME
   ```

   where *`SOAP_USERNAME`* is the username of the account to be added to the file. You will be prompted for the password.

2. Open the `$ORACLE_HOME/Apache/Apache/conf/mod_oc4j.conf` file.
3. Add the following element inside the `<IfModule mod_oc4j.c>` element (to prompt for basic authentication for any requests made to the soap backend), replacing the `$ORACLE_HOME` variable with the full path to the `ORACLE_HOME` directory:

   ```
   <LocationMatch "/fed/.*/soap*">
      AuthType Basic
      AuthName "SOAPBasicAuth"
      AuthUserFile $ORACLE_HOME/Apache/Apache/conf/.htpasswd-fed
      Require valid-user
   </LocationMatch>
   ```

4. Save and exit the file.
5. Restart Oracle HTTP Server using the command:

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

Note: After executing the command, if using RDBMS as the Oracle Identity Federation transient data store, restart OC4J_FED. Otherwise, click the **Refresh Server** button on the Oracle Identity Federation Administration Console.

Configuring Oracle Identity Federation to Connect to a Protected SOAP URL

On the client side, Oracle Identity Federation implements support for Basic Authentication when connecting to peer providers on the SOAP channel. A command-line tool is available to set up Oracle Identity Federation to use Basic Authentication with a given username/password when connecting to a specific remote provider. This tool, `fedbasicauth.jar`, updates the Oracle Identity Federation configuration so that the server can use the entered credentials when connecting to the SOAP endpoint of the remote provider.

When configuring the HTTP Basic Authn SOAP credentials, use this command to obtain help for the tool:

```
$ORACLE_HOME/jdk/bin/java -jar $ORACLE_HOME/fed/lib/fedbasicauth.jar
```

The tool responds with this information:

```
Usage: java -jar fedbasicauth.jar <options> [action]
Manage HTTP Basic Auth configuration for Oracle Identity Federation.
Options:
  -oh <path> The ORACLE_HOME directory. (REQUIRED)
  -providerid <id> The URI of the peer provider. (REQUIRED)
  -username <u> The username. (REQUIRED if action is -set)

Actions:
  -set Sets basic auth credentials (prompts for password). Basic auth is also
enabled for the provider.
  -remove Removes basic auth config for the provider.
  -enable Enables basic auth for the provider.
  -disable Disables basic auth for the provider.
  -show Shows basic auth config for the provider. Password is not displayed.
  --help Displays this usage information. (DEFAULT)
```

Some examples of tool usage follow.

Example 1

Sets username 'jsmith' (and prompted password) as basic authentication credentials for peer provider `http://idp.example.org`, and turns on basic authentication for the provider:

```
$ORACLE_HOME/jdk/bin/java
-jar $ORACLE_HOME/fed/lib/fedbasicauth.jar
-oh $ORACLE_HOME
-provider http://idp.example.org
-set -username jsmith
```

Example 2

Turns off basic authentication for provider `http://idp.example.org`:

```
$ORACLE_HOME/jdk/bin/java
-jar $ORACLE_HOME/fed/lib/fedbasicauth.jar
```

```
-oh $ORACLE_HOME
-provider http://idp.example.org
-disable
```

Example 3

Turns on basic authentication for provider `http://idp.example.org`. Assumes that the credentials are already set:

```
$ORACLE_HOME/jdk/bin/java
-jar $ORACLE_HOME/fed/lib/fedbasicauth.jar
-oh $ORACLE_HOME
-provider http://idp.example.org
-enable
```

Example 4

Turns off basic authentication for provider `http://idp.example.org`, and removes the credentials:

```
$ORACLE_HOME/jdk/bin/java
-jar $ORACLE_HOME/fed/lib/fedbasicauth.jar
-oh $ORACLE_HOME
-provider http://idp.example.org
-remove
```

7

Additional Server Configuration

Additional topics pertinent to Oracle Identity Federation server configuration and management are described here. This includes:

- Setting up Single Sign-On Services
- Working with Affiliations
- Exporting the IdP's self-signed certificate to the SP
- How to Use the Transient/One-time Identifier
- Configuring Name ID Formats
- How to Allow the IdP to Determine the Name ID Format
- How to Use Automatic Account Linking at the SP
- How to Use Automatic Account Linking at the IdP
- Interoperating with Microsoft ADFS
- Logout no-fail-on-error Option
- Logout Status
- Configuring SAML 2.0 Authentication Query Response
- Configuring SAML 2.0 Assertion ID Request
-

Setting up Single Sign-On Services

There are several ways to perform a federated single sign-on (SSO) operation, depending on the back-end in use and where the flow will be initialized. Table 7–1 shows the possible combinations:

Table 7–1 Federated Single Sign-On Combinations

Combination	Flow
OracleAS Single Sign-On with Liberty 1.x/SAML 2.0	User accesses a resource protected by `mod_osso`, triggering SAML 2.0/Liberty 1.x single sign-on, with Oracle Identity Federation acting as SP.
Oracle Access Manager with Liberty 1.x/SAML 2.0	User accesses a resource protected by webgate, triggering SAML 2.0/Liberty 1.x single sign-on, with Oracle Identity Federation acting as SP.

Table 7–1 (Cont.) Federated Single Sign-On Combinations

Combination	Flow
Oracle Access Manager with SAML 1.x/WS-Federation	User accesses a resource protected by webgate, triggering SAML 1.x/WS-Federation single sign-on, with Oracle Identity Federation acting as SP.
eTrust SiteMinder with SAML 2.0/Liberty 1.x	User accesses a resource protected by eTrust SiteMinder Agent, triggering SAML 2.0/Liberty 1.x single sign-on, with Oracle Identity Federation acting as SP.
eTrust SiteMinder with SAML 1.x/WS-Federation	User accesses a resource protected by eTrust SiteMinder Agent, triggering SAML 1.x/WS-Federation single sign-on, with Oracle Identity Federation acting as SP.
SP-initiated single sign-on with SAML 2.0/Liberty 1.x	User initiates a Liberty 1.x/SAML 2.0 single sign-on by directly accessing an Oracle Identity Federation URL, with Oracle Identity Federation acting as SP.
SP-initiated single sign-on with SAML 1.x	User initiates a SAML 1.x single sign-on by directly accessing an Oracle Identity Federation URL, with Oracle Identity Federation acting as SP.
SP-initiated single sign-on with WS-Federation	User initiates a WS-Federation single sign-on by directly accessing an Oracle Identity Federation URL, with Oracle Identity Federation acting as SP.
IdP-initiated single sign-on with SAML 2.0/Liberty 1.x	User initiates a SAML 2.0/Liberty 1.x single sign-on by directly accessing an Oracle Identity Federation URL, with Oracle Identity Federation acting as IdP.
IdP-initiated single sign-on with SAML 1.x	User initiates a SAML 1.x single sign-on by directly accessing an Oracle Identity Federation URL, with Oracle Identity Federation acting as IdP.
IdP-initiated single sign-on with WS-Federation	User initiates a WS-Federation single sign-on by directly accessing an Oracle Identity Federation URL, with Oracle Identity Federation acting as IdP.

This section explains how to configure the different combinations:

- OracleAS Single Sign-On with Liberty 1.x/SAML 2.0
- Oracle Access Manager with Liberty 1.x/SAML 2.0
- Oracle Access Manager with SAML 1.x/WS-Federation
- eTrust SiteMinder with Liberty 1.x/SAML 2.0
- eTrust SiteMinder with SAML 1.x/WS-Federation
- SP-initiated SSO with Liberty 1.x/SAML 2.0
- SP-initiated SSO with SAML 1.x
- SP-initiated SSO with WS-Federation
- IdP-initiated SSO with Liberty 1.x/SAML 2.0
- IdP-initiated SSO with SAML 1.x
- IdP-initiated SSO with WS-Federation

OracleAS Single Sign-On with Liberty 1.x/SAML 2.0

OracleAS Single Sign-On can be configured to trigger a Liberty 1.x or SAML 2.0 SSO operation when requesting a resource protected by `mod_osso`.

Note: This feature is not supported with SAML 1.x and WS-Federation protocols.

To achieve this, the OracleAS Single Sign-On partner application must be defined and must be protected by `mod_osso`. The partner application must also be configured to use the SSO Security level associated with the SASSO Authentication plug-in. You do this by editing the `ORACLE_HOME/sso/conf/policy.properties` file of the Oracle SSO deployment, and setting the partner application (defined by its hostname and port) to the same security level as the `SASSOAuthLevel` property.

For example:

```
MediumHighSecurity_AuthPlugin = oracle.security.sso.server.auth.SASSOAuth
MediumSecurity_AuthPlugin = oracle.security.sso.server.auth.SSOServerAuth
...
www.app.com\:7890 = MediumHighSecurity
...
SASSOAuthnUrl = http\://oif_host\:oif_port/sso/authn
SASSOLogoutUrl = http\://oif_host\:oif_port/sso/jsp/sasso_logout_success.jsp
SASSOAuthLevel = MediumHighSecurity
```

Save the file and restart `OC4J_SECURITY` to apply the changes.

The next time a user attempts an unauthenticated access to the protected resource, the user is redirected to Oracle Identity Federation where a Liberty 1.x/SAML 2.0 SSO operation occurs.

URL Query Parameters

When requesting the protected resource, it is possible to specify URL query parameters that Oracle Identity Federation can use to perform the SSO operation. The parameters are:

- `providerid` - This is the identifier of the Liberty 1.x or SAML 2.0 IdP to use to perform the SSO operation (optional). If missing, the default SSO provider, set in the Oracle Identity Federation administration console at Service Provider -> Global Settings -> Default SSO Identity Provider, will be used.
- `federationid` - This is the identifier of the affiliation to use for the SSO (optional). An example of such a URL is:

  `http://protected_app:port/path?providerid=http://idp.com`

  See "Working with Affiliations" on page 7-12 for more information.

Note: Check that the URL query parameter values are correctly encoded.

Refer to the Oracle SSO documentation for details about OracleAS Single Sign-On configuration.

Oracle Access Manager with Liberty 1.x/SAML 2.0

Oracle Access policies can be set up to initiate a Liberty 1.x/SAML 2.0 SSO when the user requests a resource protected by the Oracle Access Manager WebGate agent.

To do this, use the Oracle Access Policy Manager to set up a policy domain or policy that protects the resource. When creating the authentication rule for the policy domain

or policy, select the Fed SSO – SAML 2.0/Liberty 1.x authentication scheme. Oracle Identity Federation automatically creates this authentication scheme when it is configured to use Oracle Access. The scheme initiates the Liberty 1.x or SAML 2.0 single sign-on when the resource is accessed, resulting in a session for the local user associated with the federated user. Set up the authorization rules and expression for the policy domain or policy to allow access for the resulting local user.

URL Query Parameters

When requesting the protected resource, it is possible to specify URL query parameters that Oracle Identity Federation can use to perform the SSO operation. The parameters are:

- `providerid` - This is the identifier of the Liberty 1.x or SAML 2.0 IdP to use to perform the SSO operation (optional). If missing, the default SSO provider, set in the Oracle Identity Federation administration console at Service Provider -> Global Settings -> Default SSO Identity Provider, will be used.
- `federationid` - This is the identifier of the affiliation to use for the SSO (optional). An example of such a URL is:

  ```
  http://protected_app:port/path?providerid=http://idp.com
  ```

  See "Working with Affiliations" on page 7-12 for more information.

> **Note:** Check that the URL query parameter values are correctly encoded.

Refer to the *Oracle Access Manager Identity and Common Administration Guide* for configuration details.

Oracle Access Manager with SAML 1.x/WS-Federation

Oracle Access Manager policies can be set up to initiate a SAML 1.x or WS-Federation SSO when a user requests a resource protected by the Oracle Access Manager WebGate agent.

To do this, use the Oracle Access Policy Manager to set up a policy domain or policy that protects the resource. When creating the authentication rule for the policy domain or policy, select the Fed SSO – SAML 1.x or Fed SSO – WS-Federation authentication scheme. Oracle Identity Federation automatically creates these authentication schemes when it is configured to use Oracle Access Manager. The schemes use SAML 1.x or WS-Federation to initiate a federated SSO when the resource is accessed, resulting in a session for the local user associated with the federated user. Set up the authorization rules and expression for the policy domain or policy to allow access for the resulting local user.

This section contains these topics:

Using the Fed SSO - SAML 1.x Authentication Scheme

When the Fed SSO – SAML 1.x authentication scheme is in use, Oracle Identity Federation will look for a cookie named `ObSAMLDomain` in the HTTP request which indicates the IdP that is to be used for the SSO. Oracle Identity Federation automatically sets this persistent cookie to the name of the IdP's configured domain at

the end of a SAML 1.x SSO. Consequently, the user must have performed at least one successful SAML 1.x SSO for this feature to work. If Oracle Identity Federation does not find the `ObSAMLDomain` cookie, it assumes the user is local and performs a local Access Manager login.

> **Note:** This feature was known as SmartMarks in Oracle COREid Federation and is now known in release 10.1.4.2 as SP-initiated IdP discovery.

Some additional Oracle Identity Federation configuration is needed at the SP for the SAML 1.x authentication scheme to work. In the SAML 1.x or WS-Federation Domain configuration for the IdP, SP-initiated IdP discovery must be enabled and the Transfer URL and Transfer Query String fields must be set. The Transfer URL is the URL used at the IdP to initiate a SAML 1.x SSO, and the Transfer Query String provides the parameters for the SSO to be directed back to the SP.

> **Note:** This URL and query string are not specified by the SAML 1.x specifications, so they will depend on the SAML implementation used by the IdP. When Oracle Identity Federation processes the SSO, it will concatenate the Transfer URL, the Transfer Query String, and the URL for the requested resource (the target) to be used in a redirection to the IdP. This requires that any target URL parameter be the last parameter in the Transfer Query String, as demonstrated the Oracle Identity Federation example which follows.

When the IdP is another Oracle Identity Federation installation, the Transfer URL and Transfer Query String are partially set according to Oracle Identity Federation conventions. The Transfer Query String needs to be completed with information specific to the IdP configuration for the SP: the SP domain name as configured at the IdP and the SSO method to be used.

Example

Consider an IdP named Acme and an SP named Zenith, both using Oracle Identity Federation. In the Acme domain configured at Zenith:

Transfer URL =
`https://fed.acme.com/shareid/saml/ObSAMLTransferService`

Transfer Query String = `DOMAIN=Zenith&METHOD=post&TARGET=`

If the target URL is `https://www.acme.com/`, then the complete redirection URL is:

```
https://fed.acme.com/shareid/saml/ObSAMLTransferService?DOMAIN=Z
enith&METHOD=post&TARGET=https://www.zenith.com/
```

> **Note:** This is the Oracle Identity Federation URL for IdP-initiated SSO with SAML 1.x as discussed in "SP-initiated SSO with SAML 1.x" on page 7-10.

Using the Fed SSO - WS-Federation Authentication Scheme

When the Fed SSO – WS-Federation authentication scheme is in use, Oracle Identity Federation uses the `ObSAMLDomain` cookie to determine the IdP as described earlier for SAML 1.x. However, unlike SAML 1.x, if the `ObSAMLDomain` cookie is not found,

Oracle Identity Federation will display an IdP selection page listing the configured IdPs that can be used for WS-Federation. If there is only one such IdP, Oracle Identity Federation skips the selection page and immediately uses that IdP. Once the WS-Federation SSO has completed successfully, Oracle Identity Federation will set the `ObSAMLDomain` cookie to the IdP domain. So the user only has to select the IdP on the first access to the SP.

Since the WS-Federation Passive Requester Profile specifies how to perform SP-initiated SSO, no additional Oracle Identity Federation configuration is required to make the WS-Federation authentication scheme work.

Like all Access authentication schemes, the Fed SSO – SAML 1.x and Fed SSO – WS-Federation schemes have authentication levels that represent the strength of the authentication method. The default authentication level for the Fed SSO schemes is 1 (the lowest strength), but these levels may be adjusted using the Access System Console.

Note: Oracle Identity Federation also creates other authentication schemes for the configured assertion mappings, and the authentication level of the assertion mapping scheme used to create the local session must be equal to or greater than the WS-Federation SSO scheme levels.

---

eTrust SiteMinder with Liberty 1.x/SAML 2.0

Liberty 1.x/SAML 2.0 SSO can be triggered by requesting a resource protected by eTrust SiteMinder policy.

To do this, create a policy under eTrust SiteMinder, which will protect a resource using the Form Authentication scheme. You will need to add a Form Authentication scheme through the eTrust SiteMinder Admin Console. While adding the Authentication scheme, select the scheme type as HTML Form Template, then make these changes:

1. Set Protection level to `1`.
2. Under the **Scheme Setup** tab:
   a. Set Server name to `CASiteMinderHost.domain:Port`
   b. Set Target to `/siteminderagent/forms/login.fcc.redirect.html`
3. Under the **Advance** tab, set Library to `smauthhtml`.

After configuring the Form Authentication scheme, add the HTML file shown here under `<NetegritySMInstallArea>\SMWebAgent\Samples\Forms`, naming it `login.fcc.redirect.html`.

```
<html>
 <!--Redirect to OIF-->
    <body onload="setTimeout(location.href=getRedirectURL(), 1);"/>
    <script>
    //<!--

    function getRedirectURL ()
    {
        var targetParam = getQueryParam("TARGET");
        if (targetParam.indexOf("$SM$") == 0)
            targetParam = targetParam.substring
             (4, targetParam.length);
        // Redirect
        var redirectURL =
```

```
            "http://<SPHost.Domain>:<Port>/fed/sp/smredirect?providerid="
            + escape("http://<idPHost.Domain>:<Port>/fed/idp")
            +"&targetURL=" + targetParam;

        return redirectURL;
    }

    function getQueryParam (param)
    {
        var result = "";
        var url = location.href;
        var qIndex = url.indexOf("?");
        if (qIndex > -1)
        {
            var queryString = url.substring(qIndex + 1, url.length);
            var params = queryString.split("&");
            for (var i = 0, n = params.length; i < n; ++i)
            {
                var nvPair = params[i].split("=");
                if (nvPair[0] == param)
                {
                    result = nvPair[1];
                    break;
                }
            }
        }
        return result;
    }
    //-->
    </script>
</html>
```

This form redirects the user to a servlet on the SP that will:

- process the targetURL query parameter
- initiate a flow that results in the user being ultimately redirected to the URL specified by this parameter once single sign-on completes

URL Query Parameters

When requesting the protected resource, it is possible to specify URL query parameters that Oracle Identity Federation can use to perform the SSO operation. The parameters are:

- `providerid` - This is the identifier of the Liberty 1.x or SAML 2.0 IdP to use to perform the SSO operation (optional). If missing, the default SSO provider, set in the Oracle Identity Federation administration console at **Service Provider** - > **Global Settings** - > **Default SSO Identity Provider**, will be used.
- `federationid` - This is the identifier of the affiliation to use for the SSO (optional). An example of such a URL is:

  `http://protected_app:port/path?providerid=http://idp.com`

  See "Working with Affiliations" on page 7-12 for more information.

> **Note:** Check that the URL query parameter values are correctly encoded.

Refer to the eTrust SiteMinder documentation for configuration details.

eTrust SiteMinder with SAML 1.x/WS-Federation

SAML 1.x/WS-Federation SSO can be triggered by requesting URLs in a specific format.

Using SAML 1.x Authentication

To trigger SAML 1.x SSO from the IdP, use a URL in the following format:

`http(s)://<idPHost.Domain>:<Port>/shareid/saml/ObSAMLTransferService?DOMAIN=<SP-DomainName>&METHOD=POST&TARGET=<Resource-protected-by-CASiteMinder>`

where `Resource-protected-by-CASiteMinder` is any resource protected by eTrust SiteMinder.

To initiate SAML 1.x SSO from the SP, protect the resource at eTrust SiteMinder by a Form Authentication scheme as described earlier in "eTrust SiteMinder with Liberty 1.x/SAML 2.0" on page 7-6. Use a form that looks something like this:

```
<html>
 <!--Redirect to OSFS-->
    <body onload="setTimeout(location.href=getRedirectURL(), 1);"/>
    <script>
    //<!--

    function getRedirectURL ()
    {
        var targetParam = getQueryParam("TARGET");
        if (targetParam.indexOf("$SM$") == 0)
            targetParam = targetParam.substring(4, targetParam.length);
        // Redirect
        var redirectURL =
          "http://<SPHost.Domain>:<Port>/shareid/saml/ObSAMLTransferForm?"+
         "&TARGET=" + targetParam;
        return redirectURL;
    }

    function getQueryParam (param)
    {
        var result = "";
        var url = location.href;
        var qIndex = url.indexOf("?");
        if (qIndex > -1)
        {
            var queryString = url.substring(qIndex + 1, url.length);
            var params = queryString.split("&");
            for (var i = 0, n = params.length; i < n; ++i)
            {
                var nvPair = params[i].split("=");
                if (nvPair[0] == param)
                {
                    result = nvPair[1];
                    break;
                }
            }
        }
        return result;
    }
```

```
    //-->
    </script>
</html>
```

Accessing the SP resource redirects the user to an Oracle Identity Federation servlet, which looks for a cookie (named `ObSAMLDomain`) in the HTTP request that indicates which IdP to use for the SSO, and then redirects the user for authentication to that IdP. Oracle Identity Federation automatically sets this persistent cookie to the name of the IdP's configured domain at the end of an idP-initiated SAML 1.x single sign-on. This means that the user must have done at least one successful SAML 1.x SSO for the feature to work. If Oracle Identity Federation does not find the `ObSAMLDomain` cookie, it assumes the user is local and performs a local login at the Service Provider domain.

> **Note:** This feature was known as SmartMarks in Oracle COREid Federation.

Some additional Oracle Identity Federation configuration is needed at the SP for the SAML 1.x authentication scheme to work. In the SAML 1.x/WS-Federation Domain configuration for the IdP, SP-initiated IdP discovery must be enabled and the Transfer URL and Transfer Query String fields must be set. The Transfer URL is the URL used at the IdP to initiate a SAML 1.x SSO, and the Transfer Query String provides the parameters for the SSO to be directed back to the SP.

> **Note:** This URL and query string are not specified by the SAML 1.x specifications, so they will depend on the SAML implementation used by the IdP. When Oracle Identity Federation processes the SSO, it will concatenate the Transfer URL, the Transfer Query String, and the URL for the requested resource (the target) to be used in a redirection to the IdP. This requires that any target URL parameter be the last parameter in the Transfer Query String.

When the IdP is another Oracle Identity Federation installation, the Transfer URL and Transfer Query String are partially set according to Oracle Identity Federation conventions. The Transfer Query String must be completed with information specific to the IdP configuration for the SP: the SP domain name as configured at the IdP, and the SSO method to be used.

Using WS-Federation Authentication

To trigger WS-Federation SSO, use URLs in the following format:

```
http(s)://<SPHost.Domain>:<Port>/shareid/wsfederation/ObWSFedRes
ourceSTS?wa=wsignin1.0&wctx=<Resource-Protected-by-CASiteMinder>
```

Here `Resource-protected-by-CASiteMinder` is any resource protected by eTrust SiteMinder.

Oracle Identity Federation uses the `ObSAMLDomain` cookie in the HTTP request, which indicates the IdP to be used for authentication. If the `ObSAMLDomain` cookie is not found, Oracle Identity Federation displays an IdP selection page listing the configured IdPs that can be used for WS-Federation. If there is only one such IdP, Oracle Identity Federation skips the selection page and immediately uses that IdP. Once the WS-Federation SSO has completed successfully, Oracle Identity Federation sets the `ObSAMLDomain` cookie to the IdP domain. The user only needs to select the IdP on the first access to the SP.

Here is an example URL:

```
http://costarica.myorg.co.in:7779/shareid/wsfederation/ObWSFedResourceSTS?wa=wsignin1.0&wctx=http://mulshi.myorg.co.in/mytest/index.html
```

SP-initiated SSO with Liberty 1.x/SAML 2.0

When Oracle Identity Federation server is integrated with OracleAS Single Sign-On, Oracle Access Manager or CA eTrust SiteMinder, a user can initiate a Liberty 1.x/SAML 2.0 SSO operation by directly requesting a service at the Oracle Identity Federation/SP instance.

The URL to be requested on the Oracle Identity Federation server is:

```
http(s)://Oracle Identity Federation_host:Oracle Identity Federation_port/fed/sp/initiatesso
```

URL Query Parameters

It is possible to specify query parameters when requesting that URL:

- `providerid` - This is the identifier of the Liberty 1.x or SAML 2.0 IdP to use to perform the SSO operation (optional). If missing, the default SSO provider, set in the Oracle Identity Federation administration console at **Service Provider** - > **Global Settings** - > **Default SSO Identity Provider**, will be used.
- `federationid` - This is the identifier of the affiliation to use for the SSO (optional).

  See "Working with Affiliations" on page 7-12 for more information.
- `returnurl` - This is the URL to which the user is sent after a successful SSO operation. It is required if the Unsolicited Relay State property, set in the Oracle Identity Federation administration console at **Server Properties** - > **Circle Of Trust** - > **Settings** for the IdP, is empty.

An example of such a URL is:

```
http://oif_host:oif_port/fed/sp/initiatesso?providerid=http://idp.com&returnurl=ProtectedAppURL
```

Check that the query parameter values are correctly URL-encoded.

SP-initiated SSO with SAML 1.x

When Oracle Identity Federation is integrated with OracleAS Single Sign-On, Oracle Access Manager, or eTrust SiteMinder, it is possible to initiate a SAML 1.x SSO operation by directly requesting a service at the Oracle Identity Federation/SP instance using the following URL:

```
http(s)://oif_host:oif_port/shareid/saml/ObSAMLTransferForm?TARGET=targetURL
```

where `targetURL` is the URL of the requested resource. The ObSAMLTransferForm service follows the same rules as the Fed SSO – SAML 1.x authentication scheme discussed in "Using the Fed SSO - SAML 1.x Authentication Scheme" on page 7-4. It looks for the `ObSAMLDomain` cookie to specify the IdP to be used. If the `ObSAMLDomain` cookie is not found, a local login is performed.

For example:

```
http://oif_host:oif_port/shareid/saml/ObSAMLTransferForm?
TARGEt=http://host:port/protected.html
```

> **Note:** With Oracle Access Manager, you can protect the targetURL by a policy domain using any authentication scheme, as long as the authentication level of the scheme is less than or equal to the authentication level of the assertion mapping authentication scheme used to create the local user.

SP-initiated SSO with WS-Federation

When Oracle Identity Federation is integrated with OracleAS Single Sign-On, Oracle Access Manager, or eTrust SiteMinder, you can initiate a WS-Federation SSO request from the SP with a URL with the format:

```
http(s)://oif_host:oif_
port/shareid/wsfederation/ObWSFedResourceSTS?wa=wsignin1.0&
wctx=targetURL
```

where `targetURL` is the URL of the requested resource. The ObWSFedResourceSTS service follows the same rules as the Fed SSO – WS-Federation authentication scheme discussed in "Using the Fed SSO - WS-Federation Authentication Scheme" on page 7-5. If the `ObSAMLDomain` cookie is present, it is used to determine the IdP. If there is only one IdP configured for WS-Federation, that IdP is used. Otherwise an IdP selection page is displayed.

Here is an example URL:

```
http://host:port/shareid/wsfederation/ObWSFedResourceSTS?
wa=wsignin1.0&wctx=http://host:port/protected.html
```

> **Note:** With Oracle Access Manager, you can protect the targetURL by a policy domain using any authentication scheme, as long as the authentication level of the scheme is less than or equal to the authentication level of the assertion mapping authentication scheme used to create the local user.

IdP-initiated SSO with Liberty 1.x/SAML 2.0

For the Liberty 1.x and SAML 2.0 protocols, Oracle Identity Federation provides the ability to initiate an SSO operation by directly requesting a URL at the Oracle Identity Federation instance acting as an IdP; this is called an SSO IdP-initiated operation.

The url to be requested on the Oracle Identity Federation server is of the form:

```
http(s)://oif_host:oif_port/fed/idp/initiatesso
```

URL Query Parameters

It is possible to specify query parameters when requesting that URL:

- `providerid` - This is the identifier of the Liberty 1.x or SAML 2.0 SP to use to perform the SSO operation (required).
- `federationid` - This is the identifier of the affiliation to use for the SSO (optional).

  See "Working with Affiliations" on page 7-12 for more information.

- `returnurl` - This is the url to which the user is sent after a successful SSO operation (optional).

An example of such a URL is:

```
http://oif_host:oif_
port/fed/idp/initiatesso?providerid=http://sp.com&returnurl=Prot
ectedAppURL
```

> **Note:** Check that the query parameter values are correctly URL-encoded.

IdP-initiated SSO with SAML 1.x

When Oracle Identity Federation is enabled for SSO with SAML1.x, it is possible to initiate an SSO request from the IdP. The URL for the request is of the form:

```
http(s)://oif_host:oif_port/shareid/saml/ObSAMLTransferService
```

with these parameters:

- `DOMAIN` is the domain name configured for the SP
- `METHOD` is POST or ARTIFACT
- `TARGET` is the target URL that is to be accessed

For example:

```
http://oif_host:oif_
port/shareid/saml/ObSAMLTransferService?DOMAIN=SP_
domain&METHOD=post&TARGET=http://host2:port/protected.html
```

IdP-initiated SSO with WS-Federation

When Oracle Identity Federation is enabled for SSO with WS-Federation, it is possible to initiate an SSO request from the IdP, using a URL in the following format:

```
http(s)://oif_host:oif_
port/shareid/wsfederation/ObWSFedIdentitySTS
```

with these parameters:

- `wa` is `wsignin1.0`
- `wtrealm` is the request realm URI defined by the SP
- `wctx` is the target URL to be accessed

For example:

```
http://oif_host:oif_
port/shareid/wsfederation/ObWSFedIdentitySTS?wa=wsignin1.0&
wtrealm=http://sp_host/&wctx=http://sp_host:port/protected.html
```

Working with Affiliations

The run-time functioning of affiliations depend on whether the Oracle Identity Federation server is acting as an IdP or an SP.

Oracle Identity Federation Acting as IdP

When Oracle Identity Federation is an IdP, provided the affiliation/SP is present and enabled in the circle of trust, the Oracle Identity Federation server is ready to process any requests originating from service providers using the affiliation.

Oracle Identity Federation Acting as SP

As an SP, you can trigger a single sign-on operation with an IdP using an affiliation to which the SP belongs. To do so, just include a `federationid` query parameter in the URL protected by the IdM back-end, and set the parameter value to the affiliation ID.

For example with an OracleAS Single Sign-On back-end, assuming that a resource is protected by `mod_osso` and configured for Oracle Identity Federation authentication, requesting the URL of this resource with the `federationid` query parameter will instruct Oracle Identity Federation to use an affiliation when performing single sign-on with a peer IdP. Here is an example of such a URL:

```
http://protected_res_host:protected_res_
port/path?federationid=http://affiliationid
```

It is also possible to directly access the `http://oif_host:oif_port/fed/sp/initiatesso` URL with the same `federationid` query parameter. In this case, Oracle Identity Federation will trigger a single sign-on operation, and will use the Unsolicited SSO RelayState for the peer IdP as the URL to which the user is redirected after successful authentication.

Note: The Unsolicited SSO RelayState is set on the **Server Configuration** -> **Circle of Trust** -> **Edit Trusted Provider** page of the Oracle Identity Federation administration console.

Exporting the IdP's self-signed certificate to the SP

Take these steps to export an identity provider's self-signed certificate into the service provider's keystore. This procedure is utilized in SAML 1.x and WS-Federation configurations.

1. At the Identity Provider, export the self-signed certificate from the keystore by running the keytool utility in the Oracle Identity Federation home directory:

   ```
   C:\<OIFHome>\jdk\jre\lib\security>keytool -keystore
   C:\<OIFHome>\fed\shareid\oblix\config\keystore -storepass <password>
      -export -alias shareid -file <myfile>
   ```

   where `storepass` is the password that was specified when Oracle Identity Federation was installed.

2. Copy the certificate file to a location that can be accessed by the service provider.

3. At the service provider, import the IdP's certificate into the keystore:

   ```
   C:\<OIFHome>\jdk\jre\lib\security>keytool -keystore
   C:\<OIFHome>\fed\shareid\oblix\config\keystore -storepass <password>
      -import -alias <anyName>-file <myfile>
   ```

4. Restart the Oracle Identity Federation service provider from the Oracle Enterprise Manager console.

How to Use the Transient/One-time Identifier

The transient/one-time identifier, commonly referred as the anonymous identifier, is used when the Service Provider (SP) only requires the user to be authenticated by the Identity Provider (IdP) but does not need to know the user's identity.

The IdP typically knows the user's identity, since it will have authenticated the user, but the SP will have no knowledge of any aspects of the identity, except the fact that the user has been authenticated at the IdP.

Transient/one-time identifiers must be configured at both the SP and the IdP.

Take these configuration steps at the service provider:

1. Log on to the Oracle Identity Federation administration console.
2. Go to **Server Configuration** - > **Service Provider**.
3. In the Anonymous User Identifier field, enter the User ID of an account from the user data store that will be used to identify anonymous users. This User ID should be of the same type as the one used to reference users (that is, corresponding to the User ID Attribute or Login ID Column on the User Data Store page).
4. Click **Save**.
5. Go to **Server Configuration** - > **Service Provider**, then choose Liberty 1.2 or SAML 2.0 as appropriate.

   Note: Transient identifiers are not supported for Liberty 1.1.

6. For the Default Authn Request NameID Format, select Transient/One-time Identifier.
7. Click **Save**, then **Refresh Server**.

Take these configuration steps at the identity provider:

1. Log on to the Oracle Identity Federation administration console.
2. Go to **Server Configuration** - > **Identity Provider**, then choose Liberty 1.2 or SAML 2.0 as appropriate.

   Note: Transient identifiers are not supported for Liberty 1.1.

3. Click the **Select** button for Assertion NameID Formats.
4. Check the box to enable the Transient/One-time Identifier Name ID format.
5. Click **Apply**, then **Refresh Server**.

Configuring Name ID Formats

Oracle Identity Federation supports several kinds of Name ID to reference a user when exchanging SAML messages with remote providers. They include:

- X.509 Subject Name

  A name ID based on this format will typically contain a Distinguished Name (DN) referencing a user in an LDAP directory (for example: `cn=alice,cn=users,dc=oracle,dc=com`). This format is identified by the URI `urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName`.

- Email Address

  The name ID value consists of an email address (for example: `alice@oracle.com`). This Name ID type is identified by the URI `urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress`.

- Windows Domain Qualified Name

  A Windows domain qualified user name is a string of the form `DomainName\UserName`. This Name ID type is identified by the URI `urn:oasis:names:tc:SAML:1.1:nameid-format:WindowsDomainQualifiedName`.

- Kerberos Principal Name

  Indicates that the content of the Name ID element is in the form of a Kerberos principal name using the format `name[/instance]@REALM`. This format is identified by the URI `urn:oasis:names:tc:SAML:2.0:nameid-format:kerberos`.

- Persistent Identifier

  Indicates that the content of the element is a persistent opaque identifier for a principal that is specific to one identity provider and one service provider (or affiliation of service providers). This Name ID type is identified by the URI `urn:oasis:names:tc:SAML:2.0:nameid-format:persistent`.

- Transient Identifier

  Indicates that the content of the element is an opaque identifier (similar to a Persistent identifier) whose lifetime is limited to the current user session. This Name ID type is identified by the URI `urn:oasis:names:tc:SAML:2.0:nameid-format:transient`.

- Unspecified

  The content of a Name ID with this Format is not bound to the SAML 2.0 specifications, but only to the implementations interacting together. This Name ID type is identified by the URI `urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified`.

- Customizable Name ID Format

  Oracle Identity Federation provides a way for the administrator to define a Name ID format that will fit the needs of a specific deployment. With a customizable format, it is also possible to define the URI by which the format will be identified. This format is not defined by the SAML 2.0 specifications.

When generating a Name ID to include in an outgoing message, Oracle Identity Federation:

- creates a random value for the NameID when the format is of opaque type, such as Persistent or Transient, and uses a federation record to store the information about the created random name ID.
- retrieves a user attribute for the NameID when the format is of non-opaque type, such as X.509 Subject Name, Email Address, Windows Domain Qualified Name, Kerberos Principal Name, Unspecified and the Customizable Name ID Format.

The following configurations are described:

- Configuring the Name ID Formats as an IdP
- Configuring the Name ID Formats as an SP

- Configuring the Name ID Formats for a Specific Remote Provider
- Configuring Attributes in SSO Assertions with Oracle Identity Federation/IdP

Configuring the Name ID Formats as an IdP

To configure the X.509 Subject Name, Email Address, Windows Domain Qualified Name, Kerberos Principal Name, Persistent and Transient Name ID Formats, follow the instructions provided in "Select SAML 2.0 Identity Provider NameID Formats" on page 6-33.

The Unspecified and Customizable formats are not configurable through the Oracle Identity Federation Administration Console. Take these steps to configure the formats (and any other formats that must be enabled, if Unspecified and Customizable are configured):

1. Open the `$ORACLE_HOME/fed/conf/config.xml` file.
2. Locate the `FederationConfig` XML element, and set its `useLocalConfig` attribute to `true`:

```
<FederationConfig useLocalConfig="true">
```

3. Locate the XML `Config` element named `idpsaml20`, and look for the `nameidformats` propertyset. It contains the list of enabled Name ID Formats on the Oracle Identity Federation/IdP. If a specific format is not enabled, Oracle Identity Federation will not be able to process messages containing this format. For a format to be enabled, it needs to be added to the list. In the example below, all the Name ID Formats are enabled:

```
<Config name="idpsaml20">
 ...
   <propertyset name="nameidformats">
   <propertyvalue>x509</propertyvalue>   <!-- X.509 Subject Name -->
   <propertyvalue>emailaddress</propertyvalue>  <!-- Email Address -->
   <propertyvalue>windowsnamequalifer</propertyvalue> <!-- Windows Domain
Qualified Name -->
   <propertyvalue>kerberos</propertyvalue>   <!-- Kerberos Principal Name -->
   <propertyvalue>persistent</propertyvalue>   <!-- Persistent Identifier -->
   <propertyvalue>transient</propertyvalue>   <!-- Transient Identifier -->
   <propertyvalue>unspecified</propertyvalue>  <!-- Unspecified -->
   <propertyvalue>custom</propertyvalue>   <!-- Customizable Name ID format -->
   </propertyset>
 ...
</Config>
```

> **Note:** It should be stressed that if configuring Unspecified or Customizable formats, any other format that needs to be enabled must also appear in this list. Thus, if you want to enable Email, X509 and Unspecified formats, you cannot specify this combination of formats through the Administration Console, but will need to make the file edits instead as shown here.

The possible values for the `nameidformats` setting are:

- `x509` for X.509 Subject Name
- `emailaddress` for Email Address

- `windowsnamequalifer` for Windows Domain Qualified Name
- `kerberos` for Kerberos Principal Name
- `persistent` for Persistent Identifier
- `transient` for Transient Identifier
- `unspecified` for Unspecified
- `custom` for Customizable Name ID

4. Once the Name ID Formats are enabled, map the non-opaque enabled Name ID Types to user attributes (the Customizable Name ID Format requires extra configuration):

   - To map the X.509 Subject Name Format, locate the XML `Config` element named `idpsaml20`, look for the `nameformatx500` property, and set its value to the user attribute:

     ```
     <Config name="idpsaml20">
     ...
        <property name="nameformatx500">dn</property>
     ...
     </Config>
     ```

   - To map the Email Address Format, locate the XML `Config` element named `idpsaml20`, look for the `nameformatemail` property, and set its value to the user attribute:

     ```
     <Config name="idpsaml20">
     ...
        <property name="nameformatemail">mail</property>
     ...
     </Config>
     ```

   - To map the Windows Domain Qualified Name Format, locate the XML `Config` element named `idpsaml20`, look for the `nameformatwindows` property, and set its value to the user attribute:

     ```
     <Config name="idpsaml20">
     ...
        <property name="nameformatwindows">windowsDQN</property>
     ...
     </Config>
     ```

   - To map the Kerberos Principal Name Format, locate the XML `Config` element named `idpsaml20`, look for the `nameformatkerberos` property, and set its value to the user attribute:

     ```
     <Config name="idpsaml20">
     ...
        <property name="nameformatkerberos">kerberosid</property>
     ...
     </Config>
     ```

   - To map the Unspecified Format, locate the XML `Config` element named `idpsaml20`, look for the `nameformatunspecified` property, and set its value to the user attribute:

     ```
     <Config name="idpsaml20">
     ...
        <property name="nameformatunspecified">uid</property>
     ...
     ```

```
</Config>
```

- To configure the Customizable Name ID, locate the XML `Config` element named `idpsaml20`, look for the `nameformatcustom` property, and set its value to the user attribute. The format for this Name ID also needs to be configured: locate the XML `Config` element named `idpsaml20`, look for the `customnameidformat` property, and set its value to the format the administrator wants to use. Finally the Customizable Name ID format feature needs to be enabled, in addition to adding it to the list of enabled Name IDs in step 3: Locate the XML `Config` element named `idpsaml20`, look for the `customnameidenabled` property, and set its value to `true` if the Customizable Name ID needs to be enabled, otherwise set it to `false`.

  Here is an example of the configuration for the Customizable Name ID Format:

```
<Config name="idpsaml20">
...
   <property name="customnameidenabled">true</property>
   <property name="customnameidformat">Social-Security-Number</property>
   <property name="nameformatcustom">ssn</property>
...
</Config>
```

5. Finally, the Default Assertion NameID Format needs to be set. That setting will determine which kind of Name ID to send in an SSO Assertion when the Service Provider does not specify any particular one to use. Locate the XML `Config` element named `idpsaml20`, look for the `defaultnameidformat` property, and set its value to the Name ID Format to be used:

```
<Config name="idpsaml20">
...
   <property name="defaultnameidformat">x509</property>
...
</Config>
```

   The possible values for the `defaultnameidformat` setting are:

   - `x509` for X.509 Subject Name
   - `emailaddress` for Email Address
   - `windowsnamequalifer` for Windows Domain Qualified Name
   - `kerberos` for Kerberos Principal Name
   - `persistent` for Persistent Identifier
   - `transient` for Transient Identifier
   - `unspecified` for Unspecified
   - `custom` for Customizable Name ID

6. Save the file and exit.

Note: Navigating to the Server Properties -> Identity Provider -> SAML 2.0 page, then clicking the Assertion NameID Formats, making any (or no) changes on the new screen and clicking **Save** will remove any of the settings not defined in the screen; the Unspecified and Customizable Name ID Format configuration will be removed.

---

7. If using RDBMS as the Oracle Identity Federation transient data store, restart OC4J_FED. Otherwise, click the **Refresh Server** on the Oracle Identity Federation Administration Console.

Configuring the Name ID Formats as an SP

To configure the X.509 Subject Name, Email Address, Windows Domain Qualified Name, Kerberos Principal Name, Persistent and Transient Name ID Formats in Oracle Identity Federation, follow the instructions in "Service Provider - SAML 2.0 Properties" on page 6-41 and "Select SAML 2.0 Service Provider NameID Formats" on page 6-46.

The Unspecified and Customizable formats are not configurable through the Oracle Identity Federation Administration Console. Take these steps to configure the formats (and any other formats that must be enabled, if Unspecified and Customizable are configured):

1. Open the `$ORACLE_HOME/fed/conf/config.xml` file.
2. Locate the `FederationConfig` XML element, and set its `useLocalConfig` attribute to `true`:

```
<FederationConfig useLocalConfig="true">
```

3. Locate the XML `Config` element named `spsaml20`, and look for the `nameidformats` propertyset. It contains the list of enabled Name ID Formats for automatic account linking on the Oracle Identity Federation/SP. If a specific format is not enabled, Oracle Identity Federation will not be able to automatically link federation records to users when processing assertions containing this format. For a format to be enabled, it needs to be added to the list. In the example below, all the Name ID Formats are enabled:

```
<Config name="spsaml20">
 ...
   <propertyset name="nameidformats">
   <propertyvalue>x509</propertyvalue>   <!-- X.509 Subject Name -->
   <propertyvalue>emailaddress</propertyvalue>  <!-- Email Address -->
   <propertyvalue>windowsnamequalifer</propertyvalue> <!-- Windows Domain
Qualified Name -->
   <propertyvalue>kerberos</propertyvalue>   <!-- Kerberos Principal Name -->
   <propertyvalue>unspecified</propertyvalue>  <!-- Unspecified -->
   <propertyvalue>custom</propertyvalue>   <!-- Customizable Name ID format -->
   </propertyset>
 ...
</Config>
```

> **Note:** It should be stressed that if configuring Unspecified or Customizable formats, any other format that needs to be enabled must also appear in this list. Thus, if you want to enable Email, X509 and Unspecified formats, you cannot specify this combination of formats through the Administration Console, but will need to make the file edits instead as shown here.

The possible values for the `nameidformats` setting are:

- `x509` for X.509 Subject Name
- `emailaddress` for Email Address

- `windowsnamequalifer` for Windows Domain Qualified Name
- `kerberos` for Kerberos Principal Name
- `unspecified` for Unspecified
- `custom` for Customizable Name ID

4. Once the Name ID Formats are enabled, map the non-opaque enabled Name ID Types to user attributes (the Customizable Name ID Format requires extra configuration):
   - To map the X.509 Subject Name Format, locate the XML `Config` element named `spsaml20`, look for the `nameformatx500` property, and set its value to the user attribute:

     ```
     <Config name="spsaml20">
     ...
        <property name="nameformatx500">dn</property>
     ...
     </Config>
     ```

   - To map the Email Address Format, locate the XML `Config` element named `spsaml20`, look for the `nameformatemail` property, and set its value to the user attribute:

     ```
     <Config name="spsaml20">
     ...
        <property name="nameformatemail">mail</property>
     ...
     </Config>
     ```

   - To map the Windows Domain Qualified Name Format, locate the XML `Config` element named `spsaml20`, look for the `nameformatwindows` property, and set its value to the user attribute:

     ```
     <Config name="spsaml20">
     ...
        <property name="nameformatwindows">windowsDQN</property>
     ...
     </Config>
     ```

   - To map the Kerberos Principal Name Format, locate the XML `Config` element named `spsaml20`, look for the `nameformatkerberos` property, and set its value to the user attribute:

     ```
     <Config name="spsaml20">
     ...
        <property name="nameformatkerberos">kerberosid</property>
     ...
     </Config>
     ```

   - To map the Unspecified Format, locate the XML `Config` element named `spsaml20`, look for the `nameformatunspecified` property, and set its value to the user attribute:

     ```
     <Config name="spsaml20">
     ...
        <property name="nameformatunspecified">uid</property>
     ...
     </Config>
     ```

- To configure the Customizable Name ID, locate the XML `Config` element named `spsaml20`, look for the `nameformatcustom` property, and set its value to the user attribute. The format for this Name ID also needs to be configured: locate the XML `Config` element named `spsaml20`, look for the `customnameidformat` property, and set its value to the format the administrator wants to use. Finally the Customizable Name ID format feature needs to be enabled, in addition to adding it to the list of enabled Name IDs in step 3: Locate the XML `Config` element named `spsaml20`, look for the `customnameidenabled` property, and set its value to `true` if the Customizable Name ID needs to be enabled, otherwise set it to `false`.

  Here is an example of the configuration for the Customizable Name ID Format:

```
<Config name="spsaml20">
...
   <property name="customnameidenabled">true</property>
   <property name="customnameidformat">Social-Security-Number</property>
   <property name="nameformatcustom">ssn</property>
...
</Config>
```

5. Finally, the Default Authn Request NameID Format needs to be set. That setting will determine which kind of Name ID the Oracle Identity Federation/SP will request when sending an Authn Request message to the IdP. The IdP would then authenticate the user and send back an SSO Assertion containing a Name ID of the requested format. Locate the XML `Config` element named `spsaml20`, look for the `defaultauthnrequestnameidformat` property, and set its value to the Name ID Format to be used:

```
<Config name="spsaml20">
...
   <property name="defaultauthnrequestnameidformat">x509</property>
...
</Config>
```

   The possible values for the `defaultauthnrequestnameidformat` setting are:

   - `x509` for X.509 Subject Name
   - `emailaddress` for Email Address
   - `windowsnamequalifer` for Windows Domain Qualified Name
   - `kerberos` for Kerberos Principal Name
   - `persistent` for Persistent Identifier
   - `transient` for Transient Identifier
   - `unspecified` for Unspecified
   - `custom` for Customizable Name ID

6. Save the file and exit.

Notes:

- Navigating to the Server Properties -> Identity Provider -> SAML 2.0 page, then clicking the Assertion NameID Formats, making any (or no) changes on the new screen and clicking **Save** will remove any of the settings not defined in the screen; the Unspecified and Customizable Name ID Format configuration will be removed.
- Navigating to the Server Properties -> Service Provider -> SAML 2.0, then clicking **Save** will reset the Default Authn Request NameID Format setting to a value defined in the screen; since the Unspecified and Customizable Name ID Format choice are not available on the Oracle Identity Federation Administration Console, it would set the Default Authn Request NameID Format to another value if it was previously set to one of these two settings.

7. If using RDBMS as the Oracle Identity Federation transient data store, restart OC4J_FED. Otherwise, click the **Refresh Server** on the Oracle Identity Federation Administration Console.

Configuring the Name ID Formats for a Specific Remote Provider

To configure the X.509 Subject Name, Email Address, Windows Domain Qualified Name, Kerberos Principal Name, Persistent and Transient Name ID Formats, follow the instructions in "Editing a Trusted Provider" on page 6-51 and "Edit Trusted Provider: Select NameID Formats" on page 6-59.

The Unspecified and Customizable formats are not configurable through the Oracle Identity Federation Administration Console. Take these steps to configure these formats (and any other formats that must be enabled, if Unspecified and Customizable are configured):

1. Open the `$ORACLE_HOME/fed/conf/config.xml` file.
2. Locate the `FederationConfig` XML element, and set its `useLocalConfig` attribute to `true`:

   ```
   <FederationConfig useLocalConfig="true">
   ```

3. Save the file and exit.
4. Open the `$ORACLE_HOME/fed/conf/cot.xml` file.
5. Locate the XML `PeerProvider` element whose `providerID` attribute contains the Identifier of the remote provider for which the specific configuration needs to be set.
6. Locate the XML `Config` element, child of the `PeerProvider` element. Using this `Config` element, perform the steps described earlier in "Configuring the Name ID Formats as an IdP" and "Configuring the Name ID Formats as an SP" to configure the Name ID Formats. For example:

   ```
   <PeerProvider providerID="http://idp.com" ...>
   ...
      <Config>
         ...
         <propertyset name="nameidformats">
   ```

```
        <propertyvalue>x509</propertyvalue>   <!-- X.509 Subject Name -->
        <propertyvalue>emailaddress</propertyvalue>  <!-- Email Address -->
        </propertyset>
        <property name="nameformatx500">dn</property>
        <property name="nameformatemail">mail</property>
        <property name="defaultauthnrequestnameidformat">x509</property>
        ...
      </Config>
      ...
</PeerProvider>
```

7. Save the file and exit.

Notes:

- Navigating to the Server Properties -> Circle of Trust page, selecting the remote provider and clicking **Update**, making any (or no) changes on the new screen and clicking **Save** will remove any of the settings not defined in the screen; the Unspecified and Customizable Name ID Format configuration will be removed.
- Navigating to the Server Properties -> Circle of Trust page, selecting the remote provider and clicking **Update**, then clicking **Save** will reset the Default Authn Request NameID Format setting (if set) to a value defined in the screen; since the Unspecified and Customizable Name ID Format choice are not available on the Oracle Identity Federation Administration Console, it would set the Default Authn Request NameID Format to another value if it was previously set to one of these two settings.

8. If using RDBMS as the Oracle Identity Federation transient data store, restart OC4J_FED. Otherwise, click the **Refresh Server** on the Oracle Identity Federation Administration Console.

Configuring Attributes in SSO Assertions with Oracle Identity Federation/IdP

When Oracle Identity Federation is acting as the IdP, it can be configured to send attributes in SSO Assertions, on a per-provider basis and depending on the Name ID Format contained in the assertion (the administrator can configure Oracle Identity Federation so that it will include attributes in the SSO assertion only for specific Name ID Formats).

To configure attributes in SSO assertions for the X.509 Subject Name, Email Address, Windows Domain Qualified Name, Kerberos Principal Name, Persistent, and Transient Name ID Formats, follow the instructions in "Edit Trusted Provider: Attribute Mappings" on page 6-51.

The Unspecified and Customizable formats are not configurable through the Oracle Identity Federation Administration Console. Take these steps to configure these formats (and any other formats that must be enabled, if Unspecified and Customizable are configured):

1. Open the `$ORACLE_HOME/fed/conf/config.xml` file.
2. Locate the `FederationConfig` XML element, and set its `useLocalConfig` attribute to `true`:

```
<FederationConfig useLocalConfig="true">
```

3. Save the file and exit.
4. Open the `$ORACLE_HOME/fed/conf/cot.xml` file.
5. Locate the XML `PeerProvider` element whose `providerID` attribute contains the Identifier of the remote provider for which the specific configuration needs to be set.
6. Locate the XML `Config` element, child of the `PeerProvider` element, and look for the propertyset element named `sendattributefornameid`. This `propertyset` element will contain the list of Assertion Name ID Formats that indicate to Oracle Identity Federation that attributes need to be included in SSO Assertions containing this a Name ID Format. Add the desired Name ID Formats to the list, like this:

```
<PeerProvider providerID="http://idp.com" ...>
...
   <Config>
      ...
      <propertyset name="sendattributefornameid">
      <propertyvalue>x509</propertyvalue>   <!-- X.509 Subject Name -->
      <propertyvalue>emailaddress</propertyvalue>  <!-- Email Address -->
      <propertyvalue>windowsnamequalifer</propertyvalue> <!-- Windows Domain
Qualified Name -->
      <propertyvalue>kerberos</propertyvalue>   <!-- Kerberos Principal Name
-->
      <propertyvalue>persistent</propertyvalue>  <!-- Persistent Identifier -->
      <propertyvalue>transient</propertyvalue>  <!-- Transient Identifier -->
      <propertyvalue>unspecified</propertyvalue>  <!-- Unspecified -->
      <propertyvalue>custom</propertyvalue>   <!-- Customizable Name ID format
-->
      </propertyset>
      ...
   </Config>
   ...
</PeerProvider>
```

   The possible values are:

   - `x509` for X.509 Subject Name
   - `emailaddress` for Email Address
   - `windowsnamequalifer` for Windows Domain Qualified Name
   - `kerberos` for Kerberos Principal Name
   - `persistent` for Persistent Identifier
   - `transient` for Transient Identifier
   - `unspecified` for Unspecified
   - `custom` for Customizable Name ID
7. Save the file and exit.

> **Note:** Navigating to the Server Properties -> Circle of Trust page, selecting the remote provider and clicking **Update**, clicking the Attribute Mappings, making any (or no) changes on the new screen and clicking **Save** will remove any of the settings not defined in the screen; the Unspecified and Customizable Name ID Format configuration will be removed.

8. If using RDBMS as the Oracle Identity Federation transient data store, restart OC4J_FED. Otherwise, click the **Refresh Server** on the Oracle Identity Federation Administration Console.

How to Allow the IdP to Determine the Name ID Format

Typically, when performing a single sign-on operation, the service provider specifies the Name ID format for the identity provider to use when it sends the Assertion containing the user information; this is done by selecting the Default Authn Request NameID Format in the Service Provider configuration.

However, you can configure the service provider not to specify any Name ID format. To achieve this:

1. Log on to the Oracle Identity Federation administration console.
2. Go to **Server Configuration** - > **Service Provider**, then choose Liberty 1.2 or SAML 2.0 as appropriate.

> **Note:** Unspecified format is not supported for Liberty 1.1.

3. Select Unspecified as the Default Authn Request NameID Format.
4. Click **Save**, then **Refresh Server**.

When the IdP receives a single sign-on request from an SP without any requested Name ID Format, the IdP will use an existing federation to perform the single sign-on operation. If no federation exists, the IdP will create a new federation based on the default Assertion Name ID Format configured at the IdP. To configure this format:

1. Log on to the Oracle Identity Federation administration console.
2. Go to **Server Configuration** - >**Identity Provider**, then choose Liberty 1.2 or SAML 2.0 as appropriate.

> **Note:** Unspecified format is not supported for Liberty 1.1.

3. Click **Assertion NameID Formats**.
4. Select the Default Assertion NameID Format.
5. Click **Apply**, then **Refresh Server**.

How to Use Automatic Account Linking at the SP

This section explains automatic account linking at a service provider and how to use the feature. It contains these topics:

What is Automatic Account Linking at the SP?

Automatic account linking at the SP allows the service provider to directly map an identity contained in an assertion to a user. With this feature:

- a new federation can be created, when none exists for the user and the peer IdP, without having to prompt the user for credentials. This is achieved in conjunction with a federation data store.
- the Oracle Identity Federation server is not required to use a federation data store to persist federation records.

See Also: "Edit Federation Data Store" on page 6-64.

Automatic account linking is supported for SAML 2.0 SSO operations with non-opaque name identifiers, when enabled:

- X.509
- e-mail address
- Kerberos principal name
- Windows domain qualified name
- Unspecified
- Customizable Name ID Format

This feature is not supported for:

- Liberty 1.x SSO
- SAML 2.0 SSO operations using persistent and transient name identifiers

Make sure that federation creation is allowed when using this feature (see "Service Provider - Global Settings" on page 6-15).

Configuring Automatic Account Linking at the SP

Take these steps to configure automatic account linking at the SP:

1. Configure the Service Provider to request a non-opaque SAML 2.0 Name ID:
   - Go to the Oracle Identity Federation administration console.
   - Go to **Server Configuration** - > **Service Provider**, then SAML 2.0.
   - Select the Default Authn Request NameID Format to be one of: X.509, Email Address, Kerberos Principal Name or Windows Domain Qualified Name.

     Note: You can select Unspecified - in that case, the IdP will decide the Name ID format, which must be non-opaque.

   - Click **Apply**.
2. Map the selected Name ID format to a user attribute that uniquely reference a user record:
   - Go to **Server Configuration** - > **Service Provider**, then SAML 2.0.

- Click **Account Linking NameID Formats**.
- Check which Name ID formats are to be enabled, and the attribute name that will contain the user information for the specified name IDs.

  Note: `dn` references the DN of an LDAP entry.

- Click **Apply**.

3. Enable the automatic account linking feature:
   - Go to **Server Configuration** - > **Service Provider**, then SAML 2.0.
   - Check the **Auto Account Linking Enabled** property.
   - Click **Apply**, then **Refresh Server**.

How to Use Automatic Account Linking at the IdP

This section explains automatic account linking at an identity provider and how to use the feature. It contains these topics:

What is Automatic Account Linking at the IdP?

Automatic account linking at the identity provider allows the IdP to create a federation automatically, if no federation exists and if the SP did not request that one be created. With this feature:

- a new federation can be created, when none exists for the user and the peer SP, without the SP having to request it. This is achieved in conjunction with a federation data store.
- the Oracle Identity Federation server is not required to use a federation data store to persist federation records.

See Also: "Edit Federation Data Store" on page 6-64

Automatic account linking is supported for SAML 2.0 SSO operations with non-opaque name identifiers, when enabled:

- X.509
- e-mail address
- Kerberos principal name
- Windows domain qualified name
- Unspecified
- Customizable Name ID Format

This feature is not supported for:

- Liberty 1.x SSO
- SAML 2.0 SSO operations using persistent and transient name identifiers

Configuring Automatic Account Linking at the IdP

Take these steps to configure the IdP for automatic account linking:

1. Specify the Name ID format mappings:
   - Go to **Server Configuration** - > **Identity Provider**, then SAML 2.0.
   - Click Assertion NameID Formats.
   - Check which Name ID formats are to be enabled, and the attribute name that will contain the user information for the specified name IDs.

     Note: dn references the DN of an LDAP entry.

   - Click **Apply**.
2. Optionally, configure the Default Assertion Name ID Format to use a non-opaque Name ID.
3. Enable the automatic account linking feature:
   - Go to **Server Configuration** - > **Identity Provider**, then SAML 2.0.
   - Check the **Auto Account Linking Enabled** property.
   - Click **Apply**, then **Refresh Server**.

Interoperating with Microsoft ADFS

Both Microsoft Active Directory Federation Services (ADFS) and Oracle Identity Federation can perform either Identity Provider (IdP) or Service Provider (SP) roles in a federated environment.

This section explains how to configure these products to work with each other. It contains the following topics:

Terms and Definitions

For reference, here are the definitions of some comparative federation terms used in Oracle Identity Federation and Microsoft Active Directory Federation Services (ADFS).

Identity Provider/Account Partner/IdP

An identity provider/account partner is responsible for managing, authenticating, and asserting a set of identities within a given circle of trust.

Identity providers are service providers offering business incentives so that other service providers affiliate with them.

Service Provider/Resource Partner/SP

A service provider/resource partner provides services or goods to a Principal while relying on an Identity Provider/account provider to authenticate the Principal's identity.

Configuring ADFS as IdP with Oracle Identity Federation as SP

This section describes the steps needed to configure Oracle Identity Federation so that it is integrated with Active Directory Federation Services (ADFS) as IdP. The configuration consists of two nodes:

- Node A has Oracle Identity Federation installed as a Service Provider (SP) type server.
- Node B has ADFS installed as an identity provider (IdP) type server.

Configuration topics covered in this section include:

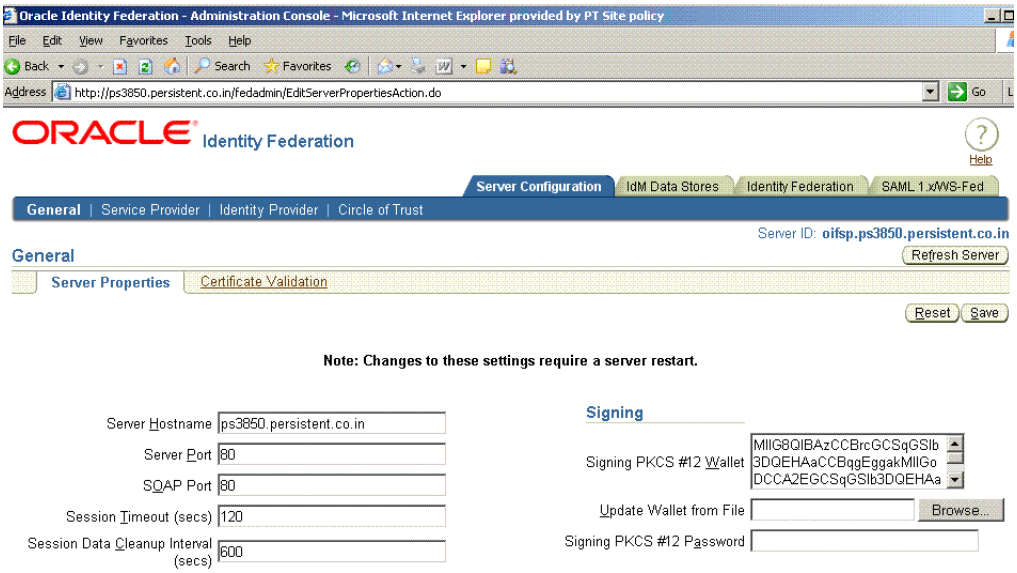
Prerequisites

Prior to performing this configuration, ensure that Oracle Identity Federation is already installed at Node A, and is configured as a service provider. Also ensure that ADFS is installed at Node B, and is configured as an identity provider.

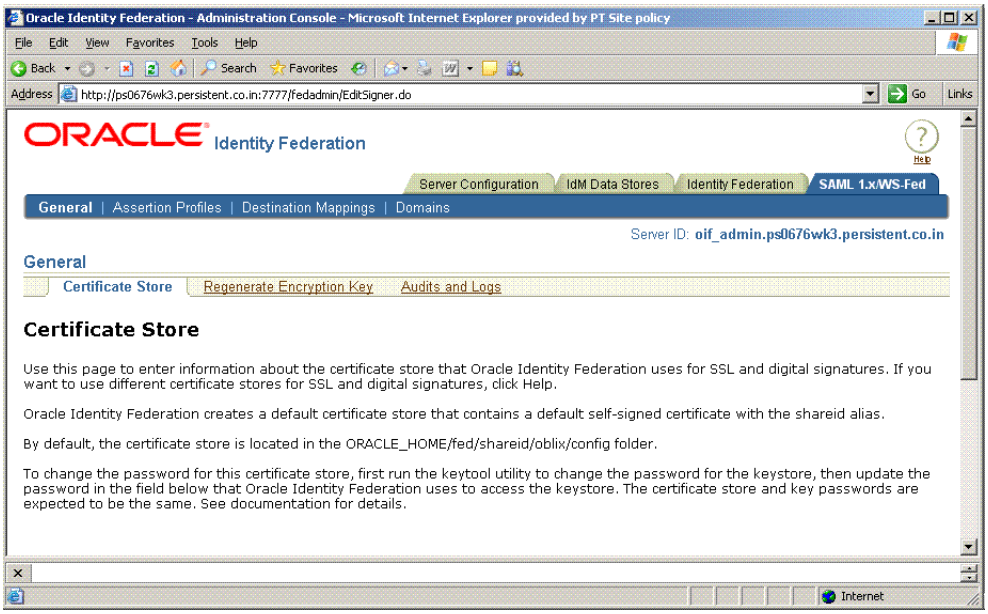
Collect Information from Oracle Identity Federation

Take these steps at node A where Oracle Identity Federation is installed:

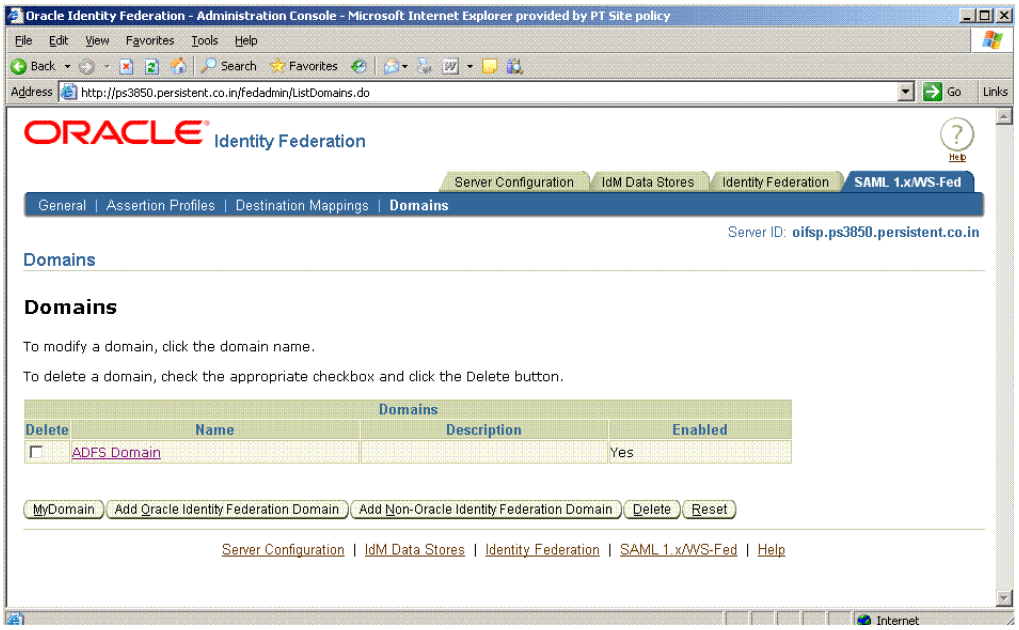
1. Log in to Oracle Identity Federation as the `oif_admin` user.



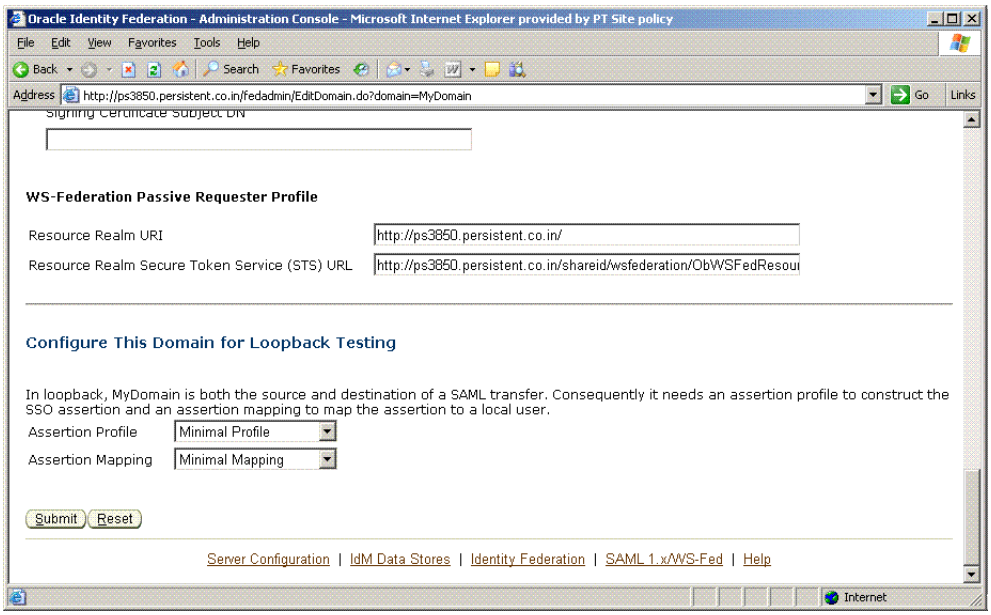
2. Click **SAML 1.x/WS-Fed**.



3. Select the **Domains** tab.



4. Click **MyDomain**.



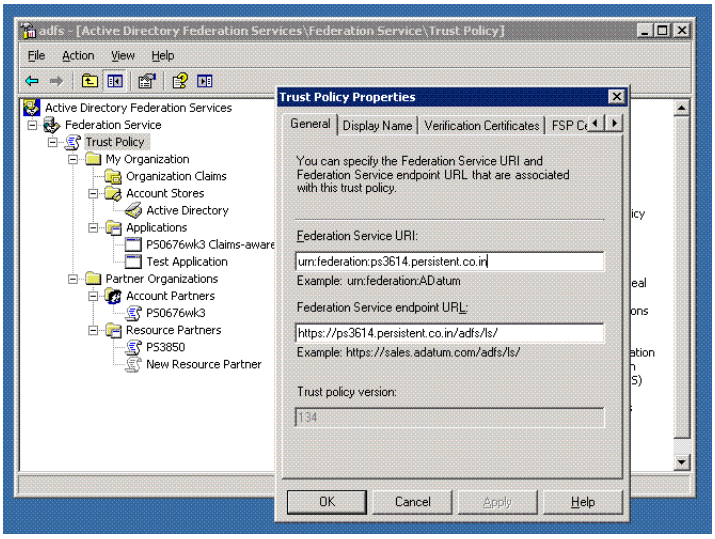
Collect this information:

- **Resource Realm URI** - this URI identifies the SP to the IdP.
- **Resource Realm Secure Token Service (STS) URL** - this is the URL to which the IdP redirects the user along with the security token.

Collect Information from ADFS

Collect this information on node B where ADFS is installed:

1. Navigate to **Start -> All Programs -> Administrative Tools -> Active Directory Federation Services**.
2. In the console tree, double-click **Federation Service**, right-click **Trust Policy**, and click **Properties**.

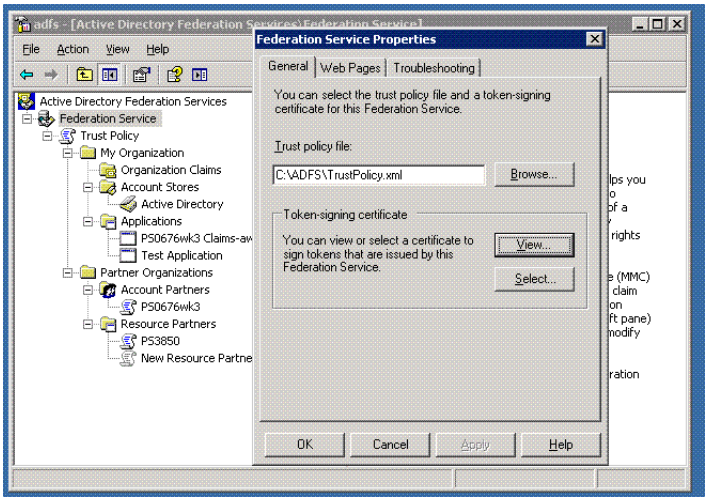


3. Collect this information:
   - **Federation Service URI** - identifies the IdP to the SP.

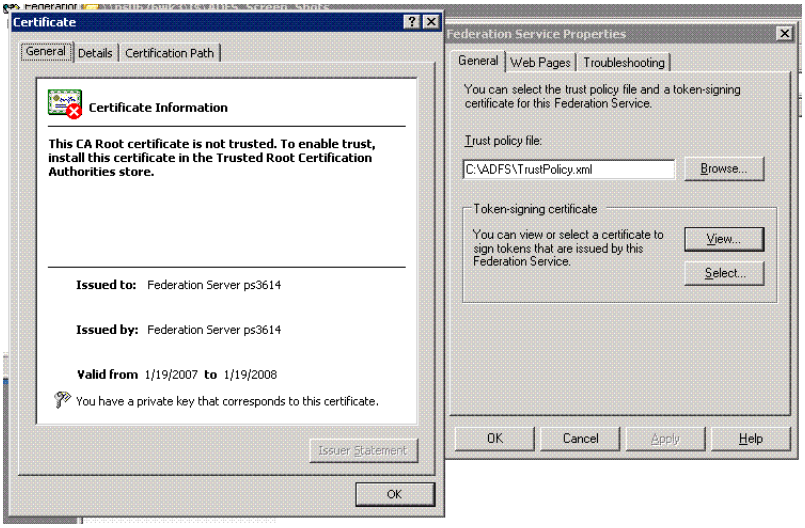
- **Federation Service endpoint URL** - this is the IdP URL to which the SP redirects the user to obtain a security token.

Export the token-signing certificate Servers that are running the Federation Service component of ADFS in an account federation service require token-signing certificates to sign security tokens that the servers produce. In this step, you export the token-signing certificate from ADFS node B to a file. Take these steps to export the certificate:

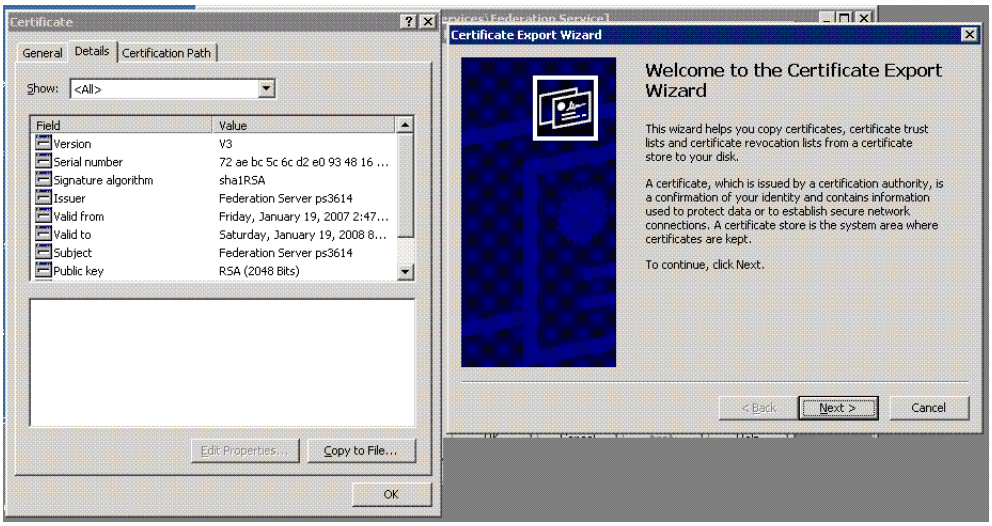
1. Navigate to **Start -> All Programs -> Administrative Tools -> Active Directory Federation Services**.
2. Right-click **Federation Service**, and click **Properties**.



3. Click **View** to review the certificate.



4. On the **Details** tab, click **Copy to File**.



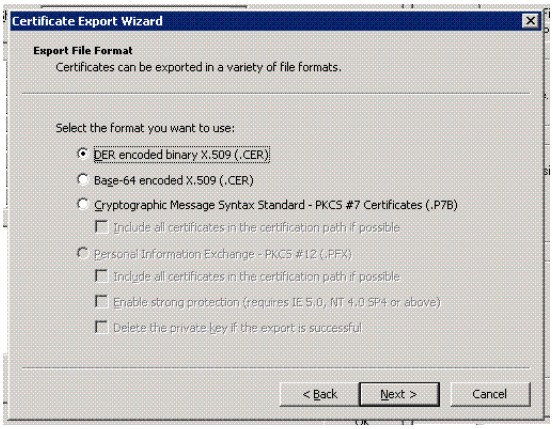
The **Welcome to the Certificate Export Wizard** page appears.

5. Click **Next**.

The **Export Private Key** page appears.

6. Select **No, do not export the private key**, then click **Next**.

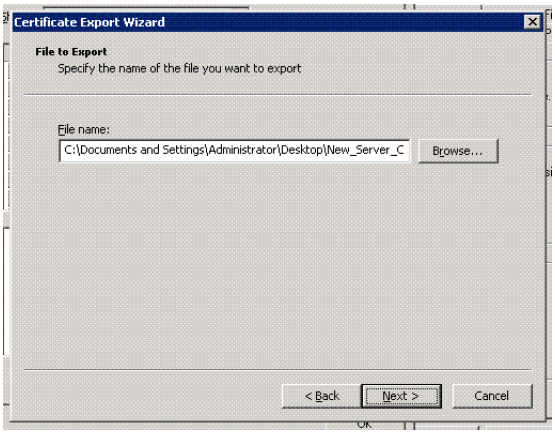
7. The **Export File Format** page appears.



Select **DER encoded binary X.509 (.Cer)** and click **Next**.

Note: DER encoded Binary X.509 and Base-64 encoded X.509 certificates use `.cer` as the filename extension. Cryptographic Message Syntax Standard - PKCS #7 certificates use `.P7B` as the filename extension.

8. The **File to Export** page appears.



Choose a directory and filename using the format *`File_Name.cer`*, and click **Next**.

Review the certificate export options you specified with the wizard, and click **Finish**.

Configure Oracle Identity Federation as Service Provider

This section explains how to configure a service provider instance of Oracle Identity Federation to recognize ADFS as an identity provider.

Import IdP's token signing certificate to the SP's keystore Take these steps to import the ADFS identity provider's token signing certificate to the SP's keystore:

1. Copy the certificate file you collected in"Export the token-signing certificate" on page 7-32 to a location that can be accessed by the service provider.
2. At the service provider, import the IdP's token signing certificate into the keystore:

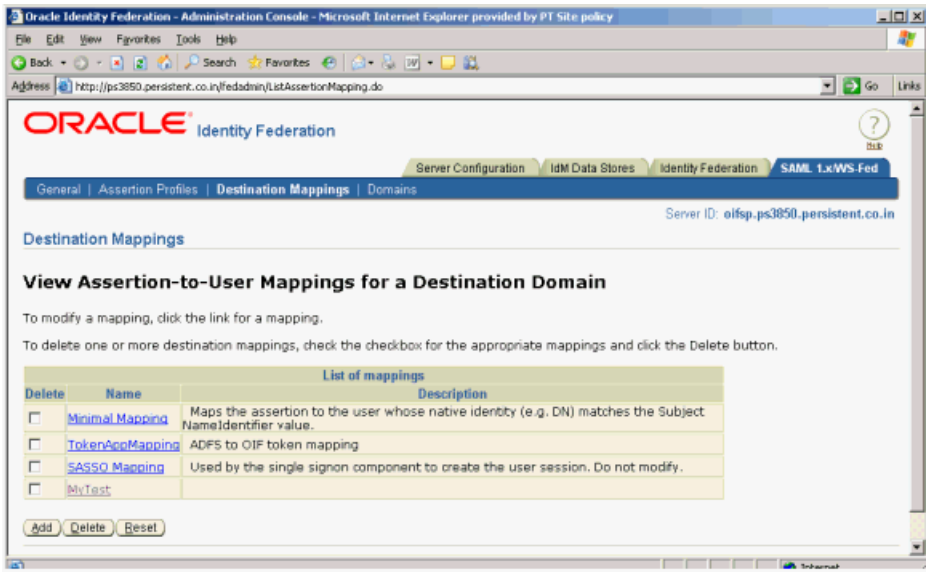
   ```
   C:\OIF Home\jdk\bin>keytool -keystore
   C:\OIF Home\fed\shareid\oblix\config\keystore -storepass password
   -import -alias anyName -file myfile
   ```

   where the parameters are:

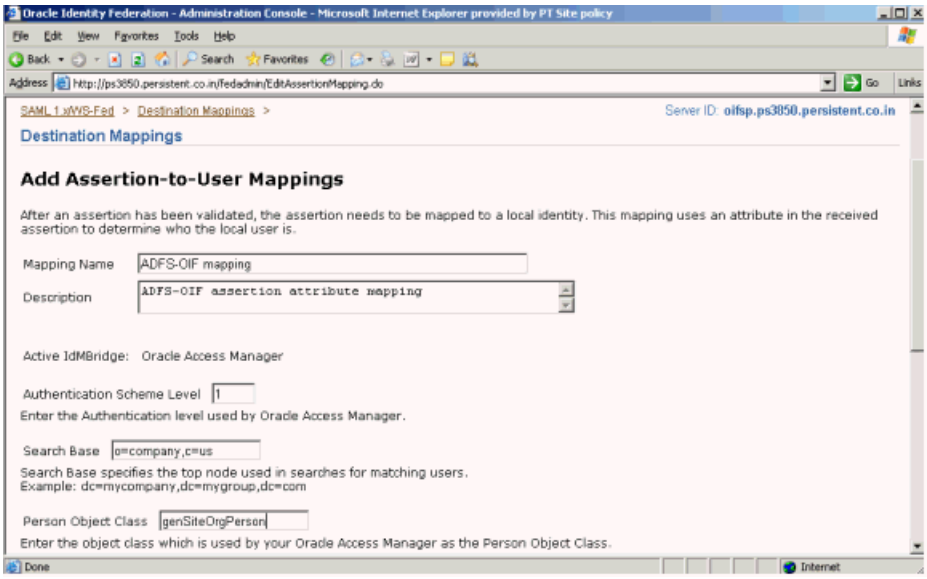
   - password - the password of the `oif_admin` user
   - anyName - unique name for alias
   - myfile - certificate files in the `File_Name.cer` format collected earlier
3. Restart the Oracle Identity Federation SP instance from the Oracle Enterprise Manager console.

Create Assertion-to-User Mappings Take these steps to add assertion-to-user mappings in Oracle Identity Federation:

1. Log in to Oracle Identity Federation as the `oif_admin` user.
2. Click **SAML 1.x/WS-Fed** -> **Destination Mappings**
3. The current assertion-to-user mappings are displayed.



4. Click **Add** to add new Assertion-to-User Mappings.



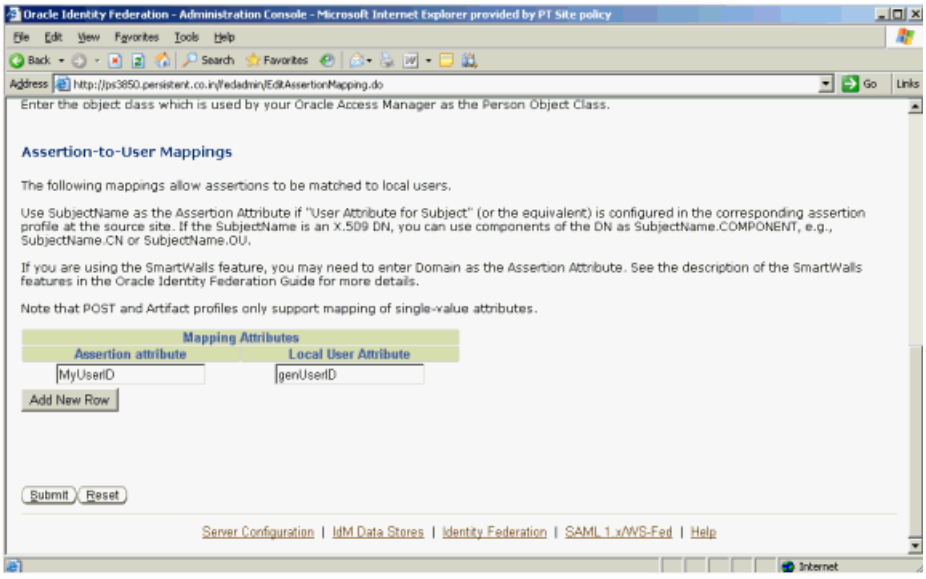
Add this information:

- Mapping Name - user-defined name; enter any desired name.
- Description - enter a brief description of the mapping.
- Authentication Scheme Level - enter the authentication level used by Oracle Access Manager.
- Search Base - enter the LDAP directory information tree (DIT) node from which Oracle Identity Federation should start the search to locate the user; for example, `o=company, c=us`.
- Person Object Class - enter the object class that Oracle Access Manager uses as the person object class, for example, `gensiteorgperson`.

Note: Authentication Scheme Level, Search Base, and Person Object Class do not have default values. You will need to supply this data.

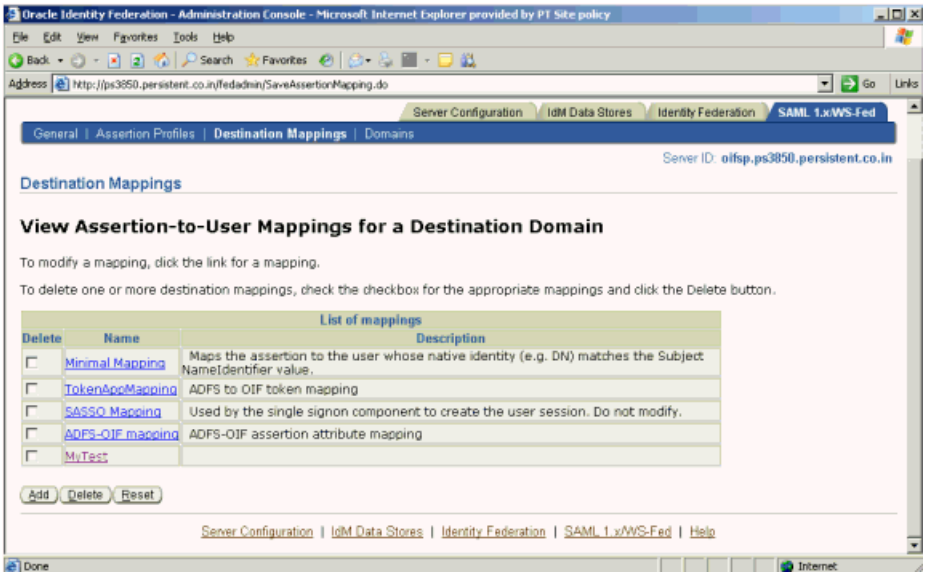
---

5. Scroll down to add local user attribute mapping.



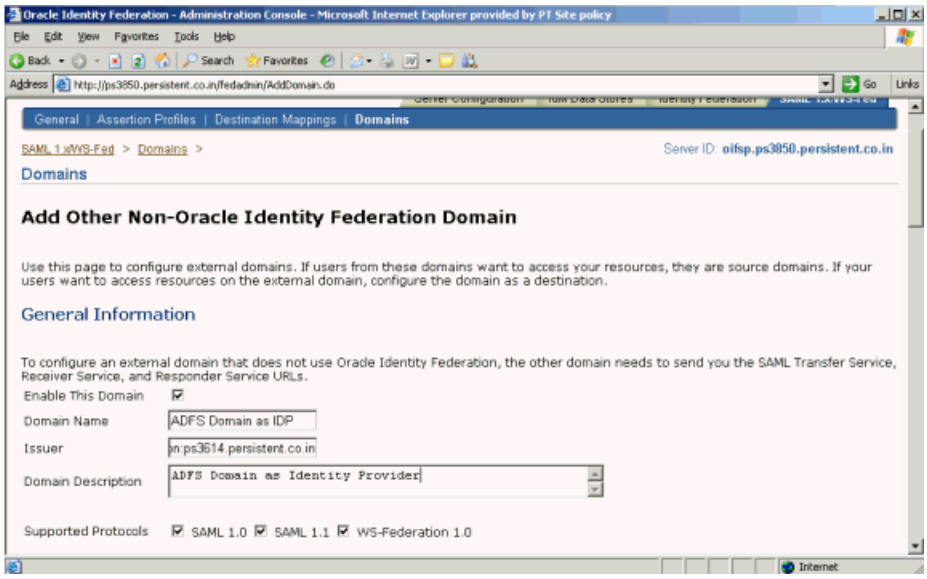
Map the assertion attribute to a local user attribute to identify the local user. To add more attributes, click **Add New Row**.

6. Click **Submit**. Check the destination mapping list to verify that your new mapping was created.



Create Non-Oracle Identity Federation Domain Take these steps to create a non-Oracle Identity Federation domain at the Oracle Identity Federation server for ADFS:

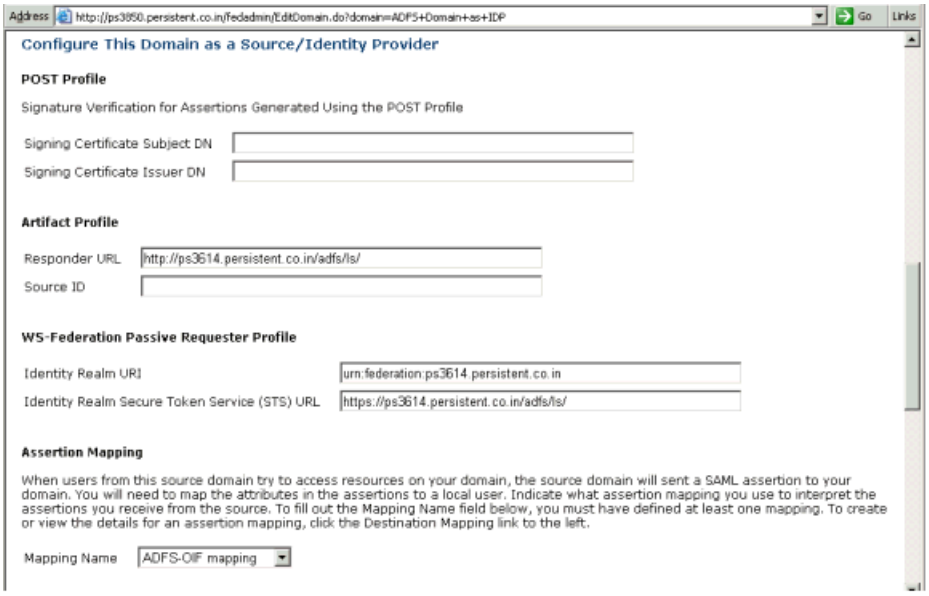
1. Log in to the Oracle Identity Federation server as the `oif_admin` user.
2. Click the **SAML 1.x/WS-Fed** tab, and navigate to **Domains**-> **Add Non-Oracle Identity Federation Domain**.
3. Click **Add Non-Oracle Identity Federation Domain**.



4. Add this information:
   - Domain Name
   - Domain Description
   - Issuer - Enter the Federation Service URI collected in "Prerequisites" on page 7-29.

   Check the **Enable This Domain** box.

   Check the required **Supported Protocols**.
5. Scroll down to configure this domain as an IdP.



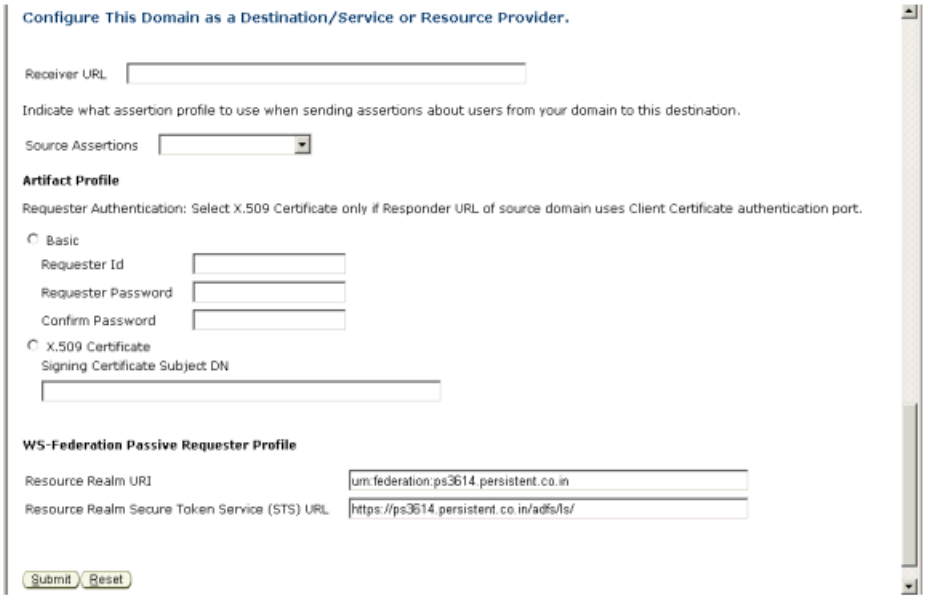
Provide this information:

- Responder URL - Enter the Federation Service endpoint URL collected in "Prerequisites" on page 7-29.

Note: Oracle Identity Federation generates a Source ID based on this Responder URL. To change the SourceID, you change the Responder URL, delete the information in the Source ID field and click Submit to regenerate the source ID. Do not manually re-generate the SourceID.

- Identity Realm URI - Enter the "Federation Service URI" collected in the "Prerequisites".
- Identity Realm Secure Token Service (STS) URL - Enter the Federation Service endpoint URL collected in the "Prerequisites".
- Mapping Name - Select the mapping created earlier in "Create Assertion-to-User Mappings" on page 7-35.

6. Scroll down to configure the Resource Realm URI and Resource Realm Secure Token Service (STS) URL.



Provide this information:

- Resource Realm URI - Enter the "Federation Service URI" collected in the "Prerequisites".
- Resource Realm Secure Token Service (STS) URL - Enter the Federation Service endpoint URL collected in the "Prerequisites".

7. Click **Submit** to submit the configuration.
8. Check the Domain list to ensure that the domain was created and is enabled.

Configure ADFS to recognize Oracle Identity Federation as SP

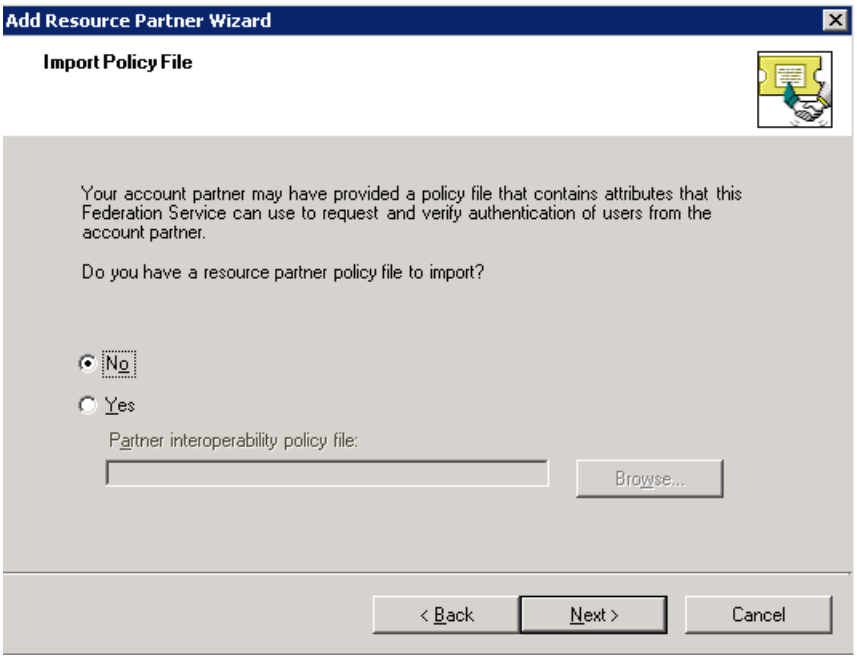
Take these steps to configure the ADFS Identity Provider to recognize Oracle Identity Federation as a service provider, by adding a Resource Partner:

1. Navigate to **All Programs** -> **Administrative Tools** -> **Active Directory Federation Services**.

2. Double-click **Federation Service**, double-click **Trust Policy**, double-click **Partner Organizations**, and right-click **Resource Partners**. Point to **New**, and click **Resource Partner**.
3. The Welcome to the Add Resource Partner Wizard page appears.

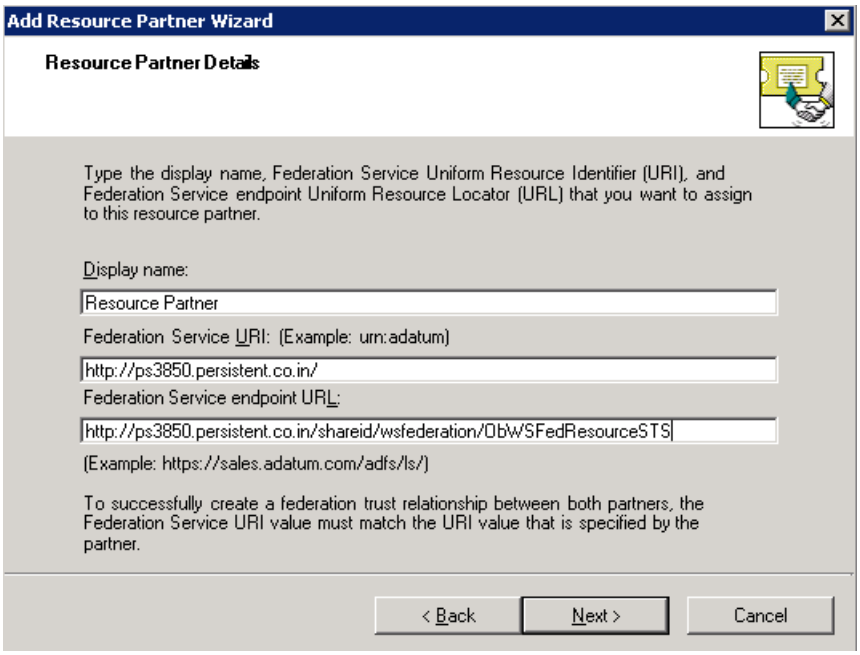
Click **Next**.

4. The Import Policy File page appears.



Select No for the question, and click **Next**.

5. The Resource Partner Details page appears.

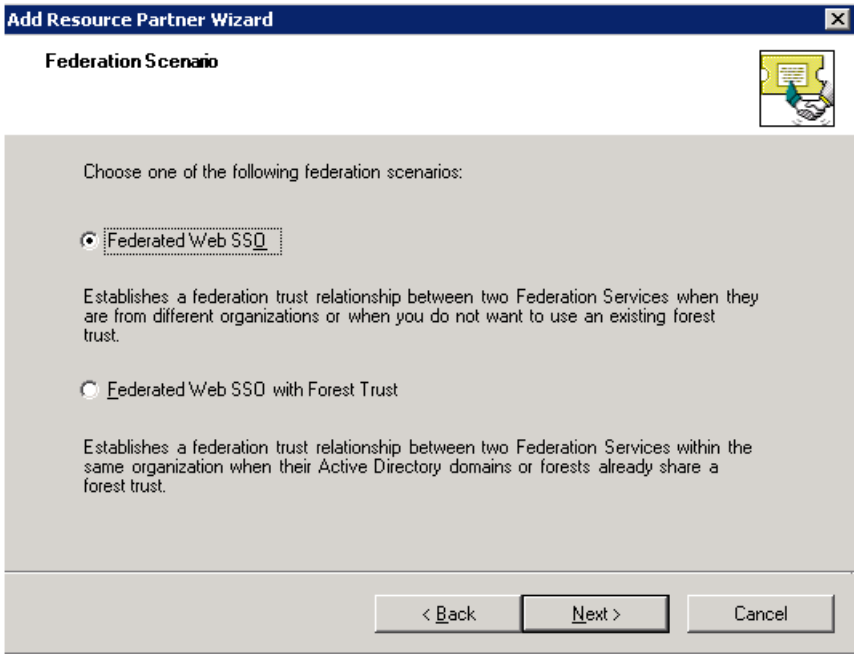


Provide this information:

- Display name – Enter "Resource Partner".
- Federation Service URI – Enter the Resource Realm URI collected in "Prerequisites" on page 7-29.
- Federation Service endpoint URL – Enter the Resource Realm Secure Token Service (STS) URL collected in "Prerequisites".

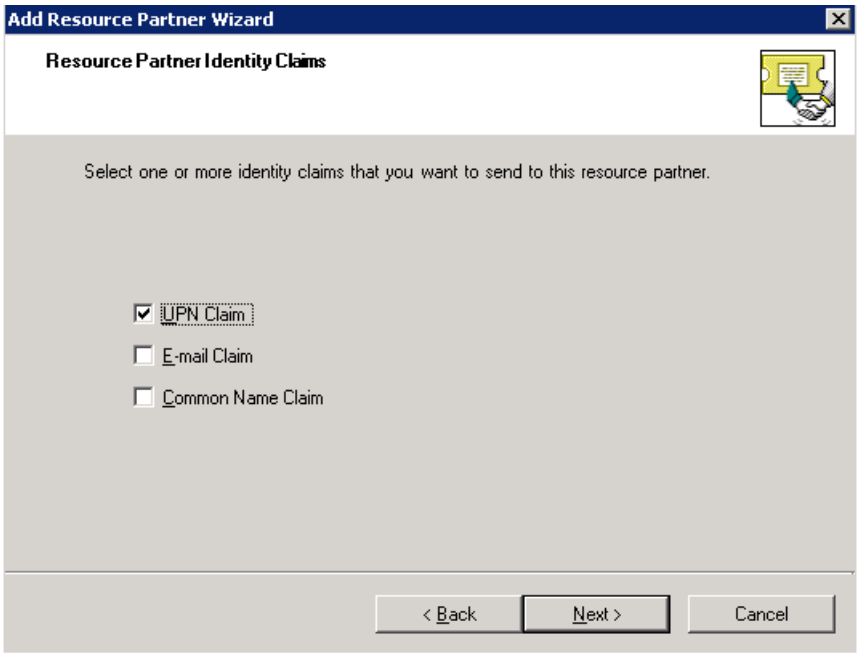
Click **Next**.

6. The Federation Scenario page appears.



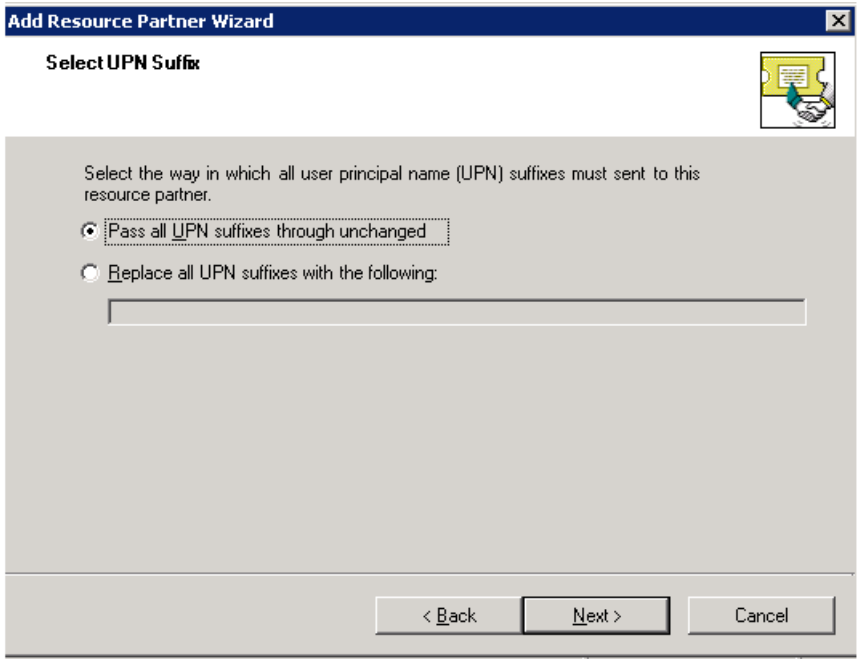
Select Federated Web SSO, and click **Next**.

7. The Resource Partner Identity Claims page appears.



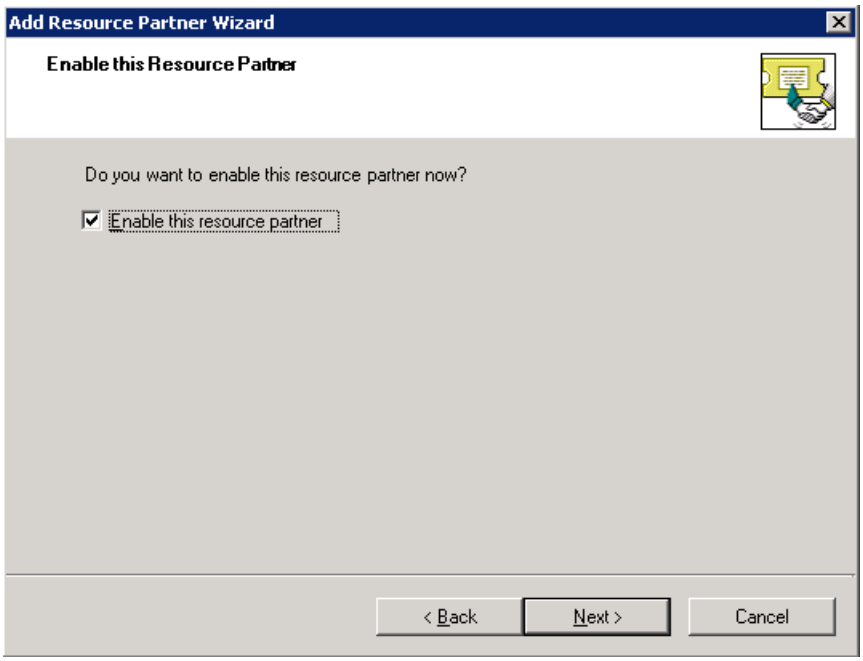
Select the UPN Claim check box and click **Next**.

8. The Select UPN Suffix page appears.



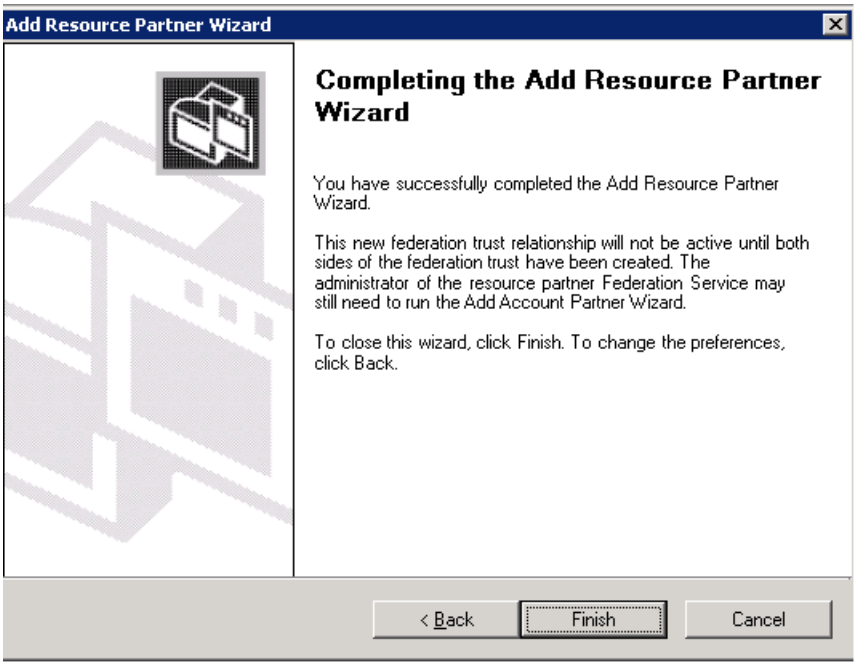
Select Pass all UPN domain suffixes through unchanged, and click **Next**.

9. The Enable this Resource Partner page appears.



Check the Enable this resource partner box and click **Next**.

10. The wizard completion page appears.



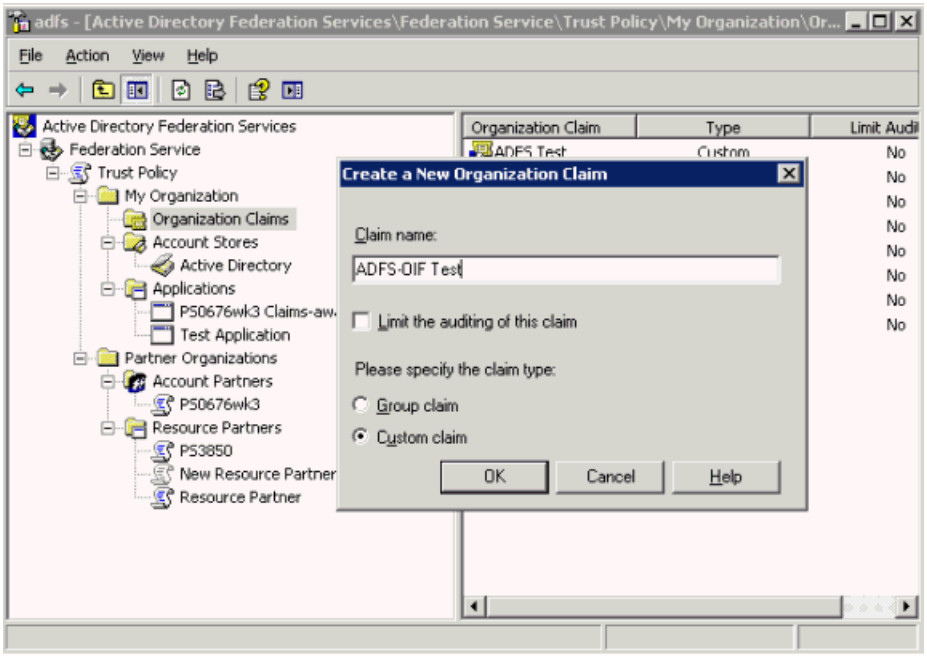
Verify your settings for the new resource partner and click **Finish**.

Configure claims

To configure claims, you must create a custom claim, map custom claim extraction to that claim, and create an outgoing custom claim mapping.

Create Organization Custom Claim Take these steps to create an Organization Custom Claim:

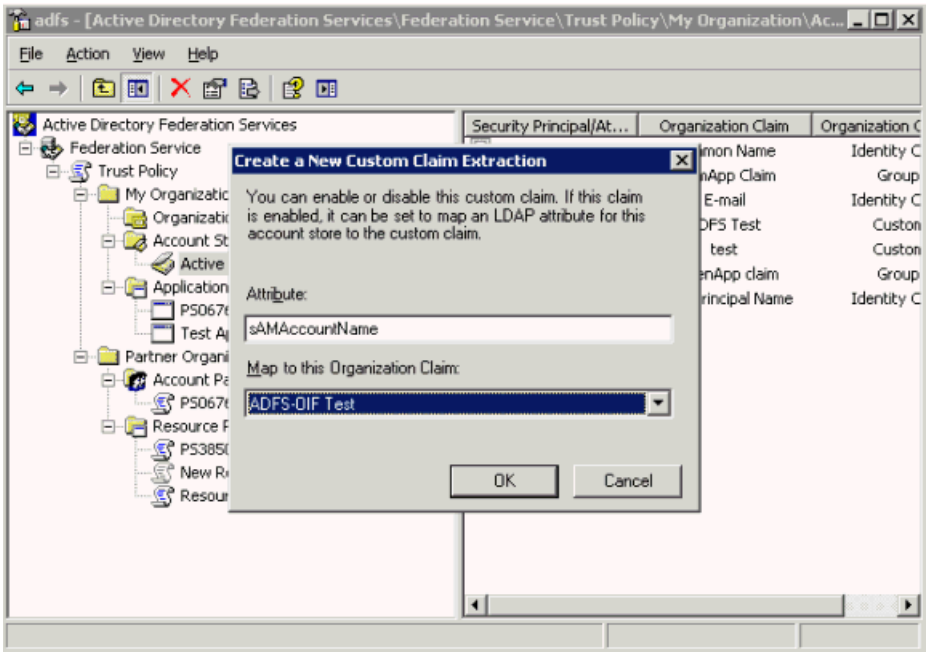
1. Navigate to **Start** -> **All Programs** -> **Administrative Tools** -> **Active Directory Federation Services**.
2. Double-click **Federation Service**, double-click **Trust Policy**, double-click **My Organization**, right-click **Organization Claims**; point to **New**, then click **Organization Claim**.



3. In the Create a New Organization Claim dialog box, in **Claim name**, type ADFS-OIF Test, Select **Custom Claim**, then click **OK**.

Map Custom Claim Extraction to Organization Custom Claim Take these steps to create a custom claim extraction:

1. Navigate to **Start** -> **All Programs** -> **Administrative Tools** -> **Active Directory Federation Services**.
2. Double-click **Federation Service**, double-click **Trust Policy**, double-click **My Organization**, double-click **Account Stores**, right-click **Active Directory**; point to **New**, and click **Custom Claim Extraction**.



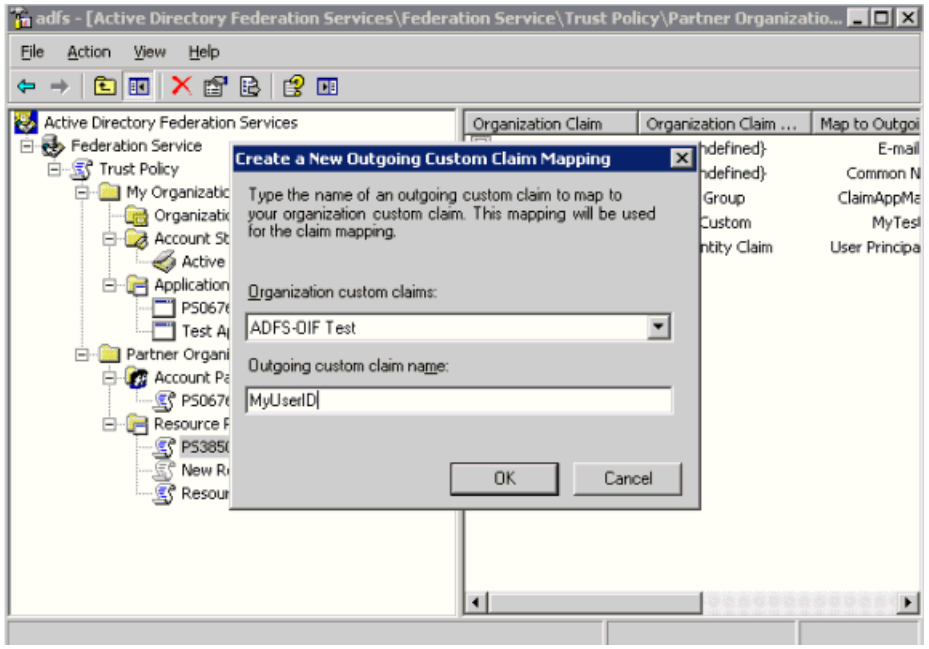
In the dialog box, select ADFS-OIF Test for **Map to this Organization Claim**.

Enter the attribute name to which you would like to map the assertion attribute.

3. Click **OK**.

Create an Outgoing Custom Claim Mapping Take these steps to create an outgoing custom claim to map to your organization custom claim:

1. Navigate to **Start** -> **All Programs** -> **Administrative Tools** -> **Active Directory Federation Services**.
2. Double-click **Federation Service**, double-click **Trust Policy**, double-click **Partner Organizations**, double-click **Resource Partners**, right-click **Resource Partner**; point to **New**, and click **Outgoing Custom Claim Mapping**.



Provide this information:

- Organization custom Claims – enter ADFS-OIF Test.
- Outgoing custom claim name – enter MyUserID. (This attribute is collected from the administrator of the destination domain.)

3. Click **OK**.

IdP-initiated SSO with WS-Federation

When ADFS is enabled for single sign-on (SSO) with WS-Federation, an SSO request can be initiated from the IdP, using a URL in the format `https://adfs_host/adfs/ls/`.

The parameters are:

- `wa` - specifies the action to perform. By including the action, URIs can be overloaded to perform multiple functions. For sign-in, this string MUST be "wsignin1.0".
- `wtrealm` - the request realm URI defined by the SP
- `wctx` - the protected resource URL to be accessed, at the SP's domain
- `wreply` - optional parameter; the URL to which responses are directed

For example:

```
https://adfs_host/adfs/ls/?wa=wsignin1.0&wtrealm=http://oif_host:oif_port
/&wctx=<protected_resource_url>&wreply=http://oif_host:oif_port
/shareid/wsfederation/ObWSFedResourceSTS
```

SP-initiated SSO with WS-Federation

When Oracle Identity Federation is enabled for SSO with WS-Federation, it is possible to initiate an SSO request from the SP, using a URL in the following format:

```
http(s)://oif_host:oif_port/shareid/wsfederation/ObWSFedResourceSTS
```

with these parameters:

- wa - This required parameter specifies the action to be performed. By including the action, URIs can be overloaded to perform multiple functions. For sign-in, this string MUST be "wsignin1.0".
- wctx - the protected resource URL to be accessed

For example:

```
http(s)://oif_host:oif_
port/shareid/wsfederation/ObWSFedResourceSTS?wa=wsignin1.0&wctx=<protected
resource url>
```

IdP-initiated Logout with WS-Federation

The URL for IdP-initiated logout with WS-Federation is:

```
https://adfs_host/adfs/ls/?wa=wsignout1.0
```

SP-initiated Logout with WS-Federation

The URL for SP-initiated logout with WS-Federation is:

```
http(s)://oif_host:
```

```
oif_port/shareid/wsfederation/ObWSFedResourceSTS?wa=wsignout1.0
```

wa is a required parameter that specifies the action to perform. By including the action, URIs can be overloaded to perform multiple functions. For logout, this string MUST be "wsignout1.0".

Configuring ADFS as SP with Oracle Identity Federation as an IdP

This section explains how to configure Microsoft Active Directory Federation Services (ADFS) as a service provider with Oracle Identity Federation acting as an identity provider. The steps illustrate a configuration scenario consisting of two nodes:

- Node A for Oracle Identity Federation
- Node B for ADFS

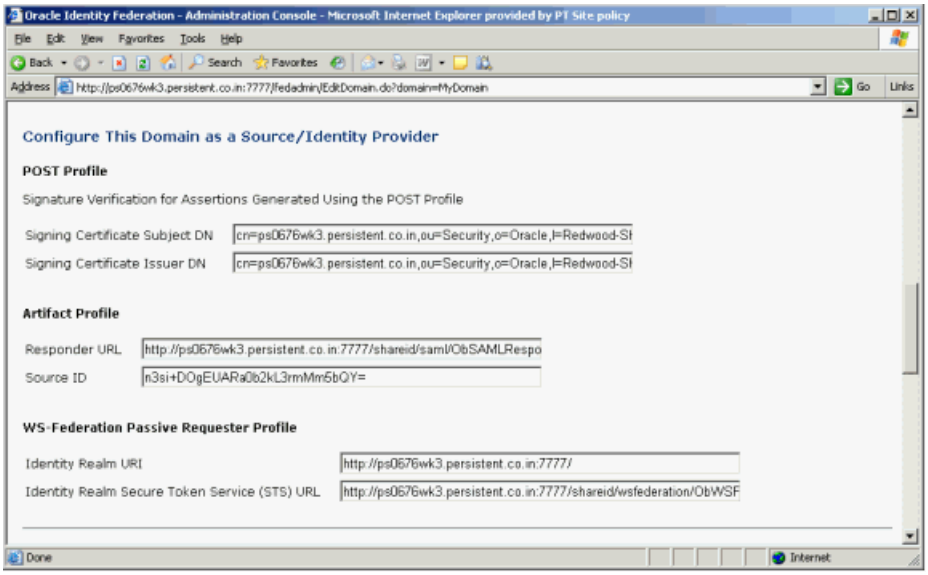
Prerequisites

Ensure that Oracle Identity Federation is installed as an identity provider (IdP) type server, and ADFS is installed as a service provider (SP) type server.

Collect Information from Oracle Identity Federation

Go to node A, where Oracle Identity Federation is installed, and collect this information:

1. Log in to Oracle Identity Federation as the `oif_admin` user.
2. Navigate to **SAML 1.x/WS-Fed** -> **Domains** -> **MyDomain**.
3. The **MyDomain** page appears.



Collect this information:

- Identity Realm URI - identifies the IdP to the SP.
- Identity Realm Secure Token Service (STS) URL - this is the URL to which the SP redirects the user to the IdP to obtain a security token.

Export the IdP's self-signed certificate to the SP Take these steps to export the identity provider's signing certificate from the Oracle Identity Federation SAML1.x/WS-Fed keystore. This procedure is used in SAML 1.x and WS-Federation configurations.

1. At the Identity Provider, export the self-signed certificate from the keystore by running the `keytool` utility in the Oracle Identity Federation home directory:

   ```
   C:\<OIF Home>\jdk\bin>keytool
   -keystore C:\OIF_Home\fed\shareid\oblix\config\keystore
   -storepass password
   -export
   -alias shareid
   -file myfile
   ```

   The parameters are:

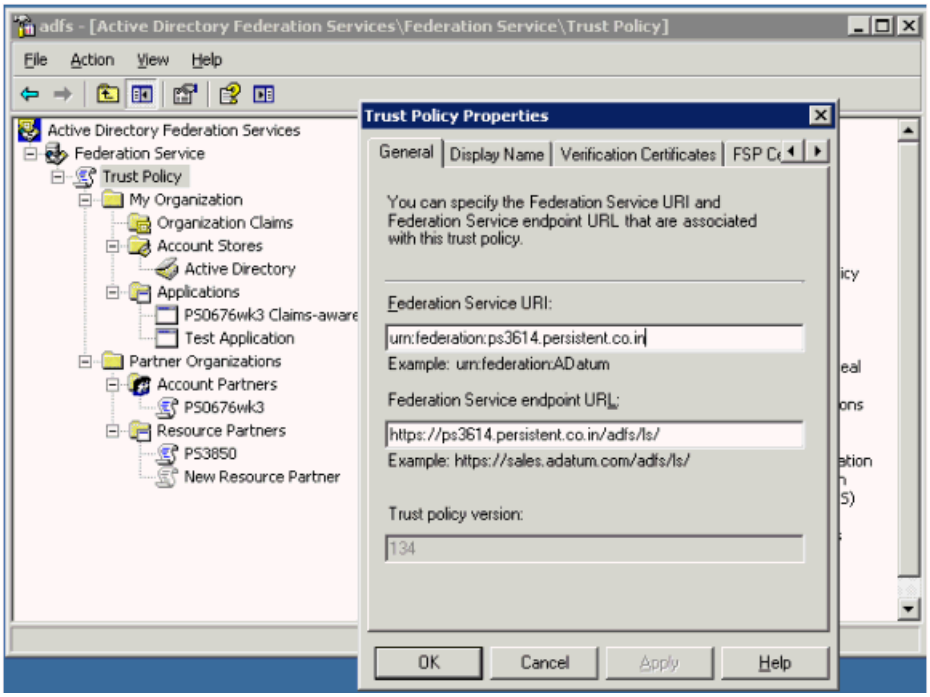
   - *password* - the password of the `oif_admin` user
   - *myfile* - A filename in the format `File_Name.cer` to store the certificate.

2. Copy this certificate file to a location that can be accessed by the service provider.

Collect information from ADFS

Go to the node B where ADFS is installed and collect the following information:

1. Navigate to **Start** -> **All Programs** -> **Administrative Tools** -> **Active Directory Federation Services**.
2. In the console tree, double-click **Federation Service**, right-click **Trust Policy**, and click **Properties**.



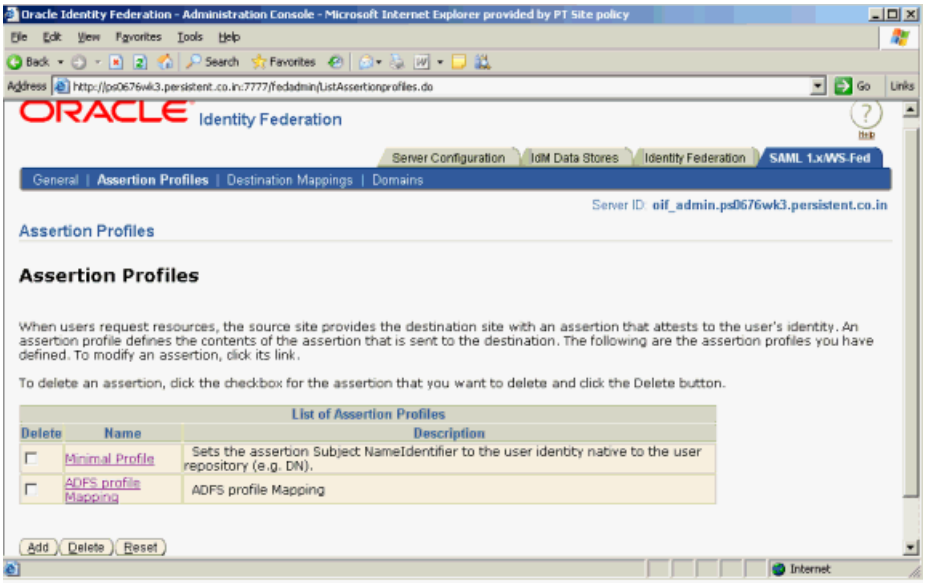
3. Collect this information:
   - Federation Service URI - URI which identifies the SP to the IdP.
   - Federation Service endpoint URL - the SP URL to which the IdP redirects the user along with the security token.

Configure Oracle Identity Federation to recognize ADFS as SP

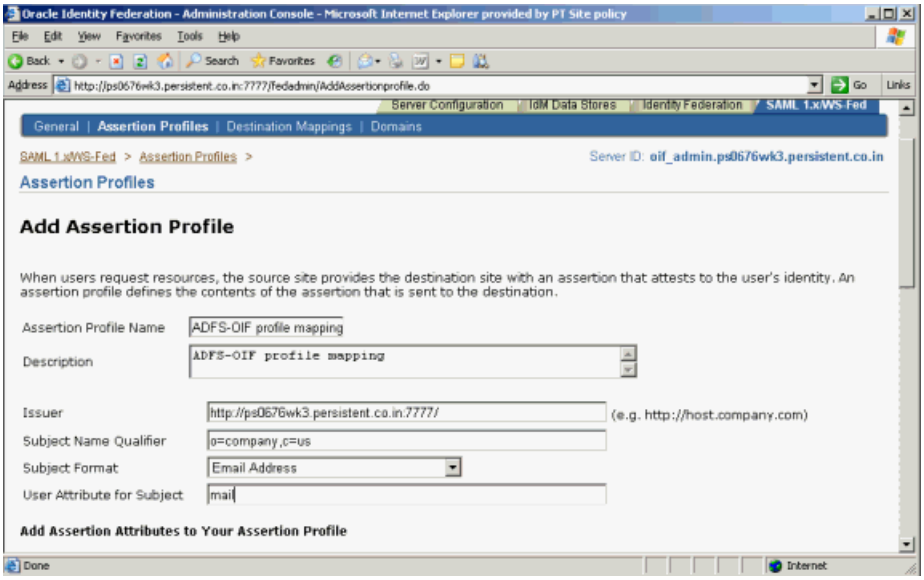
Take these steps to configure Oracle Identity Federation (acting as an identity provider) to recognize ADFS as a service provider.

Create Assertion Profile When users request resources, the source site (IdP) provides the destination site with an assertion that attests to the user's identity. Take these steps to define an assertion profile, which defines the contents of the assertion that is to be sent to the destination.

1. Log in to Oracle Identity Federation as the `oif_admin` user.
2. Navigate to **SAML 1.x/WS-Fed** -> **Assertion Profile**.



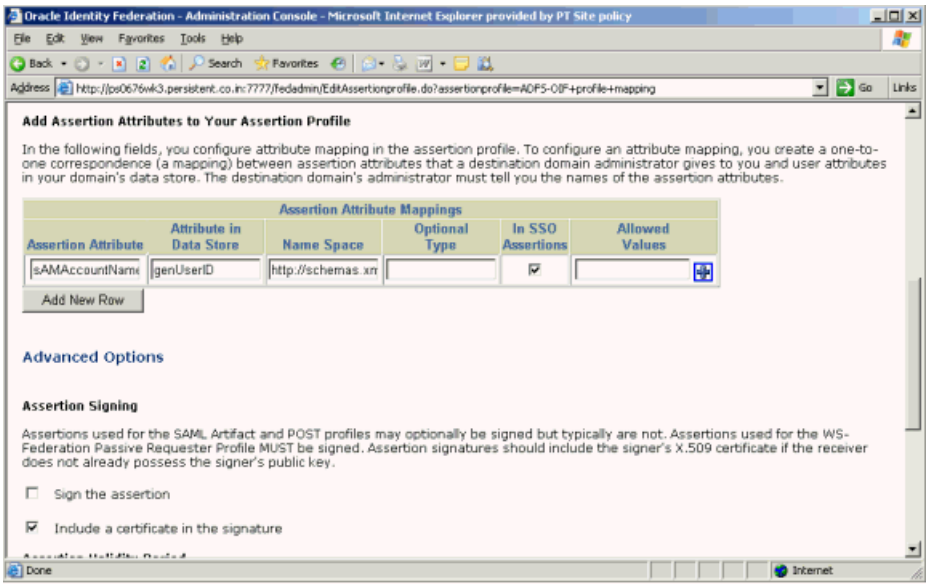
3. Click **Add** to add a new assertion.



4. Add this information:
   - Assertion Profile Name

- Description
- Issuer – for example, `http://oif_host:oif_port/`
- Subject Name Qualifier – This optional subject name qualifier allows the SAML-compliant server to determine the namespace for a user. This information may be useful to the source domain for informational purposes. Thus, an assertion for jsmith@oblix.com, may contain J. Smith's department. For example: `ou=Department,o=Company,c=US`
- Subject Format – This is the format of the Subject who is the user being identified by the assertion. The subject appears in an assertion as the `NameIdentifier` sub-element of the assertion Subject element. Choose the format from this list of attribute formats:
  - None - No format is specified.
  - Email address - The `NameIdentifier` is formatted as an email address.
  - X509 subject name - The `NameIdentifier` is in the form of a DN, for example, `cn=myname,dc=example,dc=com`.
  - Windows domain - A Windows domain qualified user name formatted as DomainName\UserName; for example, oblix\jsmith.
  - Unspecified - The contents of the NameQualifier sub-element of the assertion subject is unspecified, and it is up to the individual implementation to determine how to interpret this data. The value is set to "`urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified`" in the `shareid-config.xml` file. The Format value is used in the assertion. The SAML specification defines an unspecified format as a URI value for the Format attribute.
  - Other-A format value that is not one of the predefined values defined in the SAML specification. A format of Other allows any other URI values that SAML partners might use. Examples include: "<http://company.com/saml/nameid-format/company-id>" "urn:company.com:saml:nameid-format:company-id". These formats assume that company.com has registered its URN path. The value you specify in this field is set in the shareid-config.xml file.
- User Attribute for Subject - This is the local LDAP user attribute whose value will be used as the value for the Subject assertion element. To be more specific, this field determines the value of the `NameIdentifier` sub-element of the assertion's Subject element. The Subject element identifies the source domain user. If this field is left blank, the DN of the user's directory entry is used.

5. Scroll down to add more attributes to the assertion profile:



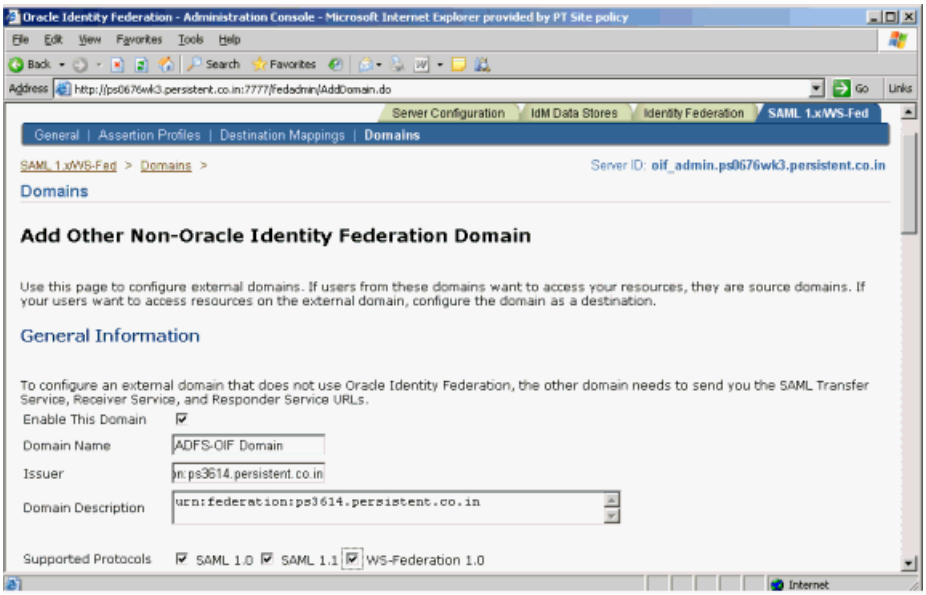
Provide this data:

- Assertion Attribute - Use this field to map assertion attributes from a peer site to your data store's user attributes.
- Attribute in Data Store - Enter local data store attribute.
- Name Space - Specify namespaces and types if other SAML-aware products or applications are going to use the namespaces or types.
- In SSO Assertions - Specify if this attribute is used in SSO assertions.

6. Click **Submit** to add the assertion profile.

Create Non-Oracle Identity Federation Domain for ADFS Take these steps in the Oracle Identity Federation server to add a non- Oracle Identity Federation domain for Microsoft ADFS:

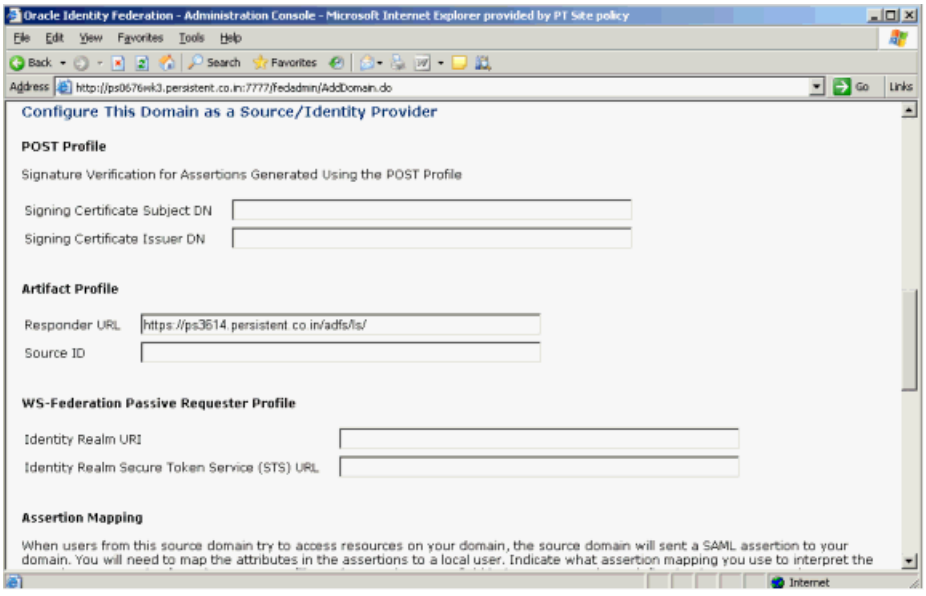
1. Log in to Oracle Identity Federation as the `oif_admin` user.
2. Navigate to **SAML 1.x/WS-Fed** -> **Domains**.
3. Click the **Add Non-Oracle Identity Federation Domain** button.



Add this information:

- Domain name
- Domain description
- Check **Enable This Domain**
- Check **Supported Protocols** according to your requirements
- Issuer - Federation Service URI collected in "Prerequisites" on page 7-29

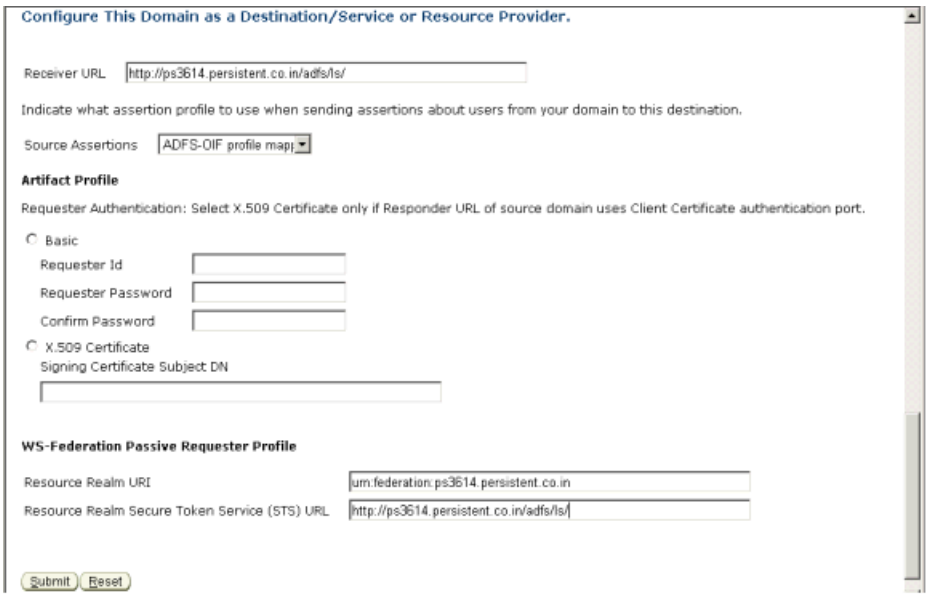
4. Scroll down to the section Configure this Domain as a Source/Identity Provider.



Enter the Responder URL. This is the Federation Service endpoint URL collected in the Prerequisites. Oracle Identity Federation generates Source ID based on Responder URL.

Note: To change the SourceID, you change the Responder URL, delete the information in the Source ID field, and click **Submit** to regenerate the source ID. Do *not* manually regenerate the SourceID.

5. Scroll down to the section Configure This Domain as a Destination/Service or Resource Provider.



Enter this information:

- Receiver URL – This is the Federation Service endpoint URL collected in the Prerequisites on page 7-29. The receiver service at the destination domain processes assertions.
- Source Assertions - Indicate what assertion profile to use when sending assertions about users from your domain to this destination.
- Resource Realm URI This is the Federation Service URI collected in the prerequisites.
- Resource Realm Secure Token Service (STS) URL - This is the Federation Service endpoint URL collected in the prerequisites.

6. Click **Submit** to submit the configuration.
7. Check the Domain list to verify that the domain was created and ensure that it is enabled.

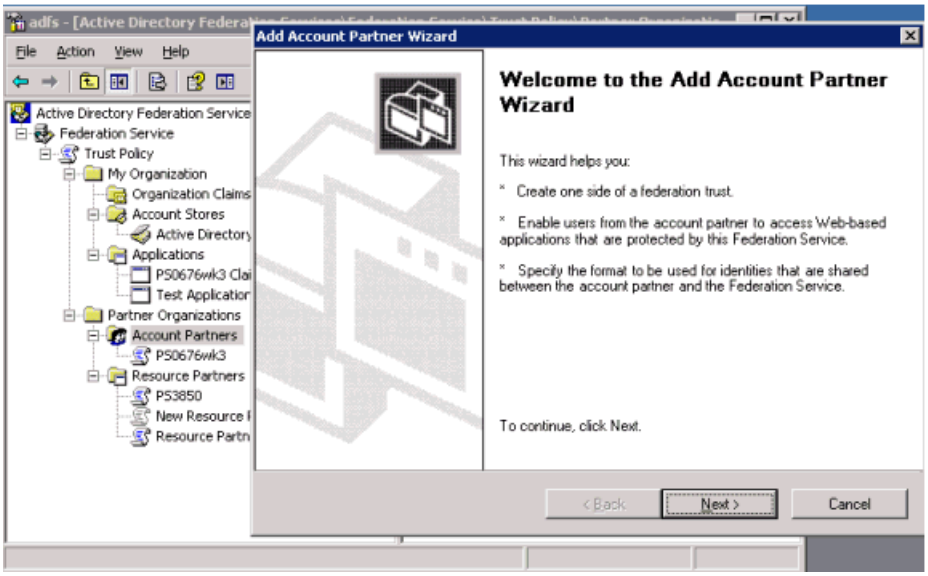
Configure ADFS as SP to recognize Oracle Identity Federation as IdP

This section explains the configuration needed in ADFS, as service provider, to recognize Oracle Identity Federation as an identity provider. This involves adding a new account partner for ADFS.

Take these steps to add an account partner:

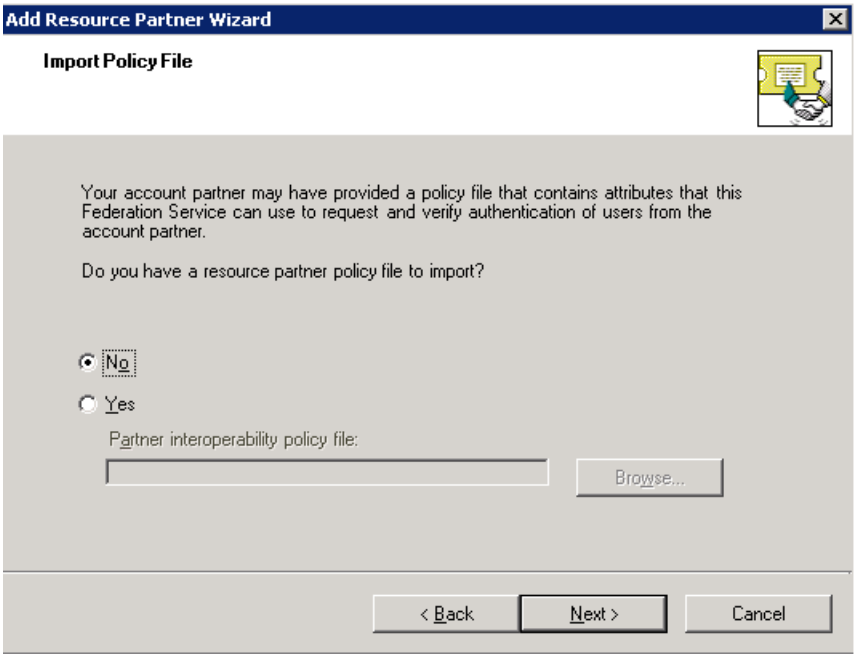
1. Navigate to **Start** -> **All Programs** -> **Administrative Tools** -> **Active Directory Federation Services**.

2. Double-click **Federation Service**, double-click **Trust Policy**, double-click **Partner Organizations**, right-click **Account Partners**, point to **New**, and click **Account Partner**.
3. The Welcome to the Add Account Partner Wizard page appears.



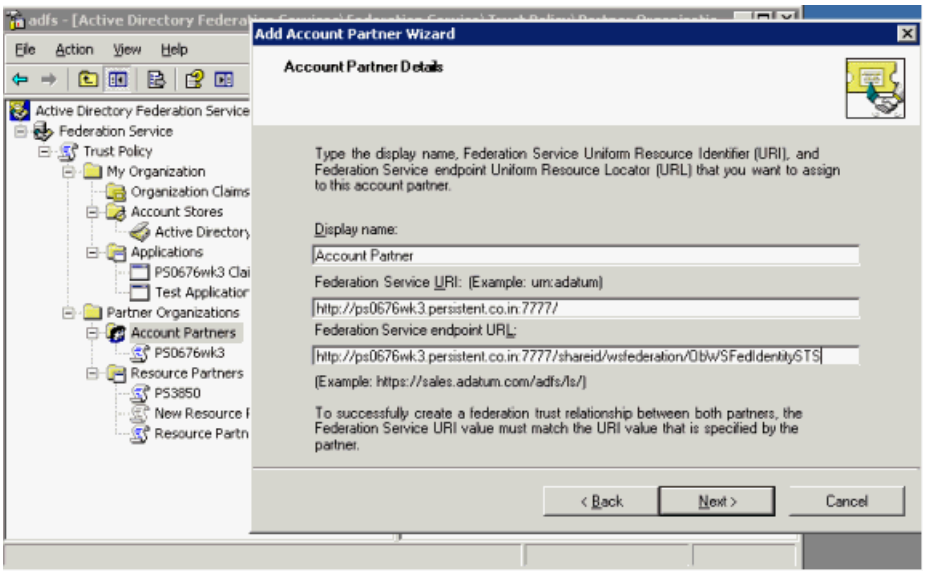
Click **Next**.

4. The Import Policy File page appears.



Select **No**, and click **Next**.

5. The Account Partner Details page appears.

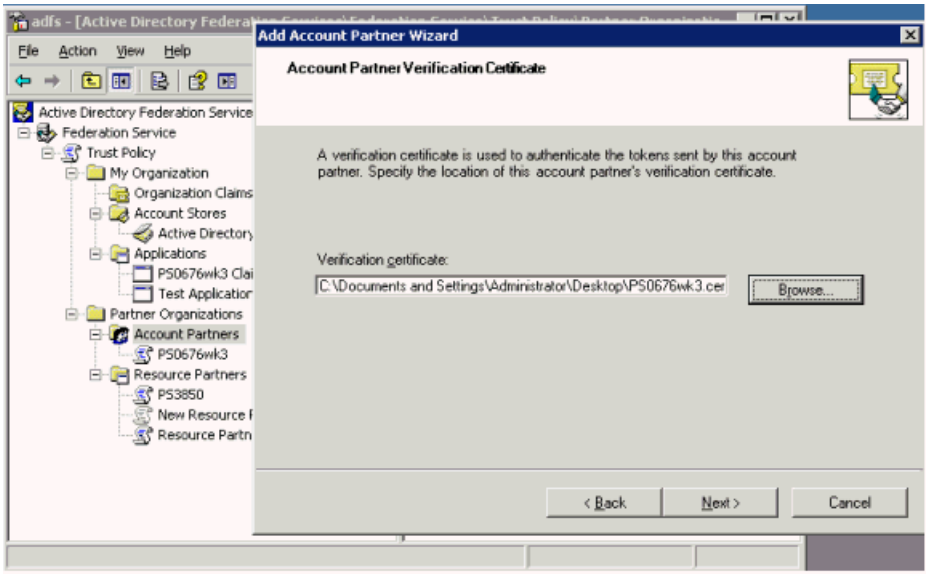


Enter this information:

- Display name - enter "Account Partner".
- Federation Service URI – enter the Identity Realm URI collected in the Prerequisites on page 7-29. *Note*: This value is case sensitive.
- Federation Service endpoint URL – enter the Identity Realm Secure Token Service (STS) URL collected in the Prerequisites.

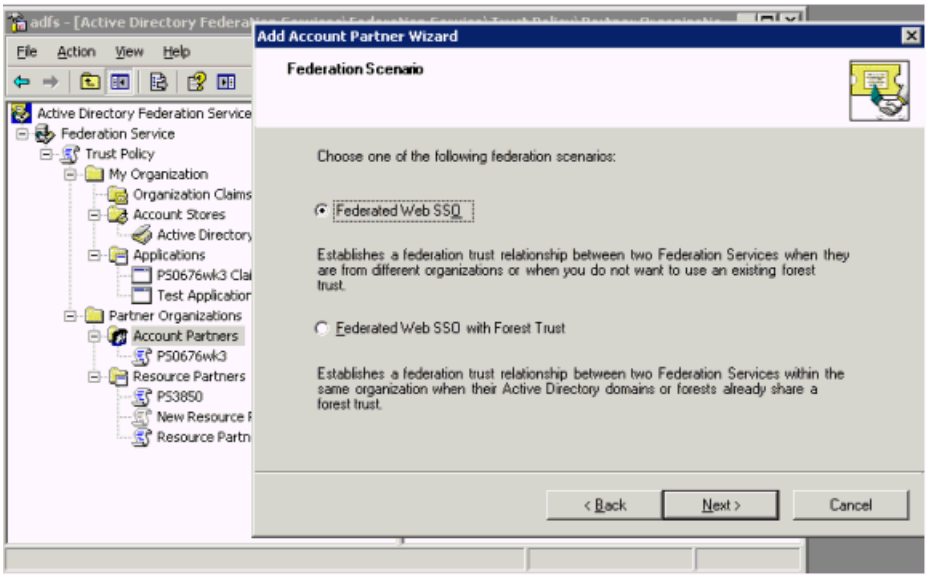
Click **Next**.

6. The Account Partner Verification Certificate page appears.



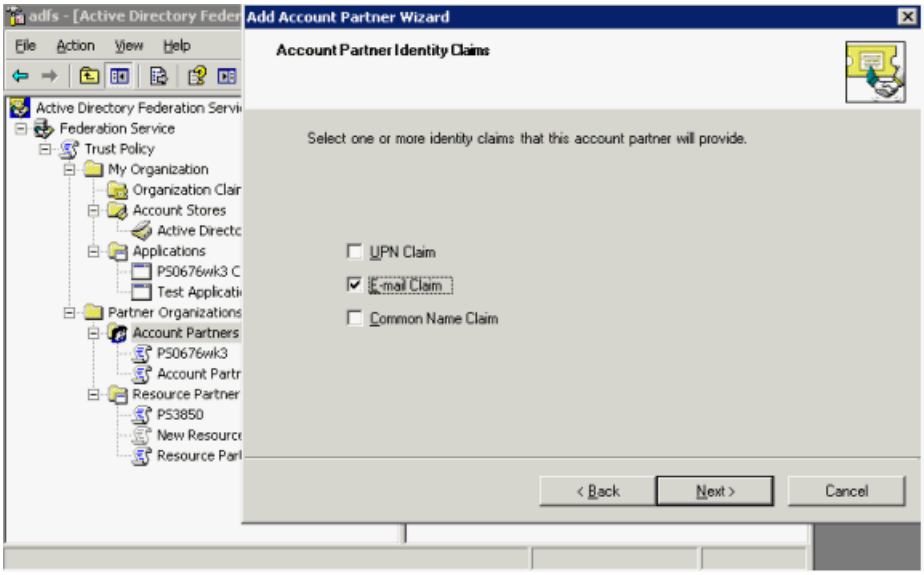
Click **Browse**. Select the certificate that you collected earlier in the Prerequisites, and click **Next**.

7. The Federation Scenario page appears.



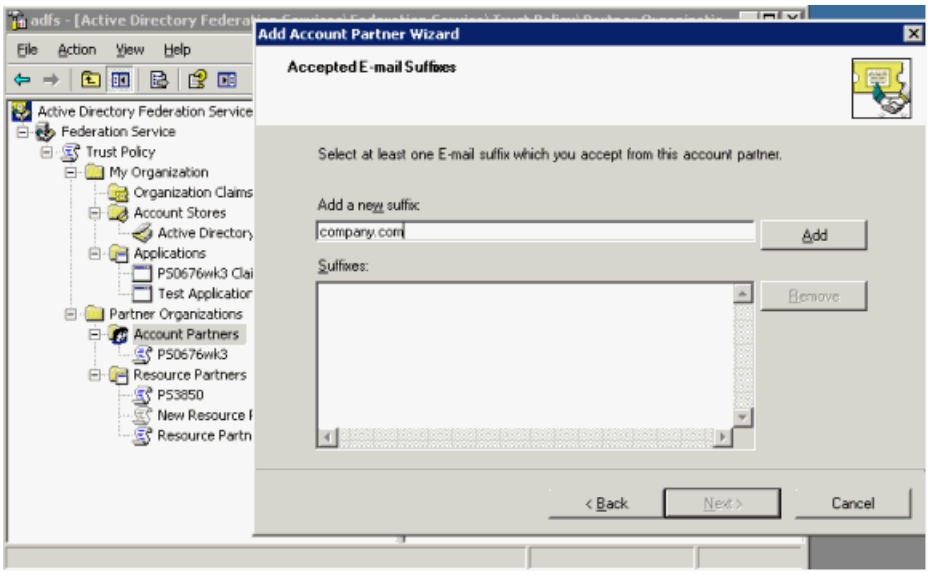
Select Federated Web SSO and click **Next**.

8. The Account Partner Identity Claims page appears.



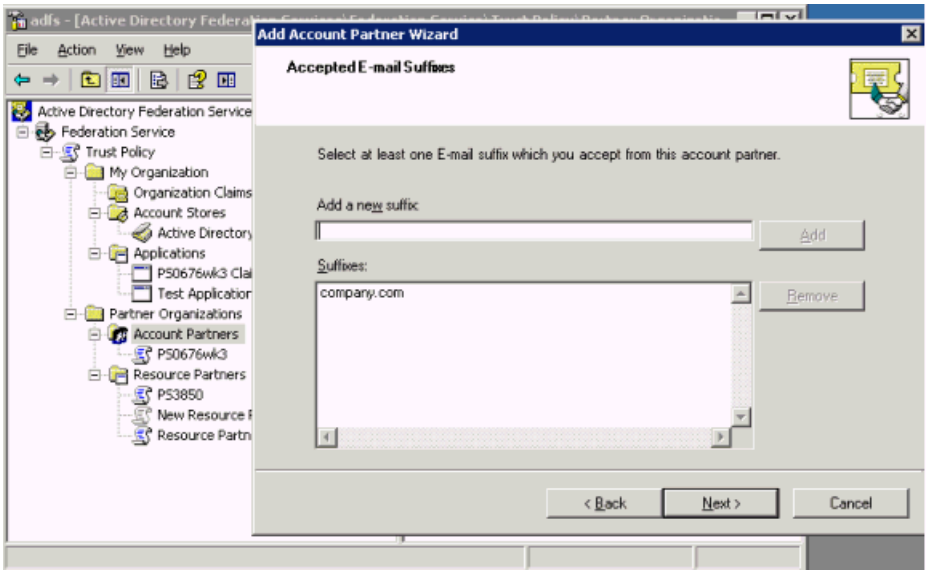
Select the Email Claim check box and click **Next**.

9. The Accepted E-mail Suffixes page appears.



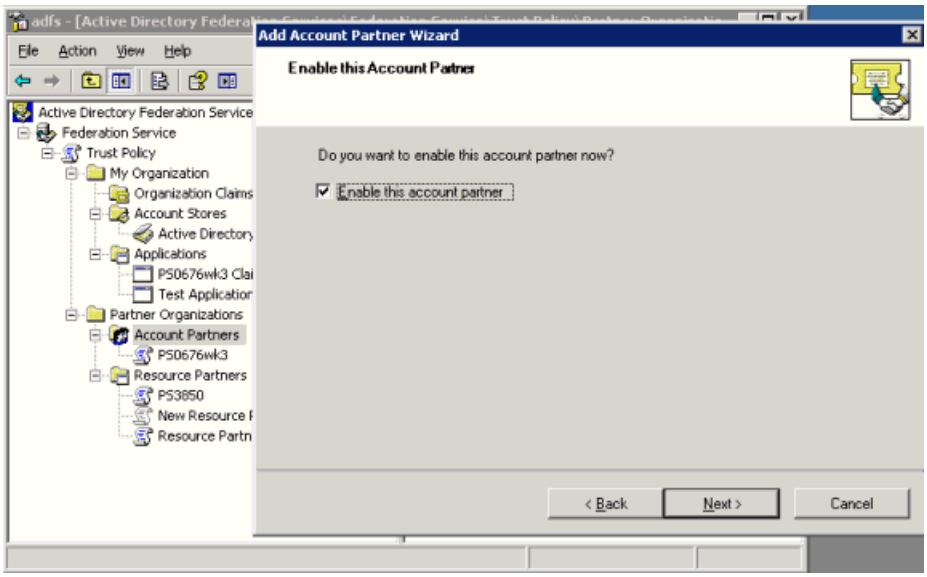
Enter a suffix (company.com in the example), and click **Add**.

10. The suffix is added and the page reappears.



Click **Next**.

11. The Enable this Account Partner page appears.



Check the **Enable this account partner** box, and click **Next**.

12. The Add Account Partner Wizard completion page appears. Click **Finish**.

Configure Claims

In Active Directory Federation Services (ADFS), an organization group claim represents a user's membership in a group or role. An organization custom claim is used by the Federation Service to provide custom information, such as an employee identification number, about a user.

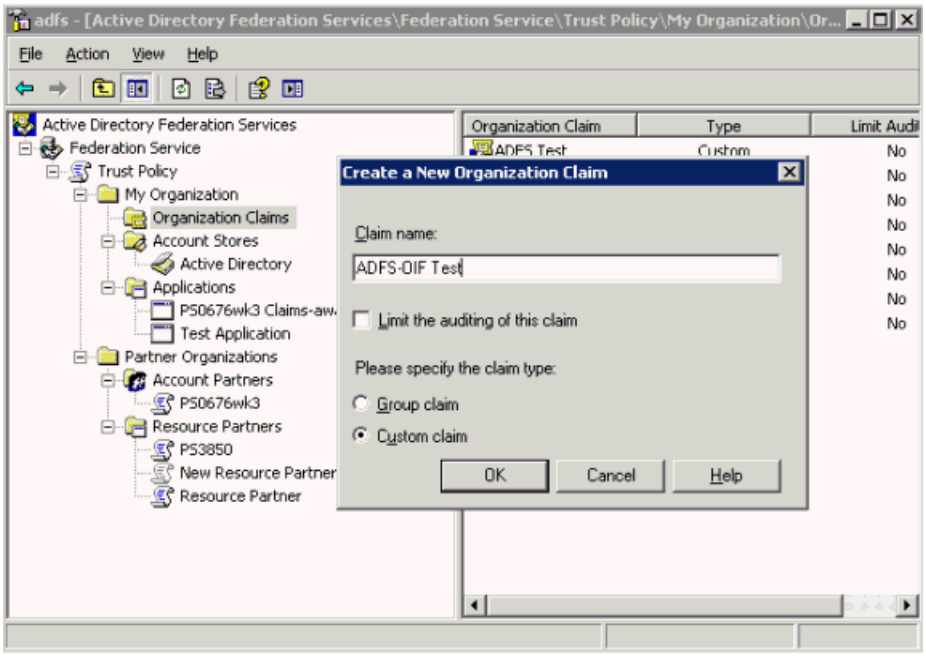
This section explains how to configure a custom claim and create the claim transform to map it to the organization custom claim.

The major tasks are as follows:

- Create custom claim for claims-aware application.
- Create custom claim extraction.
- Create incoming custom claim mapping.
- Enable e-mail identity claim for application.
- Enable claims for applications.

Create Custom Claim for Claims-aware Application Take these steps to create the custom claim:

1. Navigate to **Start** -> **All Programs** -> **Administrative Tools** -> **Active Directory Federation Services**.
2. Double-click **Federation Service**, double-click **Trust Policy**, double-click **My Organization**, right-click **Organization Claims**, point to **New**, and click **Organization Claim**.
3. The Create a New Organization Claim dialog box appears.



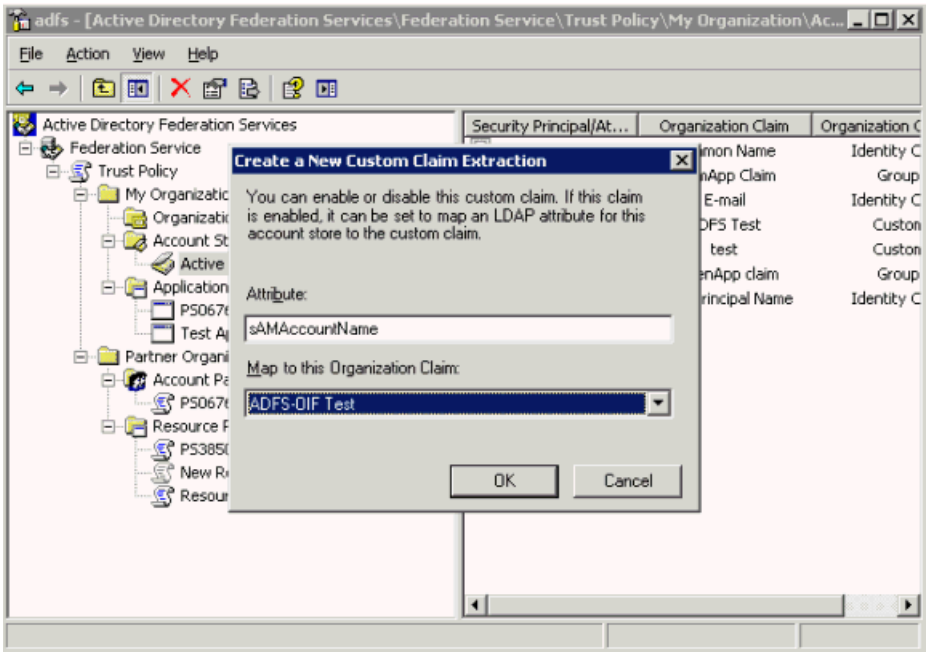
Provide this information:

4. Claim name – enter a claim name (ADFS-OIF Test in the example).
5. Check the **Custom Claim** box.
6. Click **OK**.

Create Custom Claim Extraction for Claims-aware Application In ADFS, an organization custom claim maps to a user attribute.

Take these steps to define an LDAP attribute to map the custom claims for a claims-aware application:

1. Navigate to **Start** -> **All Programs** -> **Administrative Tools** -> **Active Directory Federation Services**.
2. Double-click **Federation Service**, double-click **Trust Policy**, double-click **My Organization**, double-click **Account Stores**; right-click **Active Directory**, point to **New**, and click **Custom Claim Extraction**.
3. The Create a New Custom Claim Extraction dialog appears.



Provide this information:

- Attribute - Enter the name to which you would like to map the assertion attribute (sAmAccountName in the example).
- Map to this Organization Claim – select the claim you defined earlier (ADFS-OIF Test in the example)

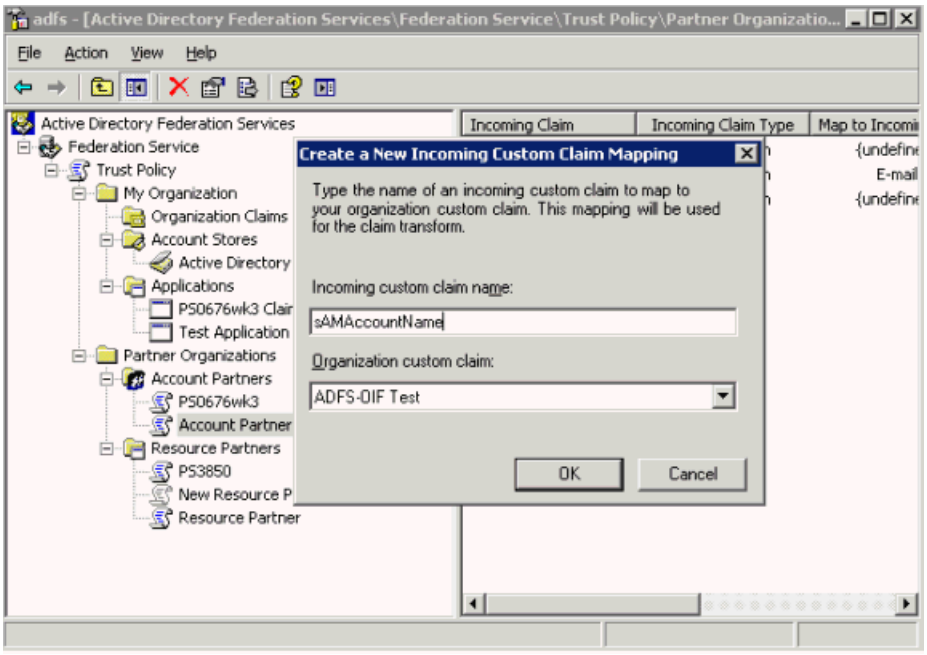
4. Click **OK**.

Create Incoming Custom Claim Mapping for Claims-aware Application The resource Federation Service uses incoming custom claim mappings to map custom claims, sent by an account partner, to claims that can be used by the resource partner to make authorization decisions.

Take these steps to create a mapping for the claim transform:

1. Navigate to **Start** -> **All Programs** -> **Administrative Tools** -> **Active Directory Federation Services**.
2. Double-click **Federation Service**, double-click **Trust Policy**, double-click **Partner Organizations**, double-click **Account Partners**; right-click **Account Partner**, point to **New**, and click **Incoming Custom Claim Mapping**.

   The Create a New Incoming Custom Claim Mapping dialog appears.

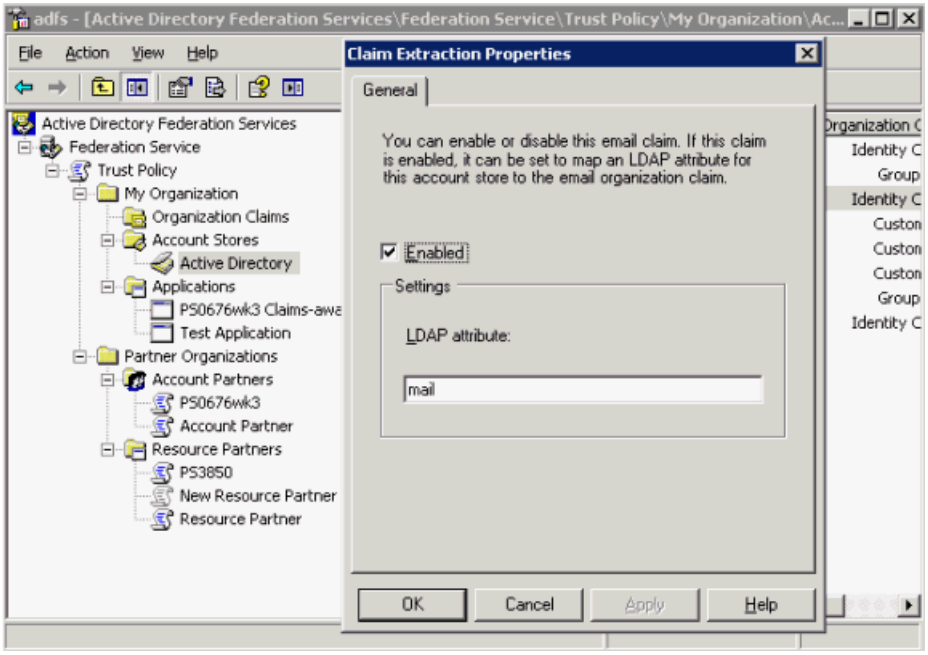


3. Provide this information:
   - Organization custom claim - Enter the custom claim name (ADFS-OIF Test in the example).
   - Incoming custom claim name – enter the attribute name (sAmAccountName in the example). *Note*: This attribute is passed on to the Source Domain Administrator by the destination domain administrator for use in creating Assertion Profile..

Enable Email Identity Claim for Claims-aware Application In Active Directory Federation Services (ADFS), organization identity claims are created along with the account or resource partner. They are incoming claims on the resource partner and outgoing claims on the account partner. Identity claims are enabled when you specify the identity type such as a UPN, e-mail, or common name.

Take these steps to establish an e-mail identity type for the claim:

1. Navigate to **Start** -> **All Programs** -> **Administrative Tools** -> **Active Directory Federation Services**.
2. Double-click **Federation Service**, double-click **Trust Policy**, double-click **My Organizations**, double-click **Account Stores**; click **Active Directory**, right click **Email Identity Claim**, and click **Properties**.
3. The Claim Extraction Properties dialog appears.

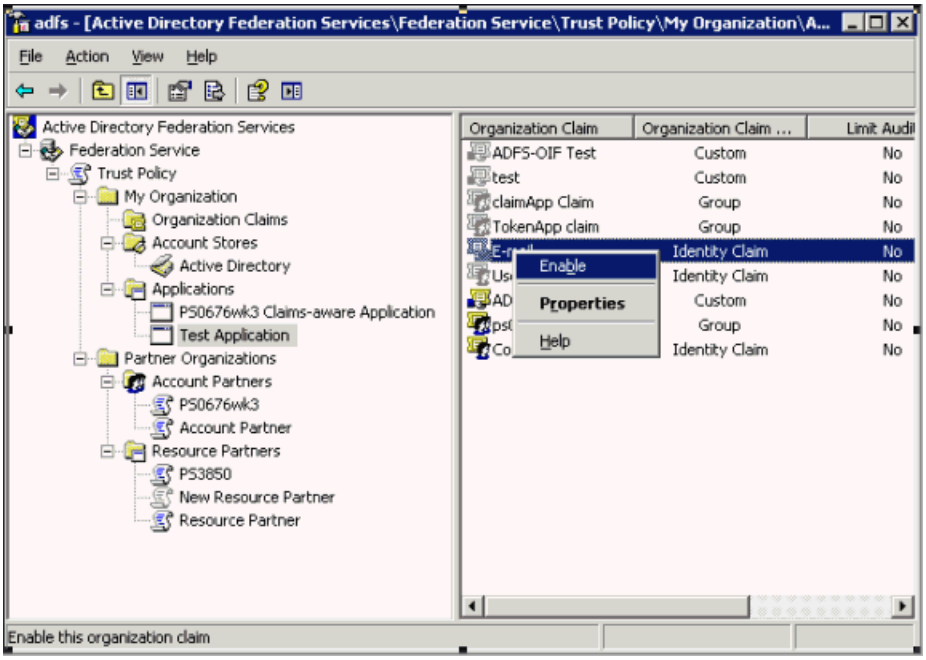


Enter the LDAP attribute and check the **Enabled** check box.

4. Click **OK**.

Enable the Identity Claim Take these steps to enable applicable claims for the application:

1. Navigate to **Start** -> **All Programs** -> **Administrative Tools** -> **Active Directory Federation Services**.
2. Double-click **Federation Service**, double-click **Trust Policy**, double-click **My Organizations**, double-click **Applications**; click the application name, right click **Email Identity Claim**, and click **Enable**.



IdP-initiated SSO with WS-Federation

When Oracle Identity Federation is enabled for SSO with WS-Federation, an SSO request can be initiated from the IdP, using a URL in the format:

```
http://oif_host:oif_
port/shareid/wsfederation/ObWSFedIdentitySTS/
```

The parameters are:

- wa - a required parameter which specifies the action to be performed. By including the action, URIs can be overloaded to perform multiple functions. *Note*: For sign-in, this string **must** be "wsignin1.0".
- wtrealm - the request realm URI defined by the SP
- wctx - the protected resource URL to be accessed at the SP's domain

For example:

```
http://oif_host:oif_port
/shareid/wsfederation/ObWSFedIdentitySTS/?wa=wsignin1.0
&wtrealm=Federation_Service_URI
&wctx=https://protected_resource/claimapp/\https://protected_resource/claimapp/
```

SP-initiated SSO with WS-Federation

When ADFS is enabled for SSO with WS-Federation, it is possible to initiate an SSO request from the SP, using a URL in the following format:

```
https://adfs_host/adfs/ls/
```

The parameters are:

- wa - This required parameter specifies the action to be performed. By including the action, URIs can be overloaded to perform multiple functions. For sign-in, this string **must** be "wsignin1.0".
- wctx - This is the protected resource URL to be accessed at the SP's domain
- wreply - This optional parameter is the URL to which responses are directed.

For example:

```
https://adfs_host
/adfs/ls/?wa=wsignin1.0
&wreply=https://protected_resource/claimapp/
&wct=2007-02-16T14:41:01Z
&wctx=https://protected_resource/claimapp/
```

IdP-initiated logout with WS-Federation

The URL is:

```
http(s)://oif_host:oif_
port/shareid/wsfederation/ObWSFedIdentitySTS?wa=wsignout1.0
```

SP-initiated logout with WS-Federation

The URL is:

```
https://adfs_host/adfs/ls/?wa=wsignout1.0
```

where `wa` is a required parameter that specifies the action to be performed. By including the action, URIs can be overloaded to perform multiple functions. For logout, this string **must** be "wsignout1.0".

Logout no-fail-on-error Option

This section describes the logout no-fail-on-error feature, which is introduced in patch set 10.1.4.2.

Overview of the no-fail-on-error Feature

During a Liberty 1.x / SAML 2.0 Logout operation, Oracle Identity Federation contacts the remote providers with which the user performed a federated SSO operation, and instructs them to terminate their local session for that user.

In Oracle Identity Federation release 10.1.4.0.1, if a remote provider did not support the Global Logout protocol, Oracle Identity Federation reported an Internal Server Error to the user's browser.

In the 10.1.4.2.0 patch set release, the administrator can configure how Oracle Identity Federation should respond when an error occurs in the logout flow. The administrator can:

- decide to force Oracle Identity Federation to finish the logout operation successfully, or
- report an error to the user's browser.

Configuring the Option

Take these steps to configure the behavior of Oracle Identity Federation during logout:

1. Open the `$ORACLE_HOME/fed/conf/config.xml` file.
2. Locate the `FederationConfig` XML element, and set its `useLocalConfig` attribute to `true`:

   ```
   <FederationConfig useLocalConfig="true">
   ```

3. Locate the XML `Config` element named `serverconfig`, and find the `slofailonerror` property:

   ```
   <Config name="serverconfig">
   ...
      <property name="slofailonerror">false</property>
   ...
   </Config>
   ```

4. Set the value of the property:
   - `true` to force Oracle Identity Federation to report a failure to the user's browser if an error occurs
   - `false` to instruct Oracle Identity Federation to successfully finish the logout operation, even if an error occurred
5. Save the file and exit
6. Restart the OC4J_FED instance.

Logout Status

Oracle Identity Federation provides logout capabilities to sign off a user session. Logouts are triggered:

- through the WS-Fed/Liberty 1.x/SAML 2.0 protocols when a remote provider invokes a logout operation; or
- by invoking a URL on the Oracle Identity Federation server that initiates the Logout flow; this involves interacting with remote providers to sign off the user.

When invoking the Oracle Identity Federation logout service through the `/fed/user/logout` URL, you can specify a return URL parameter to which the user will be redirected.

See Also: "Configuring the Logout Service" on page 6-132.

At the end of the logout operation, Oracle Identity Federation can optionally indicate the status of the signoff operation to report if the Global Logout operation was successful. Knowing the status can be useful because in some cases, there could be an error at a remote server, or one peer provider might not support the Global Logout protocol.

By default, Oracle Identity Federation does not return any logout status. When this feature is configured, a query parameter is appended to the return URL to indicate the logout status:

- the `orafed_slostatus` query parameter indicates the Liberty 1.x/SAML 2.0 logout status, with these possible values:
  - 0 means success
  - 1 means failure
- the `orafed_slowsfed` query parameter indicates the WS-Fed logout status, with these possible values:
  - 0 means success
  - 1 means failure

Take these steps to configure the logout status feature:

1. Open the `$ORACLE_HOME/fed/conf/config.xml` file.
2. Locate the `FederationConfig` XML element, and set its `useLocalConfig` attribute to `true`:

```
<FederationConfig useLocalConfig="true">
```

3. Locate the XML `Config` element named `serverconfig`, and find the `sloreturnstatus` property:

```
<Config name="serverconfig">
   ...
   <property name="sloreturnstatus">false</property>
   ...
</Config>
```

4. Set the value of the property:
   - `true` to direct Oracle Identity Federation to add the status of the Global Logout on the return URL when using Liberty 1.x/SAML 2.0 protocols
   - `false` to direct Oracle Identity Federation to omit the status of the Global Logout when using Liberty 1.x/SAML 2.0 protocols
5. Save the file and exit.

6. Open the $ORACLE_HOME/fed/shareid/oblix/config/shareid-config.xml file.
7. Set the useLocalConfig attribute to true to force changes to be persisted at restart:

```
<SHAREidConfiguration … useLocalConfig="true">
```

8. Locate the LogoutConfig XML element.
9. Set the SloReturnStatus attribute to either true or false:
   - true to direct Oracle Identity Federation to add the status of the Global Logout on the return URL when using WS-Fed protocols
   - false to direct Oracle Identity Federation to omit the status of the Global Logout when using WS-Fed protocols
10. Restart the OC4J_FED instance.

Configuring SAML 2.0 Authentication Query Response

With release 10.1.4.2.0, Oracle Identity Federation supports the SAML 2.0 authentication query protocol (AuthnQuery) on the IdP side. This protocol is used to query the IdP to request statements about authentication acts that have occurred in a previous interaction between the indicated subject and the authentication authority (IdP).

> **Note:** The protocol must not be used as a request for new authentication using credentials provided in the query.

Take these steps to configure Oracle Identity Federation to respond to authentication queries:

1. Open the $ORACLE_HOME/fed/conf/config.xml file.
2. Locate the XML element named FederationConfig and set its useLocalConfig attribute to true:

```
<FederationConfig useLocalConfig="true">
```

3. Locate the XML Config element named idpsaml20, and look for the authnresponderenabled property:

```
<Config name="idpsaml20">
...

   <property name="authnresponderenabled">false</property>
...

</Config>
```

4. Change the value of the property to true if you want Oracle Identity Federation to respond to AuthnQuery messages.
5. Save the file and exit.
6. Restart the OC4J_FED instance.

Once Oracle Identity Federation is configured in this way, you can use the SAML 2.0 AuthnQuery message element to make authentication queries to the IdP (see

[SAMLCore] in Table B–1 for details about `AuthnQuery`). In response, the IdP will return assertions with authentication statements that match the query's requirements.

Configuring SAML 2.0 Assertion ID Request

At times, the SP requester knows the unique identifier of one or more assertions (which have been exchanged in the past), and may want to obtain these assertions anew.

With release 10.1.4.2.0, Oracle Identity Federation supports the SAML 2.0 assertion ID request protocol (`AssertionIDRequest`) which allows the SP to query for existing assertions by their identifiers.

Take these steps to configure Oracle Identity Federation to support `AssertionIDRequest` messages:

1. Open the `$ORACLE_HOME/fed/conf/config.xml` file.
2. Locate the XML element named `FederationConfig` and set its `useLocalConfig` attribute to `true`:

```
<FederationConfig useLocalConfig="true">
```

3. Locate the XML `Config` element named `idpsaml20`, and look for the `assertionidresponderenabled` property:

```
<Config name="idpsaml20">
...
   <property name="assertionidresponderenabled">false</property>
...
</Config>
```

4. Change the value of the property to `true` to enable the assertion ID request functionality.
5. Save the file and exit.
6. Restart the OC4J_FED instance.

Once assertion ID request functionality is configured in this way, Oracle Identity Federation caches the most recent assertions generated by the SAML authority for later retrieval by the SP requester. You can use the SAML 2.0 `AssertionIDRequest` message element to request old assertions from Oracle Identity Federation; they are returned in a `Response` message (see [SAMLCore] in Table B–1 for details about `AssertionIDRequest`).

Additional eTrust SiteMinder Configuration

This section describes topics related to configuring Oracle Identity Federation with eTrust SiteMinder:

- Types of Policy Objects
- Creating the Policy Objects
- Configuring Oracle Identity Federation for Startup eTrust SiteMinder Operations
- Configuring Oracle Identity Federation to use a different User Data Store

> **See Also:** "Deploying Oracle Identity Federation with eTrust SiteMinder" on page 4-13 for eTrust SiteMinder configuration details and examples.

Types of Policy Objects

When integrating the Oracle Identity Federation server with eTrust SiteMinder, policy objects must be created in the SiteMinder Policy Server. These objects can be created:

- by running a Perl script on the SiteMinder Policy machine, or
- by letting Oracle Identity Federation automatically create the objects at startup

The following policy objects are created:

1. AgentGroup

   An AgentGroup regroups the Web Agents defined by a cluster of Oracle Identity Federation servers, such as in a load balanced/HA scenario.

   The default name of the group is OracleFederation, but it can be changed in the Perl script as well as in the Oracle Identity Federation configuration files.

2. Web Agent

   A Web Agent, specific to a physical Oracle Identity Federation server, is used to interact with the SiteMinder Policy Server at runtime to authenticate users and create user sessions.

   The agent is a member of the federation AgentGroup, and its name is defined to be `SMBridgeAgent.`*`OIF_INSTANCE_NAME`*, where *`OIF_INSTANCE_NAME`* is the instance name of the Oracle Application Server on which Oracle Identity Federation is running. (You can find the instance name on the Enterprise Manager console, where it is displayed at the top left of the welcome page. For example: Application Server: orclfed.stadm04.us.oracle.com).

   If using the Perl script to create the policy objects, you will need to edit the script to reflect the instance name of the running Application Server.

   Note: The instance name of an Application Server is unique across all deployments, even in a load balanced/HA scenario.

3. Other policy objects

   Oracle Identity Federation will create and use various policy objects to authenticate users, or to create an eTrust SiteMinder session in SP mode.

   These objects have a prefixed name that can be configured to allow multiple logical Oracle Identity Federation servers to integrate with a given eTrust SiteMinder server. By default the Prefix name is OracleFederation and it should not be changed (even in a load balanced/HA scenario), unless different logical Oracle Identity Federation instances/clusters will interact with the eTrust SiteMinder server. If a specific prefix needs to be used, you will need to edit the Perl script and the Oracle Identity Federation Configuration files.

   Oracle Identity Federation defines a realm that is used during authentication procedures. This realm:

   - is named *`PREFIX`*`_Login`
   - is linked to the AgentGroup described earlier
   - is protecting the `/`*`PREFIX`*`/Login` resource
   - has an authentication scheme named *`PREFIX`*`_Login`
   - has an associated rule named *`PREFIX`*`_Login`

Oracle Identity Federation defines a second realm that is used to create eTrust SiteMinder user sessions in SP mode. This realm:

- is named *PREFIX*_LoginNoPwd
- is linked to the AgentGroup described earlier
- is protecting the /*PREFIX*/LoginNoPwd resource
- has an authentication scheme named *PREFIX*_LoginNoPwd

Finally, Oracle Identity Federation defines a Response Object that it uses to retrieve user attributes from the eTrust SiteMinder server at runtime. This object is named *PREFIX*_UserAttributes.

Creating the Policy Objects

When Oracle Identity Federation is integrated with eTrust SiteMinder server, policy objects can be created automatically, or through a perl script.

Creating Policy Objects with a Script

Take these steps to develop and implement the createSMConfig.pl script:

1. Open the $ORACLE_HOME/fed/setup/sm/createSMConfig.pl file.
2. Set the following variables:
   - $adminName: this is the eTrust SiteMinder administrator ID.
   - $adminPwd: this is the eTrust SiteMinder administrator password.
   - $domain_name: this is the eTrust SiteMinder to contain the policy objects.
   - $userdir_name: this is the eTrust SiteMinder-configured name of the user directory for the domain.
   - $ip_address: this is the IP address of the machine where the Oracle Identity Federation server is installed.
   - $shared_secret: this is the secret string shared between the agent and the policy server.
   - $oif_instance_name: the instance name of the Oracle Application Server where Oracle Identity Federation is running.
   - $prefix: the prefix to use for the Oracle Identity Federation policy objects. By default it is OracleFederation.
3. Have the eTrust SiteMinder administrator create an AgentGroup, and add all web agents created by the perl script to this agent group. Each server instance would add one agent.
4. The administrator must edit the two newly created realms, and reference the AgentGroup instead of the Oracle Identity Federation Web Agent.
5. Save the script changes.

Note: If the script is used to create the policy objects and if the AgentGroup name and Prefix are changed from their default values, you must update the Oracle Identity Federation configuration. See the discussion "Creating Policy Objects Automatically" below for details.

Creating Policy Objects Automatically

Take these steps:

1. Follow the steps listed in "Edit User Data Store" on page 6-67 to set up the eTrust SiteMinder configuration information.
2. Save.
3. If the AgentGroup name or the Prefix needs to be changed, continue with the remaining steps. Otherwise, restart the OC4J_FED instance (last step).
4. Open the `$ORACLE_HOME/fed/shareid/oblix/config/shareid-config.xml` file.
5. Set the `useLocalConfig` attribute to `true` to force changes to be persisted at restart:

   ```
   <SHAREidConfiguration … useLocalConfig="true">
   ```

6. Locate the IdMBridge XML element whose `Name` attribute is `SM`.
7. Add the XML attribute named `AgentGroupName` to specify the name of the AgentGroup to use:

   ```
   <IdMBridge Name="SM" ... AgentGroupName="OIFGroup" ...></IdMBridge>
   ```

8. Add the XML attribute named `PrefixName` to specify the prefix to use:

   ```
   <IdMBridge Name="SM" ... PrefixName="OIFPrefix" ...></IdMBridge>
   ```

9. Save the file.
10. Restart OC4J_FED.

Configuring Oracle Identity Federation for Startup eTrust SiteMinder Operations

Oracle Identity Federation can be configured to perform various checks and operations at each startup.

Create Policy Objects if Missing

To check for the presence of the policy objects in the eTrust SiteMinder Policy server, and create them if missing:

1. Open the `$ORACLE_HOME/fed/shareid/oblix/config/shareid-config.xml` file.
2. Set the `useLocalConfig` attribute to `true` to force changes to be persisted at restart:

   ```
   <SHAREidConfiguration … useLocalConfig="true">
   ```

3. Locate the IdMBridge XML element whose `Name` attribute is `SM`.
4. Set the XML attribute named `SkipInitialize` to `false` to indicate that Oracle Identity Federation needs to check the policy objects:

   ```
   <IdMBridge Name="SM" ... SkipInitialize="false" ...></IdMBridge>
   ```

5. Set the XML attribute named `AllowSMCreation` to `true` to indicate that policy objects can be created if missing:

   ```
   <IdMBridge Name="SM" ... AllowSMCreation="true" ...></IdMBridge>
   ```

6. Save and restart OC4J_FED.

Check if Policy Objects are Missing

To check for the presence of the policy objects in the eTrust SiteMinder policy server, and report an error if missing (but not to create the objects):

1. Open the `$ORACLE_HOME/fed/shareid/oblix/config/shareid-config.xml` file.
2. Set the `useLocalConfig` attribute to `true` to force changes to be persisted at restart:

   ```
   <SHAREidConfiguration … useLocalConfig="true">
   ```

3. Locate the IdMBridge XML element whose `Name` attribute is `SM`.
4. Set the XML attribute named `SkipInitialize` to `false` to indicate that Oracle Identity Federation needs to check the policy objects:

   ```
   <IdMBridge Name="SM" ... SkipInitialize="false" ...></IdMBridge>
   ```

5. Set the XML attribute named `AllowSMCreation` to `false` to indicate that an error needs to be reported if the policy objects are missing:

   ```
   <IdMBridge Name="SM" ... AllowSMCreation="false" ...></IdMBridge>
   ```

6. Save and restart OC4J_FED.

Bypass Policy Object Verification and Creation

To bypass the verification and creation of the Oracle Identity Federation policy objects:

1. Open the `$ORACLE_HOME/fed/shareid/oblix/config/shareid-config.xml` file.
2. Set the `useLocalConfig` attribute to `true` to force changes to be persisted at restart:

   ```
   <SHAREidConfiguration … useLocalConfig="true">
   ```

3. Locate the IdMBridge XML element whose `Name` attribute is `SM`.
4. Set the XML attribute named `SkipInitialize` to `true` to indicate that Oracle Identity Federation should not check the policy objects:

   ```
   <IdMBridge Name="SM" ... SkipInitialize="true" ...></IdMBridge>
   ```

5. Save and restart OC4J_FED.

Configuring Oracle Identity Federation to use a different User Data Store

You can configure Oracle Identity Federation server to use a user repository other than the one used by the CA SiteMinder server. For example, the CA SiteMinder Policy server could use an LDAP server as the user repository while Oracle Identity Federation is configured to use an RDBMS.

Start by following the steps listed in "Edit User Data Store" on page 6-67 to set up the CA SiteMinder configuration information.

Next, take these steps to indicate to Oracle Identity Federation that the user data store is different from the one used by CA SiteMinder:

1. Open the $ORACLE_HOME/fed/shareid/oblix/config/shareid-config.xml file.
2. Set the `useLocalConfig` attribute to `true` to force changes to be persisted at restart:

```
<SHAREidConfiguration … useLocalConfig="true">
```

3. Locate the IdMBridge XML element whose `Name` attribute is `SM`.
4. Set the XML attribute named `SecondaryBridgeEqualToSMUserDir` to:
   - `false` to indicate that the user data store is different from the one used by the CA SiteMinder policy server.
   - `true` to indicate that the user data store is the same for both.

   For example:

   ```
   <IdMBridge Name="SM" ... SecondaryBridgeEqualToSMUserDir="false"
   ...></IdMBridge>
   ```

5. Save and restart OC4J_FED.

8

Monitoring Oracle Identity Federation

This chapter describes how to monitor Oracle Identity Federation. Topics include:

- About Oracle Identity Federation Monitoring
- Monitoring Console
- Archiving Metrics

About Oracle Identity Federation Monitoring

Oracle Identity Federation administrators can derive several important benefits from the ability to monitor key aspects of their federation server deployment. Oracle Identity Federation real-time monitoring features enable you to:

- measure application performance
- observe application usage patterns
- detect potential security issues
- monitor availability and other aspects of server operation

This section contains these topics:

- Metrics
- Monitoring Components
- Monitoring Data Flow

Metrics

Oracle Identity Federation Monitoring provides the administrator with a number of metrics collected from a site's different server instances:

- Current Oracle Identity Federation server availability (Up/Down)
- Server availability over a user-defined time period
- Authentication requests sent by the service provider over a user-defined time period, categorized by:
  - total number of authentication requests sent
  - total successful requests
  - total failed requests
- Authentication requests received by the identity provider over a user-defined time period, categorized by:

  - total number of authentication requests received
  - total successful requests
  - total failed requests
- Name identifier registration requests sent by the identity provider over a user-defined time period, categorized by:
  - total number of name identifier registration requests sent
  - total successful requests
  - total failed requests
- Name identifier registration requests received by the identity provider over a user-defined time period, categorized by:
  - total number of name identifier registration requests received
  - total successful requests
  - total failed requests
- Name identifier registration requests sent by the service provider over a user-defined time period, categorized by:
  - total number of name identifier registration requests sent
  - total successful requests
  - total failed requests
- Name identifier registration requests received by the service provider over a user-defined time period, categorized by:
  - total number of name identifier registration requests received
  - total successful requests
  - total failed requests
- Federation termination requests sent by the identity provider over a user-defined time period, categorized by:
  - total number of federation termination requests sent
  - total successful requests
  - total failed requests
- Federation termination requests received by the identity provider over a user-defined time period, categorized by:
  - total number of federation termination requests received
  - total successful requests
  - total failed requests
- Federation termination requests sent by the service provider over a user-defined time period, categorized by:
  - total number of federation termination requests sent
  - total successful requests
  - total failed requests
- Federation termination requests received by the service provider over a user-defined time period, categorized by:

- total number of federation termination requests received
- total successful requests
- total failed requests

The degree of aggregation depends on the type of information being presented. Some data is specific to a server instance, other data may apply to a server acting in a specific role such as Identity Provider, and still other data may be aggregated across all server instances.

Monitoring Components

Oracle Identity Federation monitoring components include:

1. Metrics Collection Engine

   Consisting of program logic that provides a framework for metric collection, this component is responsible for tracking and caching the metrics generated by an Oracle Identity Federation instance. Events such as requests, responses, and errors provide the inputs for metrics collection.

2. Data Transfer Module

   This tool formats the collected data into an appropriate format and makes it available for use by other monitoring components.

3. Monitoring Agent

   The Monitoring Agent periodically requests data from various Oracle Identity Federation instances. The agent consults a configuration file to determine which instances are to be queried, and the query interval. It then issues the appropriate requests to the data transfer modules of the relevant instances.

   Data collected by the Monitoring Agent is written to an in-memory cache. The data is also archived to a log file.

4. Monitoring Console

   The Monitoring Console is the interface that Oracle Identity Federation administrators use to view the collected metrics.

5. Archive Log

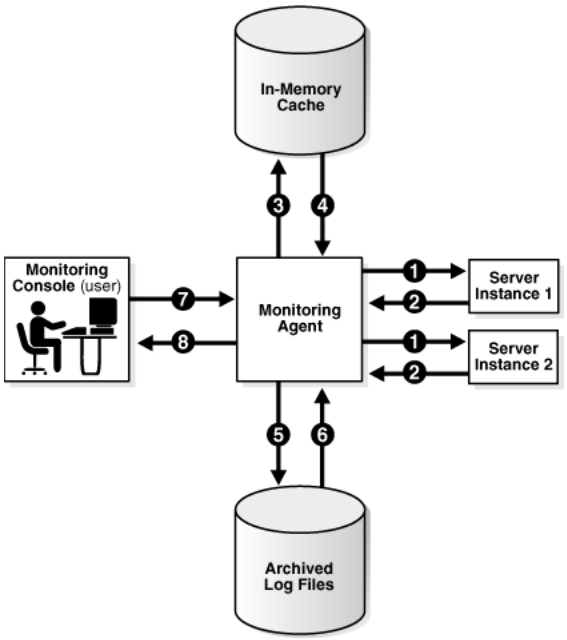
   This component allows the Monitoring Agent to save metrics to disk.

Metrics collection, handling, and transfer components reside in individual Oracle Identity Federation instances. The Monitoring Agent and Monitoring Console are bundled together as a separate J2EE application.

Monitoring Data Flow

Figure 8–1 shows how requests, metrics, and supporting data flows between and is used by the Monitoring Agent, Monitoring Console, and other Oracle Identity Federation components.

Figure 8–1 Data Flow Among Monitoring Components



The flow of data can be described as follows:

1. The Monitoring Agent periodically sends HTTP requests to Oracle Identity Federation.
2. Individual instances of Oracle Identity Federation utilize the data collection and formatting modules to gather and return their respective federation statistics to the Monitoring Agent.
3. The Monitoring Agent writes the data retrieved from Oracle Identity Federation to an in-memory cache.
4. The Monitoring Agent retrieves data from the in-memory cache when it needs to send the data to the Monitoring Console to satisfy a user request.
5. The Monitoring Agent writes the data retrieved from Oracle Identity Federation to a log file for archival purposes.
6. In the future, the Monitoring Agent will also be able to retrieve data from log files and send it to the Monitoring Console to satisfy user requests.
7. The Monitoring Console queries the Monitoring Agent for a specific set of metrics collected by Oracle Identity Federation.
8. The Monitoring Agent retrieves data from the cache (Step 4) and sends a set of metrics satisfying the query back to the Monitoring Console, where it is displayed to the user.

Monitoring Console

The Oracle Identity Federation Monitoring Console provides the following types of metrics for server administrators:

- Server availability metrics
- Protocol metrics on requests sent and received by providers, including:

- Authentication requests
- NameID requests
- Federation termination requests

See Also: For a complete list of available metrics, see "Metrics" on page 8-1.

This section describes and provides examples of the Monitoring Console pages:

- Accessing the Console
- Monitor Agent Home
- Monitor Agent IdP Statistics Home
- Monitor Agent IdP Statistics (SSO)
- Monitor Agent IdP Statistics (Identity Federation)
- Monitor Agent IdP Statistics (Peer Provider)
- Monitor Agent SP Statistics Home
- Monitor Agent SP Statistics (SSO)
- Monitor Agent SP Statistics (Identity Federation)
- Monitor Agent SP Statistics (Peer Provider)
- Metric Display at the Console

Accessing the Console

To log into Oracle Identity Federation Monitoring Console:

1. On all platforms, start the login process by pointing your browser to the login URL:

   `http://machine-name:open-port/fedmon`

2. Log in to the Monitoring Console by entering the username `oif_mon` and the password supplied during installation.

Monitoring Agent Home Tab

The home page for the Oracle Identity Federation Monitoring Console contains a Monitored Installations table, which displays a list of all monitored server instances.

See Also: "Monitor Agent Home" for details

Monitoring Console Metrics Page

Selecting a server instance takes you to the metrics pages of the Oracle Identity Federation Monitoring Console for that instance. Each metrics page consists of two panels. The top panel allows you to compose a metric query. The bottom panel displays the results.

The different metrics pages are described in "Monitor Agent Home".

Monitoring Agent Configuration Tab

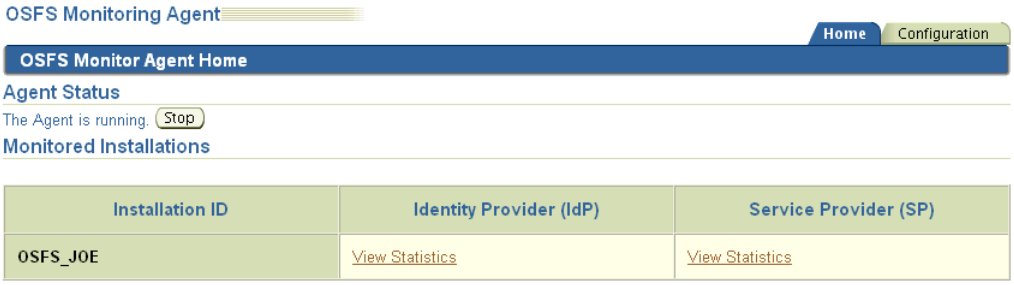
The Configuration tab allows you to monitor additional server instances and to maintain currently monitored installations.

See "Managing Monitored Installations" on page 8-14 for details.

Monitor Agent Home

The home page is the starting point for monitoring Oracle Identity Federation. It contains:

- The current status of the Monitor Agent - running or stopped. Click the button to change the status of the Monitor Agent.
- A table showing the installations being monitored by the agent. Click on the link in the Identity Provider or Service Provider column to view statistics for that role.



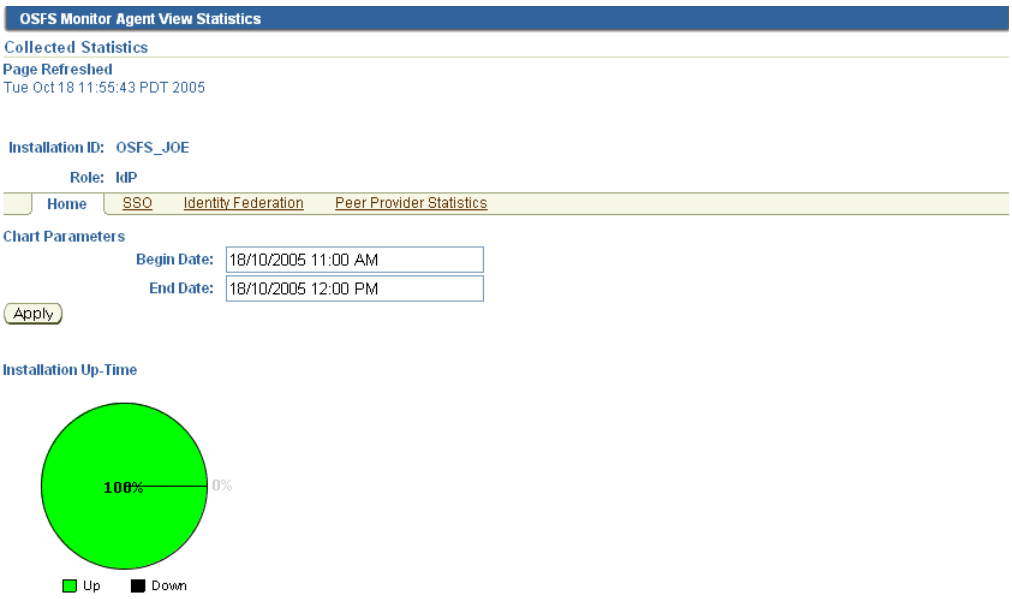
Monitor Agent IdP Statistics Home

This is the home page for viewing identity provider statistics for an installation.

Fields at the top of the page let you control chart parameters:

- **Begin Date** is the start date and time of the monitored period.
- **End Date** is the end date and time of the monitored period.

Click the **Apply** button to refresh the display using the specified parameters.



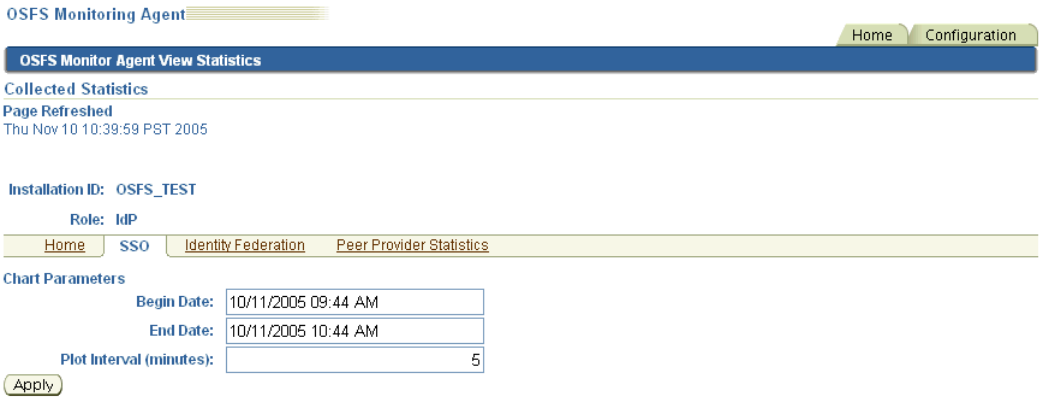
Monitor Agent IdP Statistics (SSO)

This page displays authentication requests received by an identity provider in a specified period. The server instance ID and the role (IdP) are displayed at the top of the page.

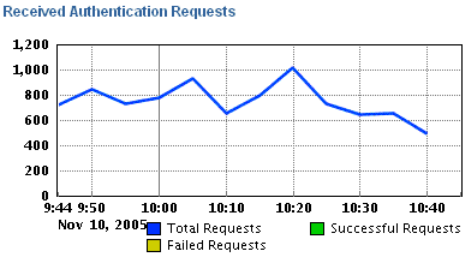
Fields at the top of the page let you control chart parameters:

- **Begin Date** is the start date and time of the monitored period.
- **End Date** is the end date and time of the monitored period.
- **Plot Interval** is the interval, in minutes, to use for the chart's horizontal axis.

Click the **Apply** button to refresh the display using the specified parameters.



The chart shows total, successful, and failed requests in the period.



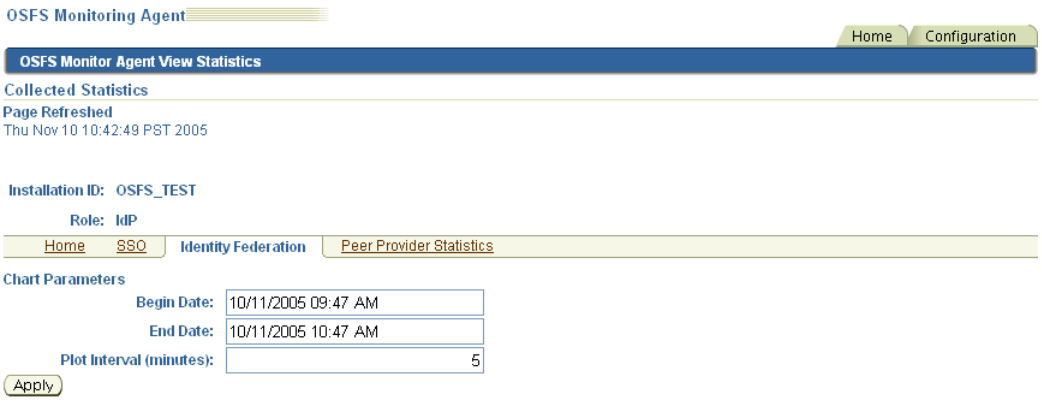
Monitor Agent IdP Statistics (Identity Federation)

This page displays Register NameID and Federation Termination requests sent to and received by an identity provider in a specified period. The server instance ID and the role (IdP) are displayed at the top of the page.

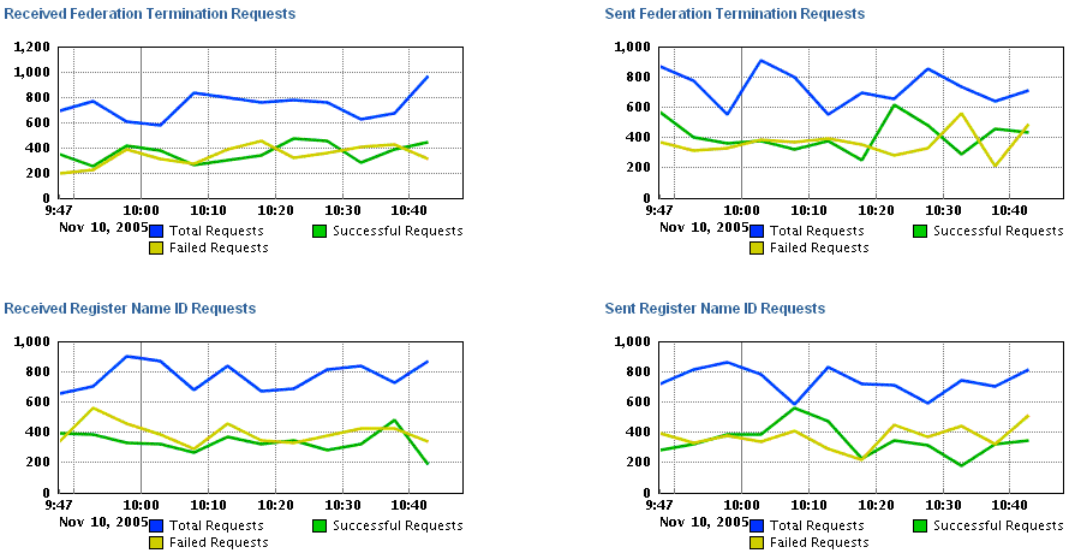
Fields at the top of the page let you control chart parameters:

- **Begin Date** is the start date and time of the monitored period.
- **End Date** is the end date and time of the monitored period.
- **Plot Interval** is the interval, in minutes, to use for the chart's horizontal axis.

Click the **Apply** button to refresh the display using the specified parameters.



The charts show Register NameID and Federation Termination requests sent and received in the period.



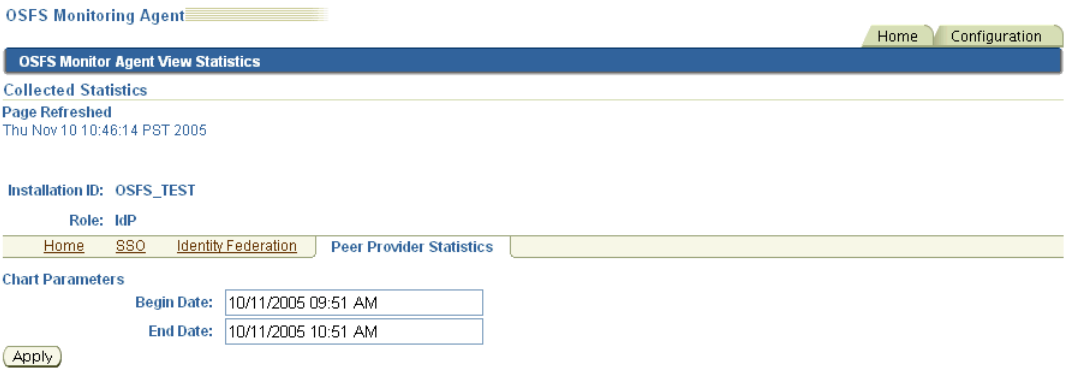
Monitor Agent IdP Statistics (Peer Provider)

This page displays summary statistics about requests exchanged between an identity provider and peer providers in the circle of trust in a specified period. The server instance ID and the role (IdP) are displayed at the top of the page.

Fields at the top of the page let you control chart parameters:

- **Begin Date** is the start date and time of the monitored period.
- **End Date** is the end date and time of the monitored period.

Click the **Apply** button to refresh the display using the specified parameters.



The display includes this information about requests exchanged with peer providers:

- Federation termination requests sent and received.
- NameID requests sent and received.
- Authentication requests received.

Received Federation Termination Requests

Peer ProviderID	Total	Successful	Failed
http://spC.oracle.com	2682	1010	1185
http://spA.oracle.com	2649	1317	1508
http://spB.oracle.com	2905	1669	1295

Sent Federation Termination Requests

Peer ProviderID	Total	Successful	Failed
http://spC.oracle.com	2529	1372	1436
http://spA.oracle.com	2576	1576	1581
http://spB.oracle.com	2652	1360	1035

Received Register Name ID Requests

Peer ProviderID	Total	Successful	Failed
http://spC.oracle.com	2853	1223	1330
http://spA.oracle.com	3010	1147	1402
http://spB.oracle.com	2945	1435	1530

Sent Register Name ID Requests

Peer ProviderID	Total	Successful	Failed
http://spC.oracle.com	2631	1081	1480
http://spA.oracle.com	2416	1319	1351
http://spB.oracle.com	2925	1520	1423

Received Authentication Requests

Peer ProviderID	Total	Successful	Failed
http://spC.oracle.com	2547	0	0
http://spA.oracle.com	2951	0	0
http://spB.oracle.com	2412	0	0

Monitor Agent SP Statistics Home

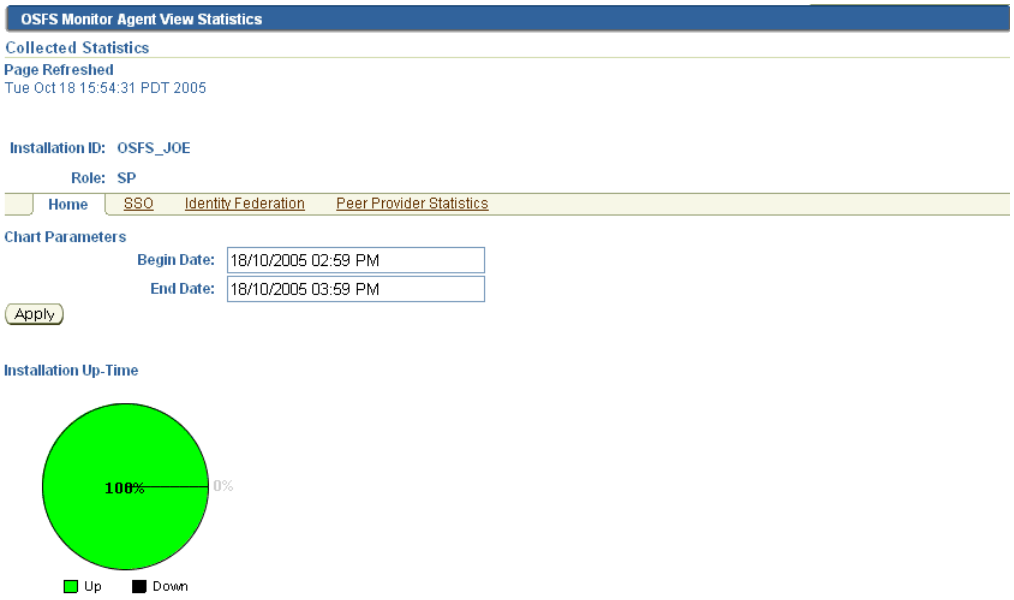
This is the home page for viewing service provider statistics for an installation.

Fields at the top of the page let you control chart parameters:

- **Begin Date** is the start date and time of the monitored period.

- **End Date** is the end date and time of the monitored period.

Click the **Apply** button to refresh the display using the specified parameters.



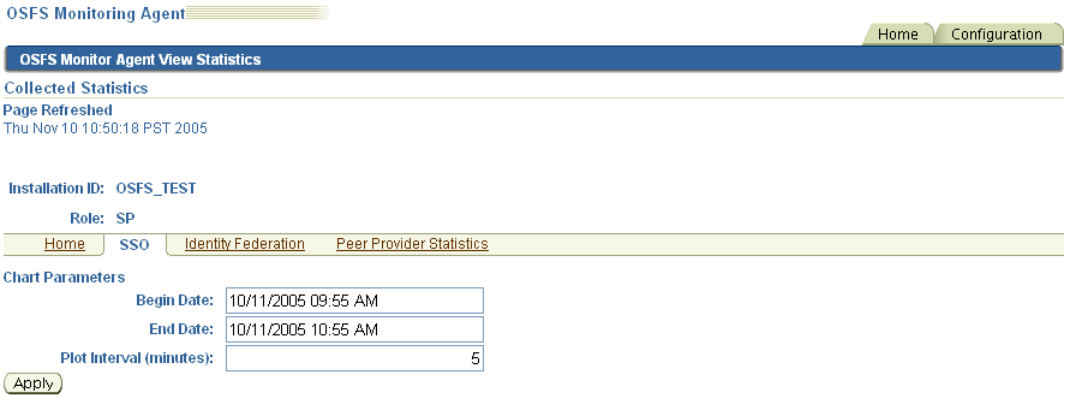
Monitor Agent SP Statistics (SSO)

This page displays authentication requests sent by a service provider in a specified period. The server instance ID and the role (SP) are displayed at the top of the page.

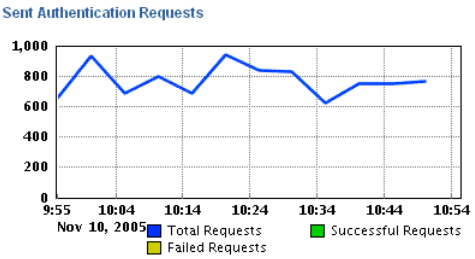
Fields at the top of the page let you control chart parameters:

- **Begin Date** is the start date and time of the monitored period.
- **End Date** is the end date and time of the monitored period.
- **Plot Interval** is the interval, in minutes, to use for the chart's horizontal axis.

Click the **Apply** button to refresh the display using the specified parameters.



The chart shows total, successful, and failed requests in the period.



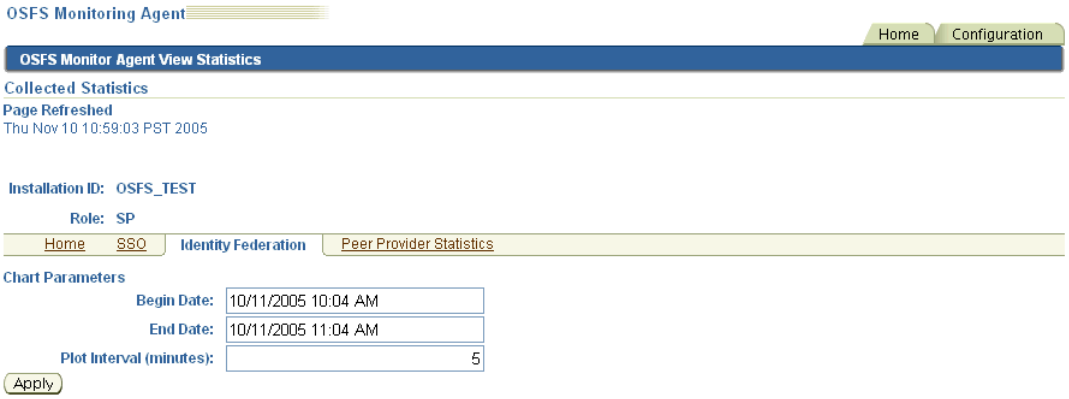
Monitor Agent SP Statistics (Identity Federation)

This page displays Register NameID and Federation Termination requests sent to and received by a service provider in a specified period. The server instance ID and the role (SP) are displayed at the top of the page.

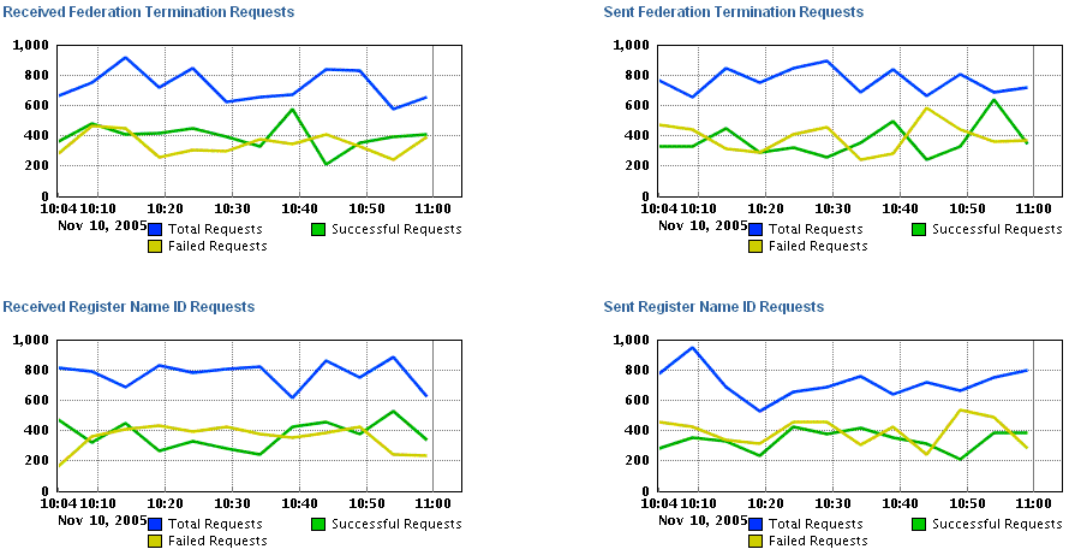
Fields at the top of the page let you control chart parameters:

- **Begin Date** is the start date and time of the monitored period.
- **End Date** is the end date and time of the monitored period.
- **Plot Interval** is the interval, in minutes, to use for the chart's horizontal axis.

Click the **Apply** button to refresh the display using the specified parameters.



The charts show Register NameID and Federation Termination requests sent and received in the period.



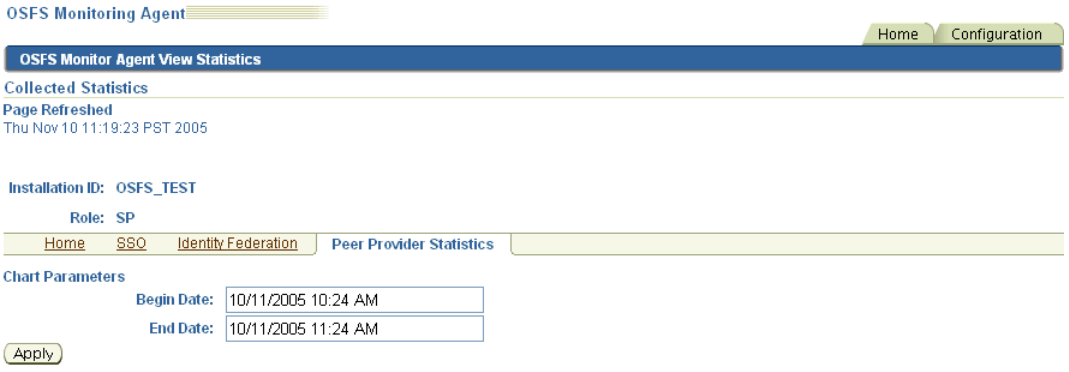
Monitor Agent SP Statistics (Peer Provider)

This page displays summary statistics about requests exchanged between a service provider and peer providers in the circle of trust in a specified period. The server instance ID and the role (SP) are displayed at the top of the page.

Fields at the top of the page let you control chart parameters:

- **Begin Date** is the start date and time of the monitored period.
- **End Date** is the end date and time of the monitored period.

Click the **Apply** button to refresh the display using the specified parameters.



The display includes this information about requests exchanged with peer providers:

- Federation termination requests sent and received
- NameID requests sent and received
- Authentication requests sent

Received Federation Termination Requests

Peer ProviderID	Total	Successful	Failed
idpC.oracle.com	2976	1728	1292
http://idpA.oracle.com	2454	1275	1284
http://idpB.oracle.com	2578	1055	1405

Sent Federation Termination Requests

Peer ProviderID	Total	Successful	Failed
idpC.oracle.com	2540	1749	1431
http://idpA.oracle.com	2696	1204	1399
http://idpB.oracle.com	2683	1570	1535

Received Register Name ID Requests

Peer ProviderID	Total	Successful	Failed
idpC.oracle.com	2818	1086	1390
http://idpA.oracle.com	2860	1324	1414
http://idpB.oracle.com	2998	1367	1223

Sent Register Name ID Requests

Peer ProviderID	Total	Successful	Failed
idpC.oracle.com	2634	1317	1299
http://idpA.oracle.com	2319	1179	1507
http://idpB.oracle.com	2831	1414	1491

Sent Authentication Requests

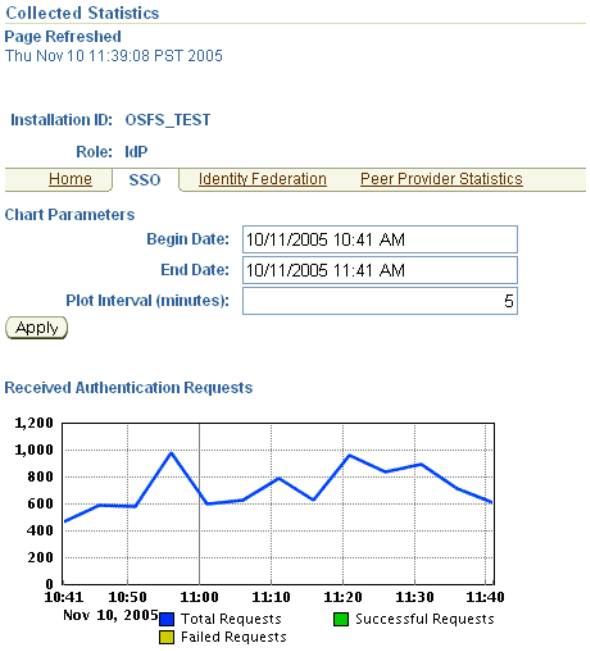
Peer ProviderID	Total	Successful	Failed
idpC.oracle.com	2709	0	0
http://idpA.oracle.com	2586	0	0
http://idpB.oracle.com	2763	0	0

Metric Display at the Console

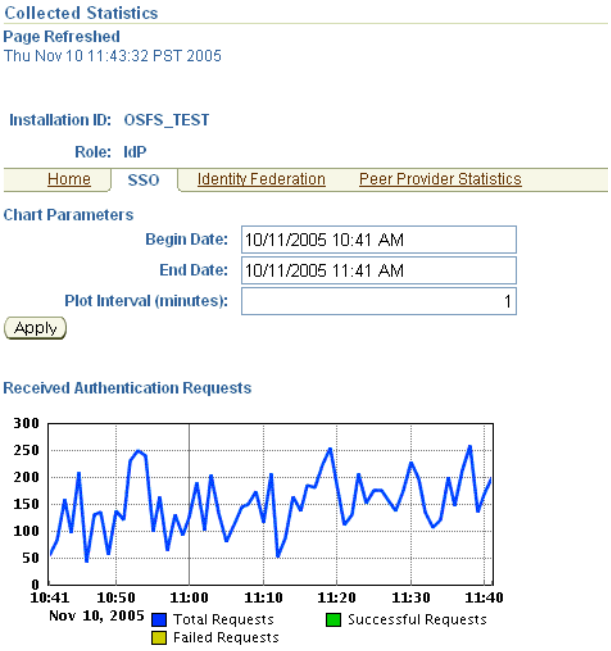
The metrics display at the Monitoring Console can be controlled in these ways:

- Refreshing the browser.
- Clicking the **Apply** button located under the Chart Parameters section of the display.
- Changing the Begin and End Date chart parameters to vary the period included in the display, and clicking **Apply**.
- Changing the Plot Interval chart parameter on time series charts to change the chart granularity.

For example, here is a display of authentication requests received at an IdP, using a 5 minute plot interval:



And the same chart using a 1 minute plot interval:



Managing Monitored Installations

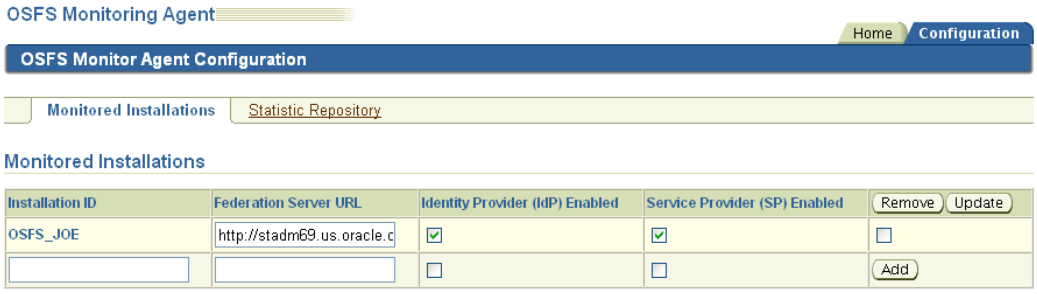
The Configuration tab of the Oracle Identity Federation Monitoring Agent allows you to monitor additional server instances and to maintain currently monitored installations.

You configure the monitoring agent with these pages:

- Monitored Installations
- Statistics Repository

Monitored Installations

The Monitored Installations page displays the Oracle Identity Federation instances being monitored by the Monitoring Agent.



Installation ID

This is the server's installation ID. Any user-friendly identifier can be chosen, since the entry does not need to correspond to any configured value such as a server ID.

Federation Server URL

This is the Oracle Identity Federation server URL, and is based on the following template:

```
http(s)://hostname:port
```

Identity Provider (IdP) Enabled

Indicates whether IdP monitoring of this instance is enabled at the Monitoring Agent site.

Service Provider (SP) Enabled

Indicates whether SP monitoring of this instance is enabled at the Monitoring Agent site.

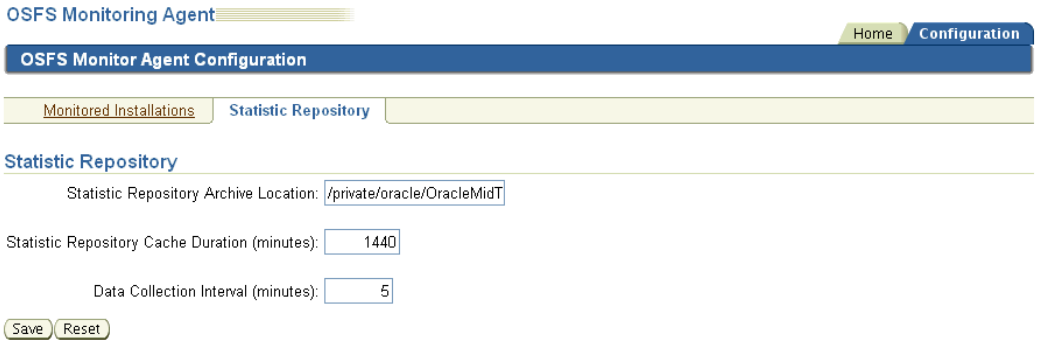
Actions

Buttons on this page provide the following functions:

- **Remove** - removes this server from the list of monitored servers.
- **Update** - updates the server information.
- **Add** - allows you to add another server to be monitored.

Statistics Repository

This page allows you to view and update information about the monitoring statistics repository.



Statistics Repository Archive Location

This is the location on disk where the repository resides.

Statistic Repository Cache Duration

This is the time, in minutes, that the repository data is maintained before being flushed from cache.

Data Collection Interval

This is the frequency, in minutes, at which the monitoring agent collects data for the monitored servers.

Actions

Buttons on this page provide the following functions:

- **Save** - updates the repository information.
- **Reset** - resets the original values that were displayed on the screen before you made any changes.

Archiving Metrics

To set up metrics archival, you use the Statistics Repository page of the Oracle Identity Federation Monitor Agent Configuration tab. The relevant fields are:

- **Statistics Repository Archive Location** - specify the location of the archive file on disk.
- **Statistic Repository Cache Duration** - specify the interval, in minutes, after which the data present in the cache is discarded.

> **Note:** Newly collected data is written simultaneously to both the cache (memory) and the archive (disk), but data is never transferred from the cache to the archive.

- **Data Collection Interval** - enter the frequency, in minutes, at which the monitoring agent should collect data for the monitored servers.

9

Advanced Topics

This chapter describes additional Oracle Identity Federation administration topics, including:

- Configuration Assistants
- Command-line Tools
- Managing Oracle Identity Federation Performance
- High Availability
- Setting Up a Load Balancer with Oracle Identity Federation
- Setting Up a Proxy for Oracle Identity Federation

Configuration Assistants

Oracle Identity Federation provides configuration assistants to support the following tasks:

- modification of the transient data store
- server uninstallation

Configuration Assistant for Changing the Transient Data Store

This Configuration Assistant allows you to change the type of transient data store used by your Oracle Identity Federation server. You can switch from a memory-based transient store to an RDBMS-based store, or from an RDBMS-based store to a memory-based store.

The Configuration Assistant performs these tasks:

1. Connects to the RDBMS to create Oracle Identity Federation database tables, if necessary
2. Packages the application with the new transient data store type
3. Redeploys Oracle Identity Federation
4. Updates the Oracle Identity Federation configuration to reflect the new transient data store type

For more information about these tasks, see "Configuration Assistant Operations".

See "Command-Line Configuration Assistant to Change the Transient Data Store" on page 9-9 for usage details.

Configuration Assistant for Uninstallation

A Configuration Assistant can also be invoked as a command-line tool to perform server deinstallation. See "Command-Line Configuration Assistant for Uninstallation" on page 9-9 for usage details.

This section contains these topics:

- Prerequisites for the Configuration Assistants
- Configuration Assistant Operations

Prerequisites for the Configuration Assistants

The following requirements must be met for the Configuration Assistants to execute properly:

- The files used by Oracle Identity Federation and the components it relies upon must be copied to their respective locations.
- OC4J must be installed and running.
- The deployment tool (`dcmctl`) must be installed and available to the Configuration Assistant.
- The Configuration Assistant must be run from the operating system account which has the correct privileges to access and configure OC4J.
- If Oracle Identity Federation is configured to use an LDAP repository to store the federation records, that server must be up and running.
- If configuring RDBMS as the new transient data store, a TNS Name must be defined locally referencing the database instance to be used. Refer to the Net Configuration Assistant to create a Local Net Service Name Configuration entry.

Before executing any tasks, the Configuration Assistant validates the parameters you specify on the command line. The assistant can:

- verify that the required parameters are present and have valid values
- check any connection parameters to LDAP or RDBMS servers by connecting to them

In case of a validation error, the Configuration Assistant displays an error message and returns a status code failure.

Configuration Assistant Operations

The Configuration Assistant performs a number of tasks related to Oracle Identity Federation maintenance:

- Repository Maintenance
- Deployment

Repository Maintenance

The Configuration Assistant configures and maintains back-end repositories needed for various tasks including federation records and artifact, message, and session information.

Deployment

After updating the `ear` application file with the configuration data supplied through Oracle Universal Installer, the Configuration Assistant deploys Oracle Identity Federation on the OC4J instance.

Command-line Tools

This section provides details about Oracle Identity Federation command-line tools:

- Bulk Federation Utility
- Command-Line Configuration Assistant to Change the Transient Data Store
- Command-Line Configuration Assistant for Uninstallation
- Command line Federation Delete Tool

Bulk Federation Utility

Oracle Identity Federation include a command-line tool that enables the administrator to perform bulk loading of user federation records.

By way of background, note that a federation represents an account linking between a user and two providers. Typically, the two providers agree to identify the individual using the data contained in the federation. When a federation is created, therefore, these providers will have agreed to use some specific piece of information to identify that user.

The bulk federation utility lets the administrator create an `ldif` file which can then be used to create federation records in an LDAP repository, and these federation records will be linked to users and peer providers.

The steps are as follows:

1. The utility reads instructions from an input file. For a specific federation record, the input data may contain the identifier of the user who will own the federation, the id of the peer provider with which the federation will be linked, and other parameters.
2. An `ldif` file containing LDAP commands is produced.
3. The administrator then executes the `ldif` file to populate the target LDAP server with federation records.

Thus the tool does not directly interact with the LDAP server, and the administrator is able to analyze the changes to the repository before bulk-loading federation data.

In addition to creating an `ldif` file for the LDAP server, the utility generates a result file that contains a list of all federations created. The administrator can then communicate this file to the peer provider for whom the federations are created, since the peer likewise needs to load the corresponding federation information in its repository.

Operating Modes of the Bulk Federation Utility

The bulk-loading tool operates in two modes:

- A *create* mode that uses a limited set of data (user id, peer provider id, and so on) to create:
  1. the federation data
  2. the corresponding LDAP commands

- A *read* mode that reads the input file containing the federation data; in this mode, the tool generates only the `ldif` file containing the LDAP commands, but no federation data is created.

Bulk Loading Flow

As an example, consider two providers named SP-A and IdP-B. Each provider has 10 users on its system, and the administrators wish to create federations for all of these users using the bulk federation load utility.

The steps they will take to achieve this are as follows:

1. One of the providers (for example, SP-A) creates an input file listing the federations to be created. For each federation data to be created, it contains: the user ID at SP-A, the remote provider ID (IdP-B), and some additional information.
2. The SP-A administrator next uses the bulk load tool in *create* mode; the tool reads the input file and creates two files: the `ldif` file containing LDAP commands for the SP-A federation LDAP server, and an output file of federation data.
3. The SP-A administrator uses the `ldif` file to populate SP-A's LDAP Federation server: at the completion of this step, the federation records have been created at SP-A.
4. At IdP-B, the administrator receives the output file and uses that file with the bulkload utility in *read* mode: the tool reads that file and creates an `ldif` file containing LDAP commands.
5. Finally, the IdP-B administrator executes the LDAP commands against its LDAP Federation server to create federation records in IdP-B: at that point, the federations are existing at both providers.

The Create Mode

In *create* mode, the tool generates new federation account-linking data based on the contents of an input file. For each federation record, the input file contains the following information:

Table 9–1 Fields of Input File in Create Mode of Bulk Load

Field	Description	Required?
UserID	This is the user identifier used by Oracle Identity Federation to uniquely reference a user in the user data store.	Yes
UserDescription	This is the human-readable user identifier that will be set on the LDAP federation record.	Yes
ProviderID	This is the identity of the peer provider with which the federation is created.	Yes
Version	This is the federation version. Specify 1.1, 1.2, or 2.0.	Yes

Table 9–1 (Cont.) Fields of Input File in Create Mode of Bulk Load

Field	Description	Required?
Type	This is the type of federation. The possible values are: ■ OIF_IDP_SP (Oracle Identity Federation as an IdP with an SP Peer Provider) ■ OIF_IDP_AFFILIATION (Oracle Identity Federation as an IdP with an affiliation or group of SPs) ■ OIF_SP_IDP (Oracle Identity Federation as an SP with an IdP Peer Provider) ■ OIF_AFFILIATION_IDP (Oracle Identity Federation as an affiliation or part of a group of SPs and an IdP Peer Provider)	Yes
AffiliationID	This is the AffiliationID in configurations where Oracle Identity Federation acts as an affiliation. *Note*: This is not the ProviderID of Oracle Identity Federation.	Yes, if Type is OIF_AFFILIATION_IDP

Here is a sample input file for use in the `create` mode:

```
UserID: user1b1
UserDescription: user1bb1
ProviderID: http://sp.oracle.com
Version: 2.0
Type: OIF_IDP_SP

UserID: user1b2
UserDescription: user1bb2
ProviderID: http://sp.oracle.com
Version: 2.0
Type: OIF_IDP_SP

UserID: user1b1
UserDescription: user1bb1
ProviderID: http://idp.oracle.com
Version: 2.0
Type: OIF_SP_IDP
```

The Read Mode

In *read* mode, the tool reads and processes existing federation account-linking data from the input file without creating any new federations. For each federation record the input file will contain the following information:

Table 9–2 Fields of Input File in Read Mode of Bulk Load

Field	Description	Required?
UserID	This is the user identifier used by Oracle Identity Federation to uniquely reference a user in the user data store.	Yes

Table 9–2 (Cont.) Fields of Input File in Read Mode of Bulk Load

Field	Description	Required?
UserDescription	This is the human-readable user identifier that will be set on the LDAP federation record.	Yes
ProviderID	This is the identity of the peer provider with which the federation is created.	Yes
Version	This is the federation version. Specify 1.1, 1.2, or 2.0.	Yes
Type	This is the type of federation. The possible values are: ■ OIF_IDP_SP (Oracle Identity Federation as an IdP with an SP Peer Provider) ■ OIF_IDP_AFFILIATION (Oracle Identity Federation as an IdP with an affiliation or group of SPs) ■ OIF_SP_IDP (Oracle Identity Federation as an SP with an IdP Peer Provider) ■ OIF_AFFILIATION_IDP (Oracle Identity Federation as an affiliation or part of a group of SPs and an IdP Peer Provider)	Yes
AffiliationID	This is the AffiliationID in configurations where Oracle Identity Federation acts as an affiliation. *Note*: This is not the ProviderID of Oracle Identity Federation.	Yes, if Type is OIF_AFFILIATION_IDP
IdPNameID	This is the federation's IdP Name Identifier.	Yes
SPNameID	This is the federation's SP Name Identifier.	No
Format	This is the format of the federation's Name Identifier.	No
Qualifier	This is the qualifier of the federation's Name Identifier.	No

Here is a sample input file for use in the `read` mode:

```
userid: user1b1
userdescription: user1bb1
providerid: http://sp.oracle.com
idpnameid: id-SHMeaEZkk4jqiHl32BmmIAcBy4o-
format: urn:oasis:names:tc:SAML:2.0:nameid-format:persistent
qualifier: http://sp.oracle.com
version: 2.0
type: OIF_IDP_SP

userid: user1b2
userdescription: user1bb2
providerid: http://sp.oracle.com
idpnameid: id-QTqtXv13IE493asxqGF-CoeBuRM-
format: urn:oasis:names:tc:SAML:2.0:nameid-format:persistent
qualifier: http://sp.oracle.com
version: 2.0
type: OIF_IDP_SP
```

```
userid: user1b1
userdescription: user1bb1
providerid: http://idp.oracle.com
idpnameid: id-He4Nvgeiz4BLzhRhlPzebViLQbU-
format: urn:oasis:names:tc:SAML:2.0:nameid-format:persistent
qualifier: http://stadm04.us.oracle.com:7780/fed/sp
version: 2.0
type: OIF_SP_IDP
```

Output Files Generated by Bulk Load

In both create and read modes, the tool produces three output files:

1. an `ldif` file
2. a result file
3. an error file

LDIF File

This file contains the LDAP statements an administrator will need to execute against the LDAP server. Executing the `ldif` file populates the server with user federation records. You typically achieve this by using an `ldapmodify` command of the form:

```
ldapmodify -a -c -x -D "cn=Admin,dc=example,dc=com" -w PASSWORD
    -h LDAP_HOSTNAME -p LDAP_PORT -f LDIF_FILE
```

Result File

The result file contains information about the federation records that were created. This file is typically communicated to third party providers, who use it to populate their federation repository with corresponding federation records.

See Table 9–2 on page 9-5 for a description of the result file.

Error File

The error file contains records of any input entries that could not be processed. The error record consists of:

- UserID - this is the identifier used by Oracle Identity Federation to uniquely reference a user in the data store.
- ProviderID - this is the identity of the peer provider with which the federation is created.

Syntax and Examples

The bulk load utility is invoked at the command line as follows:

java –jar bulkload.jar <parameters>

The parameters are:

- -f INPUT_FILE - This is the file with the information specifying the federation records to create.
- -dir OUTPUT_DIR - This is the directory in which to create the output files such as the `ldif` file, the result file and the error file.
- -oifdir OIF_DIR - This is the top-level Oracle Identity Federation directory. Typically it is the directory containing the bin, conf, lib and log directories.

- -c - This is the create mode. It specifies that federation records need to be created. The tool will generate IdPNameID, Format and Qualifier fields.
- -r - This is the read mode. It specifies that the federation records information is contained in the input file and no federation data should be created, instead it should be read from the file to produce the `ldif` and result files.

Example 1

java -jar fed/lib/bulkload.jar -f user-fed-data.dat -dir /saveDir -oifdir $ORACLE_HOME/fed -c

This command directs the tool to read a file containing a list of users for which federations are to be created, and produce 1) an `ldif` file that the administrator can execute against its LDAP server to create the records, and 2) another file that contains the federation information that should be sent to the peer providers for which the federations were created.

Example 2

java -jar fed/lib/bulkload.jar -f peer-user-fed-data.dat -dir /saveDir -oifdir $ORACLE_HOME/fed -r

This command directs the tool to read a file sent by a peer provider (a file that was created by the bulk load tool, for example), and to create an `ldif` file based on that input file that would contain commands to create federation records.

Example 3

This example brings together the concepts described earlier in this section, to illustrate all the steps to achieve a bulk load of federation users from beginning to end.

In this example, the input file of users is created at the IdP side, and the IdP federation server is populated. The result file is then used at the SP to generate the necessary data to populate the data records at the SP server.

1. Create the input file.

```
UserID: buser9
UserDescription: buser9
ProviderID: http://platinum-ob.us.oracle.com:7780/fed/sp
Version: 1.2
Type: OIF_IDP_SP

UserID: buser10
UserDescription: buser10
ProviderID: http://goofy.us.oracle.com:7785/fed/sp
Version: 1.2
Type: OIF_IDP_SP
```

2. Navigate to the `Oracle_Home\fed\lib` directory and generate the `result.ldif` and `result.txt` files. For example:

```
D:\osfshome2\jdk\bin\java -jar bulkload.jar -f users
   -dir D:\osfshome2\fed\lib -oifdir D:\osfshome2\fed -c
```

3. Import the `result.ldif` into the IdP directory. For example:

```
ldapmodify -h platinum-ob.us.oracle.com -a -f result.ldif
   -D "cn=orcladmin" -w malibu97 -p 13060
```

4. Copy the `result.txt` file to the SP, and modify the file to conform to the SP; this means changing the serverid, type, userid, userdescription, and providerid values.

5. At the SP, read the `result.txt` file into the bulk tool to generate an `ldif` file. For example:

```
C:\OraHome_030306>jdk\bin\java -jar fed\lib\bulkload.jar
   -f c:\orahome_030306\fed\lib\result.txt
   -dir c:\orahome_030306\fed
   -oifdir c:\orahome_030306\fed -r
```

6. Import the resulting `ldif` file into the directory at the SP:

```
ldapmodify -a -c -x -D "cn=Directory Manager" -w 97malibu
   -h akeim2.us.oracle.com -p 440 -f result.ldif
```

Users should now be able to do single sign-on without having to log in at the SP, because the federations already exist.

Command-Line Configuration Assistant to Change the Transient Data Store

For a description of this utility, see "Configuration Assistants" on page 9-1.

Syntax and Examples

The installation configuration assistant is invoked at the command line as follows (the tool will prompt for the RDBMS password):

java -jar $ORACLE_HOME/fed/lib/install.jar <parameters>

The parameters are:

- -oh $ORACLE_HOME - This is the location of the ORACLE_HOME variable referencing the Oracle Application Server installation directory. Required.
- -transient <TYPE> This is the type of repository to use as the transient data store in Oracle Identity Federation. The possible values are memory or rdbms. Required.
- -dbtnsname <TNSNAME> - This is the RDBMS TNS name. Required if an RDBMS is used for the transient data store.
- -dbusername <USERNAME> - This is the RDBMS username. Required if an RDBMS is used for the transient data store.
- -uselocalconfig <BOOLEAN> - This is an optional parameter that can be used with -transient rdbms, to control whether this instance's local configuration settings will overwrite any settings currently stored in the database. The possible values are "true" or "false". If "true", configuration settings in the Oracle Identity Federation server's local files will replace any settings currently stored in the selected database. If "false", or this parameter is omitted, settings stored in the RDBMS (if there are any) will overwrite the local configuration files. The default is "false".

  As mentioned earlier, this is an optional parameter that can be specified if an RDBMS is used for the transient data store.

Requirements

Prior to running the configuration assistant, set the following environment variables:

- $ORACLE_HOME: environment variable referencing the Oracle Home directory. For example:
  - *Windows*

    ```
    set ORACLE_HOME=c:\DIR
    ```

  - *Linux bash shell*

```
export ORACLE_HOME=/DIR
```

- On Linux, you need to set the `LD_LIBRARY_PATH` to include the `$ORACLE_HOME/lib` directory. This assumes that the `ORACLE_HOME` environment variable is already set. For example:

  ```
  export LD_LIBRARY_PATH=$ORACLE_HOME/lib
  ```

- On Windows, you need to set the `PATH` environment variable to include the `%ORACLE_HOME%/lib` and `%ORACLE_HOME%/bin` directories. This assumes that the `ORACLE_HOME` environment variable is already set. For example:

  ```
  set PATH=%ORACLE_HOME%/bin;ORACLE_HOME%/lib;%path%
  ```

Example 1

$ORACLE_HOME/jdk/bin/java -jar $ORACLE_HOME/fed/lib/install.jar -oh ORACLE_HOME -transient memory

Note that the transient data will be stored in memory.

Example 2

$ORACLE_HOME/jdk/bin/java -jar $ORACLE_HOME/fed/lib/install.jar -oh ORACLE_HOME -transient rdbms -dbusername USERNAME -dbtnsname TNSNAME

Note that the transient data will be stored in RDBMS.

Command-Line Configuration Assistant for Uninstallation

The configuration assistant provides an uninstall feature to perform the following operations:

- Destroy all federation records created by the Oracle Identity Federation server being un-installed.
- Destroy all auxiliary and non sensitive data if all federation records were destroyed in the first step. Oracle Identity Federation is the only server using the LDAP federation server to store data, so it is safe to destroy the structural data.
- Remove the attributes and object classes that the configuration assistant added to the LDAP schema at installation time.

> **Note:** This feature is not supported for Microsoft Active Directory as the directory does not allow entries to be removed from its schema.

- Destroy the RDBMS transient tables, if any, used to store persistent data.

The configuration assistant notifies the un-installer if:

- Oracle Identity Federation used an RDBMS database to store persistent data
- An LDAP store was used to store federation records

If an LDAP or RDBMS operation needs to be performed, the uninstallation tool would prompt for LDAP or RDBMS connection information that is then passed to the configuration assistant. That way, the administrator can specify the correct information and credentials required to perform administrative tasks in the LDAP or RDBMS server.

See Also: For complete Oracle Identity Federation uninstallation procedures, see "Un-installing Oracle Identity Federation" on page 5-15

Syntax and Examples

The uninstallation configuration assistant is invoked at the command line as follows (the tool will prompt for the LDAP/RDBMS password):

```
java -jar $ORACLE_HOME/fed/lib/uninstall.jar <parameters>
```

The parameters are:

- -uninstall - Launches the uninstallation process. Required.
- -oh ORACLE_HOME - This is the location of the ORACLE_HOME variable referencing the OracleAS installation directory. Required.
- -removefed true|false - Indicates whether to remove federation records from the LDAP server.
- -ldap true|false - Indicates whether to remove the Oracle Identity Federation schema from the LDAP server.
- -ldaptype - Indicates the type of repository. The possible values are `oid`, `msad`, `iplanet`, and `tivoli`.
- -ldapurl - This is the LDAP URL.
- -ldapusername - This is the username of the LDAP administrator.
- -db - Indicates whether to remove transient Oracle Identity Federation tables from the RDBMS server.
- -dbtnsname <TNSNAME> - This is the RDBMS TNS name. Required if -db is `true`.
- -dbusername <USERNAME> - This is the RDBMS username. Required if -db is `true`.

Note: To see a list of parameters and possible values, run the tool without specifying any parameters.

Requirements

If you plan to remove the RDBMS tables created by Oracle Identity Federation, take these steps prior to running the uninstallation tool:

- Set environment variables.

  On Linux

  Set the `LD_LIBRARY_PATH` variable to include the `$ORACLE_HOME/lib` directory. This assumes that the `ORACLE_HOME` environment variable is already set. For example:

  ```
  export LD_LIBRARY_PATH=$ORACLE_HOME/lib
  ```

  On Windows

  Set the `PATH` environment variable to include the `%ORACLE_HOME%/lib` and `%ORACLE_HOME%/bin` directories. This assumes that the `ORACLE_HOME` environment variable is already set. For example:

  ```
  set PATH=%ORACLE_HOME%/bin;ORACLE_HOME%/lib;%path%
  ```

- Retrieve the TNS Name referencing the RDBMS. This value, which you can find in the `$ORACLE_HOME/network/admin/tnsnames.ora` file, is the identifier referencing the connection information used to connect to the database.

Example 1

java -jar fed/lib/uninstall.jar -uninstall -oh $ORACLE_HOME -removefed true -ldap false -ldapurl ldap://ldap.com -ldapusername admin -db false

This command removes all the federation records from the LDAP server.

Example 2

java -jar fed/lib/uninstall.jar -uninstall -oh ORACLE_HOME -removefed true -ldap true -ldaptype oid -ldapurl ldap://ldap.com -ldapusername admin -db false

This command:

- removes all the federation records from the LDAP server
- removes the Oracle Identity Federation schema from the LDAP server

Example 3

java -jar fed/lib/uninstall.jar -uninstall -oh ORACLE_HOME -removefed true -ldap true -ldaptype oid -ldapurl ldap://ldap.com -ldapusername admin -db true -dbtnsname orcl -dbusername scott

This command:

- removes all the federation records from the LDAP server
- removes the Oracle Identity Federation schema from the LDAP server
- removes the transient Oracle Identity Federation tables from the RDBMS server

Command line Federation Delete Tool

With the 10.1.4.2.0 patch set, Oracle Identity Federation provides a new command line tool to perform bulk delete operations of federation records from the Federation Data Store.

This tool removes federation records directly from the LDAP server used as the Federation Data Store without notifying the remote provider; thus, the Liberty 1.x/SAML 2.0 protocols will not be exercised to notify the provider that the federation record was locally deleted.

Use the tool to remove federation records when:

- the remote provider no longer exists
- the remote provider is not in the circle of trust
- the remote provider does not support the Federation Termination protocols
- the local administrator wishes to remove federation records without notifying the remote provider

---

Note: To remove a federation record **and** notify the remote provider, use the Identity Federation tab in the Oracle Identity Federation Administration Console to delete the federation: that operation will connect to the remote provider to indicate that the record is being deleted.

You can use the federation delete tool in different ways:

- to remove all federation records of a specific user
- to remove all federation records linked to a specific remote provider
- to remove all federation records of a list of users

Examples of each type of usage appear below.

Syntax and Examples

Invoke the federation delete tool at the command line as follows (the LDAP password is entered during command execution):

```
java -jar $ORACLE_HOME/fed/lib/fedadmin.jar <parameters>
```

The parameters are:

- -oh *ORACLE_HOME* - This is the ORACLE_HOME directory. Required.
- -ldapurl *url* - the URL of the LDAP Federation Data Store
- -ldapusername *username* - the LDAP username to use to connect to the LDAP Federation Data Store
- -providerid *id* - the Provider ID for which federations need to be removed
- -userid *id* - the user ID for which federations need to be removed
- -useridfile *file* - absolute path to the file containing the list of user IDs for which federations need to be removed

Example 1

To remove all federation records of a specific user, the administrator specifies the identifier of the user.

For example, the following command deletes all federation records belonging to the user `alice`:

```
java -jar $ORACLE_HOME/fed/lib/fedadmin.jar -oh $ORACLE_HOME -ldapurl ldap://ldap.com
-ldapusername ADMIN -userid alice
```

> **Note:** The identifier value must be of the same type as the **User ID Attribute** configured in the User Data Store section of the Oracle Identity Federation Administration Console.

Example 2

To remove all federation records linked to a specific remote provider, the administrator specifies the Provider ID of the remote provider.

For example, the following command deletes all federation records linked to the provider identified by `http://idp.com/idp`:

```
java -jar $ORACLE_HOME/fed/lib/fedadmin.jar -oh $ORACLE_HOME -ldapurl ldap://ldap.com
-ldapusername ADMIN -providerid http://idp.com/idp
```

Example 3

To remove all federation records of a list of users, the administrator specifies a file containing identifiers of the users:

For example, the following command:

```
java -jar $ORACLE_HOME/fed/lib/fedadmin.jar -oh $ORACLE_HOME -ldapurl ldap://ldap.com
-ldapusername ADMIN -useridfile users.txt
```

where the file users.txt contains:

```
alice
bob
charlie
```

deletes all federation records belonging to users `alice`, `bob`, and `charlie`.

> **Note:** The identifier values must be of the same type as the **User ID Attribute** configured in the User Data Store section of the Oracle Identity Federation Administration Console.

Managing Oracle Identity Federation Performance

This section explains topics relevant to Oracle Identity Federation server performance, including:

- Setting Concurrent Connection Limits
- Setting JDBC Connection Limits
- Tuning Oracle HTTP Server

> **See Also:** *Oracle Application Server Performance Guide*

Setting Concurrent Connection Limits

Communication between two federation servers occurs by means of the SOAP-over-HTTP protocol. Oracle Identity Federation provides these default connection limits:

- The number of concurrent connections to a single host is limited to 4.0
- The number of maximum total connections is limited to 100.

You can override the default limits by setting the relevant connection parameters:

1. Log on to the Enterprise Manager console.
2. At the console, navigate to **OC4J_FED** -> **Administration** -> **Server Properties**.
3. Add these entries to the Java Options configuration:

   ```
   -Dhttp.fed.host=VALUE1
   -Dhttp.fed.max.conn=VALUE2
   ```

   where `VALUE1` and `VALUE2` are values set by the administrator.
4. Restart OC4J_FED.

Setting JDBC Connection Limits

When Oracle Identity Federation uses a database as its transient data store, the server uses a connection manager to open and maintain SQL connections to the database.

By default, connection manager uses the following settings when working with an RDBMS as the transient data store:

JDBC Connection Setting	Default Value
Minimum number of connections	1
Maximum number of connections	150
Maximum inactivity timeout (the connection will be closed if inactive after this period)	7200 seconds
Connection wait timeout (maximum time to wait for a new connection)	30 seconds

The administrator can override these settings using these steps:

1. Log on to the Enterprise Manager console.
2. At the console, navigate to **OC4J_FED** - > **Administration** - > **Server Properties**
3. Add the following entries to the Java Options configuration corresponding to the settings:
   - `-Dfed.jdbc.min.conn=`*VALUE* to set the minimum number of connections
   - `-Dfed.jdbc.max.conn=`*VALUE* to set the maximum number of connections
   - `-Dfed.jdbc.max.usage=`*VALUE* to set the maximum inactivity timeout in seconds
   - `-Dfed.jdbc.conn.timeout=`*VALUE* to set the connection wait timeout in seconds
4. Save the settings and restart OC4J_FED.

Tuning Oracle HTTP Server

By default, Oracle HTTP Server is configured for general use cases.

To efficiently handle heavy loads, you may need to tune Oracle HTTP Server for improved performance. To achieve this, open the `ORACLE_HOME/Apache/Apache/conf/httpd.conf` file and modify the following properties:

> **Note:** The values given here are for illustration only. The actual values you employ will vary depending on your machine and operating system configuration.

Setting	Example Value	Meaning
`KeepAlive`	off	Turns off the `KeepAlive` feature. When `KeepAlive` is on, connections are not freed until timeout occurs.
`MaxClients`	512	Increases the number of clients that Apache can process concurrently.
`MinSpareServers`	20	This is the minimum number of processes Apache can use.

Setting	Example Value	Meaning
`MaxSpareServers`	250	This is the maximum number of processes Apache can use. *Caution*: This setting can be memory-intensive.
`StartServers`	100	This is the number of processes to create at startup. *Caution*: This setting can be memory-intensive.

Note: You may also need to optimize the IdM framework that is integrated with Oracle Identity Federation (Oracle SSO, Oracle Access Manager, or SiteMinder) with respect to HTTP connections.

Finally, you should also take into account the IdM components that are integrated with the Oracle Identity Federation server, and tune their settings appropriately. For example, tuning Oracle HTTP Server settings for OracleAS Single Sign-On may be helpful in optimizing the performance of the federation server.

High Availability

Oracle Identity Federation leverages Oracle Application Server to support several strategies for ensuring high availability. These strategies can be employed both within an application server instance and across a cluster that includes multiple application server instances. Oracle Identity Federation supports the **Cold Failover Cluster** (CFC) or active-passive high availability configuration, where the Oracle Application Server Infrastructure is typically deployed on a two-node hardware cluster with a shared storage device. This configuration requires the use of an RDBMS as the transient data store.

Note: To support High Availability configurations, the "Advanced Install" option *must* be selected during Oracle Identity Federation installation. For more information, see "Advanced Installation Procedure" on page 3-7.

Key features of Oracle Identity Federation that impact high availability are described in these sections:

- Web Application Session State Replication
- Centralized Storage of Configuration Information
- Data Tier
- Additional Information

Web Application Session State Replication

Oracle Identity Federation is a web application deployed to OC4J (in the OC4J_FED container), and multiple HTTP requests from the same client may need to access the application. However, if Oracle Identity Federation running on the OC4J server experiences a problem resulting in failure of the OC4J process or instance, the state

associated with a client request may be lost. Here are some ways to guard against such failures:

1. Using an RDBMS as the transient data store

   In this configuration, Oracle Identity Federation saves the transient session and profile state in the database. When the active server instance goes down, the passive (standby) server instance is brought online and restores session and profile state using information from the database, thus avoiding the loss of the Oracle Identity Federation state due to server failure. Storing transient data in the RDBMS is a requirement for high availability. You must use this feature with Oracle Identity Federation on multiple OC4J processes or instances.

2. Running on Multiple OC4J Processes

   In this configuration, the OC4J instance on which Oracle Identity Federation runs is configured for multiple OC4J processes. The OC4J instance configuration is valid for one or more OC4J processes, and these processes communicate the Web session state to each other, thereby providing protection against software problems like an OC4J process failure or hang.

3. Running on Multiple OC4J Instances

   In this configuration, multiple active instances of Oracle Identity Federation are deployed to multiple OC4J instances. The Web session state must be replicated across the server instances using the replication facilities provided by the OC4J clustering framework, and the state is managed using Oracle Enterprise Manager Application Server Control.

If using load balancers, it is important to ensure that all Oracle Identity Federation sessions are directed to the same server instance.

See Also: "Setting Up a Load Balancer with Oracle Identity Federation" on page 9-18.

Centralized Storage of Configuration Information

Oracle Identity Federation stores configuration information centrally in the RDBMS transient data store when configured for such a store. This ensures that if configuration changes occur in one federation server instance, all other server instances are able to use the most up-to-date configuration settings.

Note: It takes approximately 10 seconds for the changes to propagate to all active server instances.

When using SAML 1.x or WS-Federation protocols, you will need to manually replicate the `ORACLE_HOME/fed/shareid/oblix/config/keystore` file to the different Oracle Identity Federation instances. This file contains the peer providers' certificate, which is used to verify signatures in SAML 1.x/WS-Federation protocol messages.

Data Tier

In order to ensure high availability of Oracle Identity Federation, it is necessary to ensure high availability of the specific data repositories that are being used with the server (LDAP, RDBMS, and so on). Refer to product- and vendor-specific documentation for details about configuring specific data stores.

For RDBMS repositories, a good starting point is the *Oracle Database High Availability Overview*, which is part of the Administration document set in the Oracle9*i* Database Server documentation library.

Configuring Redundant LDAP Servers

If your site contains a redundant infrastructure of LDAP servers, for example a topology where the servers are fronted by a load balancer that re-routes requests to a different LDAP server when a directory instance goes down, the Oracle Identity Federation server should be configured using these steps:

1. Log on to the Oracle Enterprise Manager console.
2. At the console, navigate to **OC4J_FED** - > **Administration** - > **Server Properties**.
3. Add the following entry to the Java Options field:

   ```
   -Dfed.ldap.ha=true
   ```

4. Save the settings and restart OC4J_FED.

Additional Information

For more information about high availability features that you can leverage in your Oracle Identity Federation installation, see the *Oracle Application Server High Availability Guide*.

For information about Cold Failover Cluster configurations for high availability, see the *Oracle Application Server High Availability Guide* in the 10g Release 2 (10.1.2) documentation library.

Setting Up a Load Balancer with Oracle Identity Federation

This section explains configuration and other steps required for installing Oracle Identity Federation behind a load balancer.

Take these steps to set up a load balancer with Oracle Identity Federation:

> **Note:** These steps are specific to the F5 load balancer.

1. Follow the standard installation process to install all Oracle Identity Federation instances on corresponding load balanced host machines. Details are provided in Chapter 3, "Installing Oracle Identity Federation".
2. There is no need to choose a virtual hostname, as it is required for Cold Failover Cluster in the advanced installation procedure.
3. Choose the transient data store in the advanced installation procedure.
4. Install all Oracle Identity Federation instances pointing to the same transient data store.
5. In the F5 load balancer administration console, create a pool with all Oracle Identity Federation instances, and enable the application persistence property for this pool. Set the persistence type to `Active HTTP Cookie`, set the method to `insert`, and the set expiration to the desired value.
6. Create a virtual server member mapped to the newly created pool.
7. Enable HTTP monitoring on all member nodes in the pool.

8. On all Oracle Identity Federation servers instances, change the ServerName and Port parameters in the `httpd.conf` file to the Virtual Server name and port.
9. For all Oracle Identity Federation servers that are both load-balanced *and* integrated with OracleAS Single Sign-On, register the load balancer URL (for example, `http://lbr.us.oracle.com:80`) to all load-balanced Oracle Identity Federation servers by running this command from the command prompt:

```
<OIF_HOME>/sso/bin/ssoreg.bat -oracle_home_path %ORACLE_HOME% -site_name
    <Load_Balancer_Host_Name> -config_mod_osso TRUE -mod_osso_url
    <Load_Balancer_URL:Port>
```

10. On all Oracle Identity Federation servers, update the configuration by running this command from the command prompt:

```
dcmctl updateconfig
```

11. Restart the HTTP server from the Oracle Enterprise Manager console on all the Oracle Identity Federation server instances.
12. In the Oracle Identity Federation administration console, under **Server Configuration** - > **Server Properties**, change the server hostname and port number to the Virtual Server name and port number of the load balancer.
13. Restart the Oracle Identity Federation instances from the Oracle Enterprise Manager console.
14. Distribute the new metadata file to the peer providers.

Additional Considerations for SAML 1.x or WS-Federation

When using SAML 1.x or WS-Federation protocols, you will need to manually replicate the `ORACLE_HOME/fed/shareid/oblix/config/keystore` file to the different Oracle Identity Federation instances. This file contains the peer providers' certificate, which is used to verify signatures in SAML 1.x/WS-Federation protocol messages.

Additionally, when setting up SAML 1.x or WS-Federation SSO in load-balanced mode, make sure that the URLs in MyDomain at the Load-balanced provider, and in the load-balanced Domain at the other provider, have the hostname and address of the load-balancer machine (as opposed to, for example, the hostname and address of one of the load-balanced Oracle Identity Federation instances).

Additional Steps for the Oracle Identity Federation Monitoring console

The Oracle Identity Federation monitoring console cannot be load balanced; this is because only one server should be responsible for monitoring one or more Oracle Identity Federation instances. Therefore, you need to modify the Oracle HTTP Server settings to correctly enable the monitoring features on the console, as well as on the Oracle Identity Federation servers.

Make sure that the HTTP ports and URLs used for monitoring operations are not integrated with the load balancer. You can accomplish this by creating an Oracle HTTP Server Virtual host that will forward requests made to the `/fed` and `/fedmon` URLs to the local OC4J_FED container.

The new virtual host will need these settings:

- `ServerName`, `Port`, and `Listen` directive values for the virtual host set to the local values

- Oc4jMount directives that forward `/fed` and `/fedmon` URLs to the OC4J_FED container

For example, the definition of such a virtual host in `ORACLE_HOME/Apache/Apache/conf/httpd.conf` could look like this:

```
Listen 7799
NameVirtualHost *:7799
<VirtualHost *:7799>
    ServerName local.machine.com
    <IfModule mod_oc4j.c>
        Oc4jMount /fed OC4J_FED
        Oc4jMount /fed/* OC4J_FED
        Oc4jMount /fedmon OC4J_FED
        Oc4jMount /fedmon/* OC4J_FED
    </IfModule>
</VirtualHost>
```

Take these steps:

1. Modify the `httpd.conf` file using the example as a guide.
2. Execute the command:

   ```
   dcm/bin/dcmctl updateconfig
   ```

3. Restart Oracle HTTP Server:

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

4. After creating the virtual host, access the monitoring console using the URL of the virtual host.
5. Add the URLs of Oracle Identity Federation servers to monitor. The URL should point to the local virtual host value, not the load-balanced URL.

The monitoring console will now be ready to configure to monitor the Oracle Identity Federation server.

Setting Up a Proxy for Oracle Identity Federation

This section explains how to set up a proxy server for Oracle Identity Federation.

Due to the standalone nature of Oracle Identity Federation, you cannot utilize the usual procedures for setting up an application server proxy. Instead, use the steps provided here to set up an Oracle HTTP Server as a proxy for Oracle Identity Federation:

1. Install Oracle HTTP Server for the proxy server.
2. Edit the `ORACLE_HOME/Apache/Apache/conf/httpd.conf` file.
   a. Uncomment these lines:

      ```
      <IfModule mod_proxy.c>
          ProxyRequests On
      </IfModule>
      ```

   b. Unset the `ProxyRequests` directive:

      ```
          ProxyRequests Off
      ```

   c. Add these directives after `ProxyRequests Off`:

      ```
      ProxyPass /sso HTTP://OIF-HOST:OIF-PORT/sso
      ```

```
ProxyPass /fed HTTP://OIF-HOST:OIF-PORT/fed
ProxyPass /shareid HTTP://OIF-HOST:OIF-PORT/shareid
ProxyPassReverse /sso HTTP://OIF-HOST:OIF-PORT/sso
ProxyPassReverse /fed HTTP://OIF-HOST:OIF-PORT/fed
ProxyPassReverse /shareid HTTP://OIF-HOST:OIF-PORT/shareid

#ProxyPass /fedadmin HTTP://OIF-HOST:OIF-PORT/fedadmin
#ProxyPass /fedmon HTTP://OIF-HOST:OIF-PORT/fedmon
#ProxyPass /uixi http://OIF-HOST:OIF-NON-SSL-PORT/uixi
#ProxyPassReverse /fedadmin HTTP://OIF-HOST:OIF-PORT/fedadmin
#ProxyPassReverse /fedmon HTTP://OIF-HOST:OIF-PORT/fedmon
#ProxyPassReverse /uixi http://OIF-HOST:OIF-NON-SSL-PORT/uixi

#ProxyPass /osso_login_success HTTP://OIF-HOST:OIF-PORT/
   osso_login_success
#ProxyPass /osso_logout_success HTTP://OIF-HOST:OIF-PORT/
   osso_logout_success
#ProxyPassReverse /osso_login_success HTTP://OIF-HOST:OIF-PORT/
   osso_login_success
#ProxyPassReverse /osso_logout_success HTTP://OIF-HOST:OIF-PORT/
   osso_logout_success
```

where:

- *HTTP* is `http` for an open connection or `https` for a secure connection from the proxy to Oracle Identity Federation
- *OIF-HOST* is the hostname of the Oracle Identity Federation server
- *OIF-PORT* is the HTTP or HTTPS port number of the Oracle Identity Federation server. Omit the entry for HTTP port 80 or HTTPS port 443.
- *OIF-NON-SSL-PORT* is the http port number of the Oracle Identity Federation server. Omit the entry for HTTP port 80.

Note:

- Uncomment the `/fedadmin` or `/fedmon` directives only if you need to expose the Oracle Identity Federation administration or monitoring consoles to users on the internet.
- Uncomment the `/uixi` directives if any of the `/fedadmin` or `/fedmon` directives is enabled.
- Uncomment the `/osso_login_success` and `/osso_logout_success` directives if Oracle Identity Federation is integrated with Oracle Single Sign-On Server.

Caution: Oracle recommends that you do not make the administration and monitoring consoles available through the proxy.

---

d. Comment out the `mod_oc4j` include:

```
#include "ORACLE_HOME/Apache/Apache/conf/mod_oc4j.conf
```

e. If using SSL from the proxy to the Oracle Identity Federation server, add an additional directive after `ProxyPassReverse` directives:

```
SSLProxyWallet file:OHS_
HOME/Apache/Apache/conf/ssl.wlt/default
```

where `OHS_HOME` is the install directory for the proxy Oracle HTTP Server. If you have not already done so, also import the certificate of the CA that issued the Oracle Identity Federation server certificate in this wallet. See "Configuring SSL Server on Oracle Identity Federation" on page 6-135 for details.

3. If using SSL with the proxy, follow the instructions in "Configuring SSL Server on Oracle Identity Federation" on page 6-133. Omit the section about editing the `mod_oc4j.conf` file.

   If you require SSL client certificate authentication for the IdP/source side of the SAML 1.x Artifact profile, do the following for the proxy Oracle HTTP Server:

   a. Copy the Perl script shown here to the appropriate location.

      Unix: `ORACLE_HOME/perl/lib/site_perl/5.6.1/Apache/SSLClientCertHeader.pm`

      Windows: `ORACLE_HOME\Apache\Apache\mod_perl\site\5.6.1\lib\Apache\SSLClientCertHeader.pm`

      This script sets the `ssl_client_certificate` header that Oracle Identity Federation uses to complete the client certificate authentication.

```
package Apache::SSLClientCertHeader;

use Apache::Constants qw(OK DECLINED);
use Apache::URI ();
use strict;

sub handler {
    my $r = shift;
    $r->subprocess_env;
    my $clientcert = $r->subprocess_env("SSL_CLIENT_CERT");
    my $outstr;
    # Remove newlines from certificate before setting header.
    for my $i (split "\n", $clientcert) {
        if (!($i =~ /^-/)) {
            chomp($i);
            $outstr .= $i;
        }
    }
    $r->header_in("ssl_client_cert", $outstr);
    return OK;
}

1;
```

   b. Add these directives to the `VirtualHost` for the SSL client certificate authentication port in `ORACLE_HOME/Apache/Apache/conf/ssl.conf`:

```
<VirtualHost _default_:CLIENT-CERT-PORT>
    . . .
    SSLOptions +ExportCertData +CompatEnvVars
    PerlModule Apache::SSLClientCertHeader
    PerlFixupHandler Apache::SSLClientCertHeader
</VirtualHost>
```

Copy an existing virtual host first, similar to the steps for just configuring the Oracle Identity Federation server with client certificate mode. The new virtual host should have the client certificate port number of the proxy. Also, update the responder port on the Domain page of the Oracle Identity Federation administration console with the client certificate port of the proxy server.

4. Restart Oracle HTTP Server to make the configuration changes effective. You can test the proxy by invoking:

   ```
   HTTP://PROXY-HOST:PROXY_
   PORT/shareid/saml/ObSAMLTransferService
   ```

   You should get an Oracle Identity Federation error with Error ID TSE002.

5. Determine the proxy HTTP or HTTPS ports through the Ports page of the Enterprise Manager Console:

   - Oracle HTTP Server Listen port (http)
   - Oracle HTTP Server Listen (SSL) port (https)

6. Reconfigure Oracle Identity Federation to use the proxy host and port for its external URLs. Make these changes in the Oracle Identity Federation administration console:

   a. **Server Configuration** - > **General**:

      - Server Hostname

      - Server Port

      - SOAP Port

   b. **SAML 1.x/WS-Fed** - > **Domains** - > **MyDomain**:

      - Error URL

      - Transfer URL

      - Responder URL

      - SourceID (blank to regenerate for new ResponderURL)

      - Identity Realm STS URL

      - Receiver URL

      - Resource Realm STS URL

      Note: Do not modify the Signing Certificate Subject DN or Issuer DN, as these must continue to match the generated certificate.

7. Restart Oracle Identity Federation to make the General server configuration changes effective.

8. If using Oracle Access Manager as the IdM, use the Access System Console to update the Fed SSO authentication schemes:

   Access System Console - > **Access System Configuration** - > **Authentication Management**:

   - Fed SSO - SAML 1.x

   - Fed SSO - WS-Federation

   - Fed SSO - SAML 2.0/Liberty 1.x

Change the **Challenge Redirect** parameter for each scheme to use the proxy host and port.

9. Communicate the changes to partners using this Oracle Identity Federation, if necessary. Partners using SAML 2.0 or Liberty 1.x will need to download new metadata. Partners using SAML 1.x or WS-Federation will need to manually update their configurations with the Oracle Identity Federation MyDomain configuration set in Step 6.

10. If Oracle Identity Federation is integrated with Oracle Single Sign-On, some additional steps are required.

    a. Update the Oracle Single Sign-On Partner application:

    - Go to the Oracle Single Sign-On administration console (`http://osso_host:osso_port/pls/orasso`).

    - Click **Single Sign-On Administration**.

    - Click **Administer Partner Applications**.

    - Choose the partner application referencing the Oracle Identity Federation server, and edit the application configuration.

    - Replace the http(s), hostname and port number by the proxy's values for Home URL, Success URL, and Logout URL.

    - Click **OK**.

    b. Update the `policy.properties` file:

    - Open the `ORACLE_HOME/sso/conf/policy.properties` file in the Oracle Single Sign-On deployment.

    - Replace the http(s), hostname and port number by the proxy's value for the `SASSOAuthnUrl` and `SASSOLogoutUrl` entries.

    - Save and close the file.

    c. Restart the `OC4J_SECURITY` instance of the Oracle Single Sign-On deployment by using the command:

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

A

Troubleshooting Oracle Identity Federation

This appendix describes common problems that you may encounter when configuring and using Oracle Identity Federation, and explains how to solve them.

Problems and Solutions

This section describes common problems and solutions arranged in these topical groups:

- General Issues
- Oracle Identity Federation Configuration Issues
- Oracle Single Sign-On Login Issues
- Oracle Access Manager Configuration Issues
- Operating System Configuration Issues
- Runtime/Single Sign-On Issues
- Oracle Identity Federation Administration Console Issues

General Issues

This section describes general issues and workarounds. It includes the following topics:

- Reauthentication after Session Timeout with OracleAS Single Sign-On and SAML 1.x or WS-Federation
- Attribute Sharing with the Microsoft Internet Information Server
- Redirection Loops with Oracle Access Manager
- Truncated Text in Japanese Version of Oracle Universal Installer
- Unused Assertion Profile With Invalid Attribute Mapping Can Cause SSO Failure
- Signed SAML 1.0 Assertions Can Cause SSO Failures
- Encrypting Network Connections

Reauthentication after Session Timeout with OracleAS Single Sign-On and SAML 1.x or WS-Federation

Problem

This issue concerns a scenario where Oracle Identity Federation is used as a service provider, OracleAS Single Sign-On is the user data store and an OracleAS Single

Sign-On session is created for a federated user using SAML 1.x or WS-Federation. When that session expires, the service provider's Oracle Identity Federation server tries to reauthenticate the session using SAML 2.0. If SAML 2.0 is not enabled on the service and identity providers, the reauthentication will fail, typically with a 500 Internal Server Error.

Solution

This problem can be avoided by configuring OracleAS Single Sign-On. Open the `ORACLE_HOME/sso/conf/policy.properties` file and protect all the partner applications with the default SSO server authentication plugin; configure the SASSO authentication plugin to have a higher security level than the OracleAS Single Sign-On server plugin.

With this configuration, when a user authenticated by SAML 1.x or WS-Federation protocol accesses a resource protected by OracleAS Single Sign-On, and the session times out, the user will be redirected to theOracleAS Single Sign-On server for local authentication instead of seeing an error from Oracle Identity Federation or an incorrect IdP.

Attribute Sharing with the Microsoft Internet Information Server

Problem

The attribute sharing feature cannot be used with Microsoft Internet Information Servers (IIS) with Oracle Access Manager WebGate agents installed. For this feature an authentication plugin sets an HTTP header with the SubjectDN from the client's X.509 certificate, and an authorization plugin retrieves the header to initiate a SAML attribute query. However, because of the way the IIS WebGate performs SSL client certificate authentication, the SubjectDN header cannot be retrieved by the authorization plugin. In this case the following error is reported at the user's browser:

```
Oracle Access Manager Operation Error Access to the URL
<targetURL> has been denied for user <OblixAnonymous user DN>.
```

Also, the following error messages are written to the `OBACCESS_INSTALL/access/oblix/config/logs/authz_attribute_plugin_log.txt` file:

```
SubjectDN header ObNullString
```

and

```
SubjectDN is missing. Assume local user and return Continue
```

Solution

There is no workaround for this problem, other than to use a different web server such as the Oracle HTTP Server or the Sun One Web Server.

Redirection Loops with Oracle Access Manager

Problem

When Oracle Identity Federation is used as an identity provider with the Oracle Access Manager user data store, a user initiating additional SAML 1.x or WS-Federation single sign-ons might experience a redirection loop at the browser.

This occurs if the Oracle Access Manager AccessGate configured for Oracle Identity Federation has an Idle Session Timeout less than the Maximum user session time. In this case, if the user waits for the idle session timeout to elapse and then initiates another SSO, the redirection loop will occur.

Solution

This can be avoided by setting the Oracle Identity Federation AccessGate's Idle Session Timeout equal to or greater than the Maximum user session timeout (which is the default setting).

Truncated Text in Japanese Version of Oracle Universal Installer

The following issue is observed during a Japanese-language installation session:

1. Start Oracle Universal Installer.
2. Choose the "Oracle Identity Federation 10g" installation option.
3. Proceed to the "Select Installation Method" page.

The text describing the first radio button ("Basic"), is truncated.

Unused Assertion Profile With Invalid Attribute Mapping Can Cause SSO Failure

Problem

If Oracle Identity Federation is used as an identity provider with an RDBMS user data store, a configured SAML 1.x assertion profile with a non-existent user attribute will cause all single sign-ons (SSOs) using the SAML 1.x and WS-Federation profiles to fail, even if they do not use the invalid profile.

When a user logs into an Oracle Identity Federation identity provider with the RDBMS user data store, Oracle Identity Federation attempts to retrieve all user attributes in all configured assertion profiles. If any of the attributes are invalid, the SSO will fail.

The user will receive a `500 Internal Server Error`. If debug logging is enabled, the `federation.log` file will show the following error:

```
RDBMSBridge.authenticate(): ERROR - SQL Exception thrown by
JDBC: java.sql.SQLException: ORA-00904: "<attribute name>":
invalid identifier
```

Solution

The workaround is to correct the invalid user attribute in the offending assertion profile, or delete the offending assertion profile.

Signed SAML 1.0 Assertions Can Cause SSO Failures

Problem

Because SAML 1.0 does not fully specify how the XML Signature standard is to be used, Oracle Identity Federation cannot - within the context of a SAML response - correctly generate a signed SAML 1.0 assertion, nor verify a received signed SAML 1.0 assertion. Consequently, signatures on SAML 1.0 assertions used for the Artifact and POST SSO profiles are incorrect. If a user attempts to perform a single sign-on (SSO) using a SAML 1.x assertion profile with assertion singing enabled, and SAML 1.1 is not enabled for MyDomain or the destination domain, the service provider/destination site may not be able to verify the signature on the SSO assertion, causing the SSO to fail. If the destination site uses Oracle Identity Federation, the `federation.log` file will show:

```
RECEIVER: ERROR: An invalid SAML Response was received: XML
SIGNER: ERROR: Invalid signature or altered contents
```

Solution

The workaround is to use the SAML 1.1 protocol instead of SAML 1.0. (In fact, one of the reasons for the SAML 1.1 revision was to allow better use of XML Signatures.)

> **Note:** Signed assertions are not required, nor are they commonly used, for the SAML 1.x SSO profiles.

Encrypting Network Connections

By default, JDBC does not encrypt network connections between Oracle Identity Federationand the Oracle9*i* Database Server. Sites can optionally use Oracle Advanced Security to encrypt these connections.

In configuring Oracle Identity Federation to use Oracle Internet Directory or other LDAP servers to authenticate users, a site may choose whether to use SSL to connect to the LDAP server. If you do not use SSL, unencrypted passwords may be sent over network connections between Oracle Identity Federation and the LDAP server.

Oracle Identity Federation Configuration Issues

This section describes server configuration issues:

Administration Console Is Not Accessible After Changing Transient Data Store

Problem

You may be unable to access the Oracle Identity Federation administration console in this situation:

1. The command-line configuration assistant is executed to change the RDBMS database used for the transient data store. The command format is as follows:

   ```
   java -jar install.jar -transient rdbms <parameters>
   ```

2. After the command is executed, the Oracle Identity Federation administration console is not accessible, and the federation logs or the OPMN logs show errors like the following:

   ```
   Invalid username/password
   ```

This issue is seen when switching the Oracle Identity Federation transient store from one database to another, using a different username/password combination, or when using the same database but with different credentials.

This problem arises because Oracle Identity Federation is already set up for RDBMS transient data store, but when the command-line configuration assistant is executed, the database password does not get reset; this results in the invalid username/password error when trying to perform any Oracle Identity Federation operations.

Solution

Use these steps to work around the problem:

1. Log on to the Oracle Enterprise Manager 10*g* Grid Control Console.
2. Navigate to **OC4J_FED** - > **Administration** - > **Security**.
3. In the Users list, click the jazn.com/oif_db entry.
4. Enter the correct password to access the RDBMS.
5. Apply, and restart the OC4J_FED instance.

Signing SAML Response with Assertion

When an Oracle Identity Federation IdP is configured to send signed both Response messages and Assertions, only the Assertions are signed.

This affects SSO and attribute sharing profiles for the Liberty 1.x and SAML 2.0 protocols. This does not affect profiles where a Response message does not contain an Assertion.

Assertions Using SAML 1.x POST Method Fail in Japanese Locale

Problem

In the Japanese locale, assertions using the SAML 1.x POST method fail with this error:

```
ERROR: The SAML Response was not signed by the expected
authority (RVE013)
```

The problem is due to the translated strings for OU and ST in the Signing Certificate Subject DN and the Signing Certificate Issuer DN.

Solution

As a workaround to this problem, the OU and ST values need to be replaced with the equivalent English strings. You can obtain the English value of the strings from the Issuer and Subject DN in the MyDomain configuration.

Requester ID in SAML 1.x Artifacts

Problem

When using the SAML 1.x Artifact mode, and the requester ID on the Domain page contains a colon ":", single sign-on using the artifact profile fails with this error at the service provider:

```
ERROR - RESPONDER: ERROR Unknown requester <domain name>
```

This problem is due to the use of the colon character, which is invalid in this context.

Solution

When specifying the requester ID for each domain, make sure that the string does not contain a ":" character.

Logout Displays No Return Page

Problem

When invoking the Oracle Identity Federation logout URL, the sign-off operation is performed, but the server does not display a result page; instead, it displays the last page visited by the browser, and the operation appears to have aborted even though logout was successful.

Solution

The logout service takes a `returnurl` parameter, which is required for correct operation. This problem occurs if no `returnurl` parameter is specified when invoking the Oracle Identity Federation logout URL. To avoid the problem, specify the `returnurl` parameter to point to the result page.

No JSESSIONID cookie Error

Problem

This problem is seen in a load-balancing configuration:

1. The IdP has two load-balanced Oracle Identity Federation instances.
2. A SAML 1.x SSO request of this type is issued:

   ```
   http://stitiasfw2.us.oracle.com:80/shareid/saml/ObSAMLTransfe
   rService?TARGET=http://target.us.oracle.com:80/test.html&DOMA
   IN=some-host.us.oracle.com&METHOD=POST
   ```

3. You log in with valid user credentials.
4. The following error results:

   ```
   ERROR - LOCAL LOGIN: ERROR: No JSESSIONID cookie in a POST
   request.
   ```

Solution

Ensure that the URLs in MyDomain at the load-balanced provider, and in the load-balanced domain at the other provider, have the hostname and address of the load-balancer, and not the hostname and address of one of the load-balanced Oracle Identity Federation instances.

Failed to find orclfednamevalue Error

Problem

A schema violation error occurs when performing a Liberty 1.x / SAML 2.0 single sign-on operation, with the federation data store residing in an LDAP server. The `$Oracle_Home/fed/log/federation.log` shows the following error message:

```
javax.naming.directory.SchemaViolationException:
[LDAP: error code 65 -
Failed to find orclfednamevalue in mandatory or optional
attribute list.]
```

This problem is seen if the schema of the federation data store's LDAP server has not been upgraded to include the Oracle Identity Federation attributes and object classes.

Solution

Upgrade the LDAP schema either at installation time (with the Advanced Installation mode), or after installation.

Upgrade Schema at Installation

To perform the upgrade at installation time, take these steps:

1. Choose the Advanced Installation mode.
2. On the "Select Configuration Options" page, check the "Federation Data in LDAP Server" box. This indicates that the federation records will be stored in an LDAP server whose schema must be upgraded.

3. On the "Specify Federation Data Store" page, enter the LDAP connection information. The schema will then be upgraded as part of the installation process.

Post-Installation Schema Upgrade

To perform the upgrade post-installation, note that the Oracle Identity Federation installation includes `LDIF` files that you can execute using the `ldapmodify` tool to upgrade the schema of an LDAP server.

The `LDIF` file to use depends on the type of LDAP server used:

- `$Oracle_Home/fed/setup/ldap/userFedSchemaOid.ldif` if you use Oracle Internet Directory
- `$Oracle_Home/fed/setup/ldap/userFedSchemaIPlanet.ldif` if you use the Sun One Directory Server
- `$Oracle_Home/fed/setup/ldap/userFedSchemaAD.ldif` if you use Microsoft Active Directory Server. In this case, you need to edit the LDIF file to replace the string `%DOMAIN_DN%` with your active directory domain suffix.

  An example suffix is `dc=mydomain,dc=mycompany,dc=com`.

Using `ldapmodify`, you can upgrade the LDAP schema with the LDIF file. For example:

```
ldapmodify -c -D BIND_DN_USERNAME
   -w PASSWORD
   -f $Oracle_Home/fed/setup/ldap/userFedSchemaOid.ldif
   -h LDAP_HOSTNAME -p LDAP_PORT -x
```

Oracle Single Sign-On Login Issues

This section describes issues that you may encounter when Oracle Identity Federation is the service provider (SP), and Oracle Single Sign-On is configured as the identity provider (IdP) at the back end. It contains these topics:

- Incorrect Login Page Appears
- Bookmarked Login Pages
- Error When Reissuing SAML 1.x URL After Timeout

Incorrect Login Page Appears

Problem

The following setup produces an incorrect login page:

1. Oracle Identity Federation is configured as a service provider.
2. The partner application is configured to be protected by Oracle Identity Federation.
3. When a user tries to access the protected resource, the Oracle Single Sign-On login page appears instead of the intended Oracle Identity Federation login page.

This problem can occur if the partner application is incorrectly configured for `mod_osso`, causing the user to be prompted for local authentication.

Solution

The steps required to ensure that the partner application is correctly configured for `mod_osso` are outlined here. Detailed information appears in the *Oracle Application Server Single Sign-On Administrator's Guide.*

1. Shut down Oracle HTTP Server and OC4J_SECURITY.
2. Edit the Oracle HTTP Server configuration file, `ORACLE_HOME/Apache/Apache/conf/httpd.conf`, located in the Oracle Application Server Infrastructure directory:
   - Add the osso_module to the server's loaded modules using the `AddModule` (Windows) and `LoadModule` (Windows, Linux) directives. See the *Oracle Application Server Single Sign-On Administrator's Guide* for an example.
   - Add a virtual host to create a new partner application listener that will be protected by `mod_osso`. See the *Oracle Application Server Single Sign-On Administrator's Guide* for details about configuring `mod_osso` with virtual hosts.
3. Run `ssoreg` to manually configure `mod_osso` and the Oracle Single Sign-On server.

   Note:

   - Check that the `osso_APPLICATION_NAME.conf` matches the value defined in the virtual host configuration.
   - See the *Oracle Application Server Single Sign-On Administrator's Guide* for details of `ssoreg` syntax and parameters, including instructions on how to provide directives for each protected host in the `httpd.conf` file.
4. Restart Oracle HTTP Server (OHS) And OC4J_SECURITY.

Additional information about integrating Oracle Identity Federation with Oracle Single Sign-On appears in the *Oracle Application Server Single Sign-On Administrator's Guide*.

Bookmarked Login Pages

Problem

Attempting to log in by means of a bookmarked login page returns an error. This is seen when a user follows this sequence:

- Perform a single sign-on (SSO) operation using SAML 2.0, Liberty, or WS-Federation protocols.
- On the login page, bookmark the page.
- Open a new browser instance and go to the bookmarked login page. Log in with valid user credentials.

The user will receive an error and SSO will fail.

Solution

Do not bookmark the login page. Oracle Identity Federation does not support the use of bookmarked login pages.

Error When Reissuing SAML 1.x URL After Timeout

Problem

This problem occurs when the service provider back-end is using OracleAS Single Sign-On. It occurs in this situation:

1. A SAML 1.x transfer URL is issued.
2. After waiting for the duration of the session timeout at the SP (set in OracleAS Single Sign-On and the Oracle Identity Federation administration console), reissue the same transfer URL in the same browser instance and log in again.
3. An internal server error is seen, or unusual redirection behavior may be observed.

When a resource is protected by OracleAS Single Sign-On and configured to use Oracle Identity Federation authentication, the only protocols that the OracleAS Single Sign-On server can initiate for authentication are Liberty 1.x and SAML 2.0. Consider a user who is authenticated using SAML 1.x or WS-Federation protocols, and who obtains access to such a resource. When its session times out, OracleAS Single Sign-On will redirect the user to Oracle Identity Federation for authentication using Liberty 1.x or SAML 2.0, which may result in server errors if the protocols are not configured, or unexpected redirection as the user is redirected to an IdP different from the SAML 1.x or WS-Federation login.

Solution

The immediate workaround is to close the browser instance and reissue the SAML 1.x SSO request in a new browser instance.

For a permanent solution, configure OracleAS Single Sign-On so that, when a user is authenticated by SAML 1.x/WS-Fed protocols, and accesses a resource protected by OracleAS Single Sign-On, when the session times out, the user is redirected back to OracleAS Single Sign-On for local authentication:

1. Open the `Oracle_Home/sso/conf/policy.properties` file.
2. Protect all the partner applications with the default OracleAS Single Sign-On server authentication plugin.
3. Configure the SASSO authentication plugin to have a higher security level than the OracleAS Single Sign-On server plugin.

Oracle Access Manager Configuration Issues

This section describes issues that you may encounter when configuring Oracle Access Manager components at the back end. It contains these topics:

AccessGate Permission Error

Problem

The following setup produces an AccessGate configuration error:

1. The Access Server SDK specifies a certain user under whom the AccessGate runs.
2. Oracle Identity Federation is installed on the Linux or Solaris platform under a different user.
3. When you apply the AccessGate configuration page for the Oracle Access Manager user data store, you receive the error:

```
AccessGate configuration failed. Reason: Preparing to connect
to Access
Server. Please wait. Error: Permission denied.
```

This error results from having different owners for the Oracle Identity Federation and Access Server SDK installations.

Solution

When Oracle Identity Federation is installed on Linux or Solaris, ensure that the AccessServerSDK files, installed at `ORACLE_HOME/fed/shareid`, have the same owner and group as the Oracle Identity Federation installation.

Non-ASCII AccessGate ID

Problem

If you attempt to configure an Oracle Access Manager User Data Store with an AccessGate, and the AccessGate ID contains non-Latin characters (for example, "ÆÖ2"), you get the error:

```
AccessGate configuration failed. Reason: Preparing to connect to
Access Server. Please wait. Client authentication failed, please
verify your AccessGate ID.
```

The problem also occurs when the AccessGate ID contains non-ASCII Latin-1 characters (for example, "Ådmïn").

Solution

Use only ASCII characters in the AccessGate ID for the Oracle Identity Federation AccessGate.

Setting LD_ASSUME_KERNEL Value

Problem

The Oracle Identity Federation instance (`OC4J_FED`) crashes when operating with an Oracle Access Manager back-end.

This problem may be due to an incorrect setting for the `LD_ASSUME_KERNEL` environment variable. This variable must be set to `2.4.19`, because the Access Server SDK supports the Linux threading model and not the native posix thread library (NPTL).

For example, if `LD_ASSUME_KERNEL` is not set, you may see this type of error in the Oracle Identity Federation `federation.log` and `federation-error.log` files:

```
06/06/15 10:05:22: ERROR ShareIDLogger
- ObRareqService: EXCEPTION during initialization: (SVX001)
com.oblix.access.ObAccessException:
Env variable LD_ASSUME_KERNEL not set to 2.4.19.
      at com.oblix.access.ObConfig.jni_initialize(Native Method)
      ...
```

An "internal server error" message may also be displayed at the user's browser.

Solution

Refer to "Integrate Oracle Identity Federation and Oracle Access Manager" on page 4-9. Step 3 describes how to set `LD_ASSUME_KERNEL` using the AccessServerSDK installer.

Using the Same Cookie Domain for Two Back-ends

Problem

If your configuration involves two providers with Oracle Access Manager back-ends (IdP and SP2, for example), and both instances are using the same cookie domain, you may see an error when attempting single sign-on. The error message in the log file looks like this:

```
06/05/21 01:24:40: ERROR - COREID BRIDGE: ERROR The session
token loggedoutcontinue is invalid:
com.oblix.access.ObAccessException: Session token passed to the
ObUserSession constructor is null or invalid. (NBE006)
```

This problem occurs when multiple Oracle Access Manager providers are using the same cookie domain.

Solution

You can resolve the issue by changing the Oracle Access Manager instances to use different cookie domains, or by using a different back-end (such as an LDAP back-end) at the IdP.

Operating System Configuration Issues

This section describes issues related to the configuration of the operating system of the machine where Oracle Identity Federation is installed:

- File Descriptors on Linux
- Search Fails Against Microsoft Active Directory with an Unknown Host Exception

> **Note:** Some issues listed in this section may have a system-wide impact on the Oracle Identity Federation server, while others may only impact a specific component such as a particular federation partner.

File Descriptors on Linux

Problem

You may experience intermittent Oracle Identity Federation server crashes with this error message in the `federation-error.log` file:

```
java.net.SocketException: Too many open files
```

This error occurs when the file descriptor limit is reached.

Solution

Increase the file descriptor limit, which is specified in the `/etc/security/limits.conf` configuration file.

> **Note:** If no file descriptor limit is defined, the server uses a default value of 1024.

In this example, the file descriptor limit is being set to a value of 16K:

```
soft nofile 16384
hard nofile 16384
```

Reboot the machine after changing the value.

Search Fails Against Microsoft Active Directory with an Unknown Host Exception

When searching for information in an Active Directory environment that is configured for LDAP referrals, the referrals fail if the host being referred to is in a different domain than the Active Directory server.

Problem

When a user requests a resource, at times verification of the user's identity can fail due to an inability to validate the user's identity in the directory. This error can occur in an Active Directory environment when the user's browser runs on a non-Windows computer, or if the user's browser runs on a Windows computer that is not in the Active Directory server domain.

This problem can arise due to LDAP referral chasing. An LDAP referral occurs when a domain controller does not have the section of the directory tree where a requested object resides. The domain controller refers the client to another destination so that the client can conduct a DNS search for another domain controller. If the client is configured to chase referrals, the search can continue.

For the scenario where the user has a Windows-based computer, an issue can occur with LDAP referrals if the client's domain controller does not have a trust relationship with the Active Directory domain controller.

Solution

If you encounter this issue, add the entry for the Active Directory host's address in the following list:

WINDOWS_HOME_DIRECTORY`\system32\drivers\etc\hosts`

On Windows XP, the list is located here:

`C:\WINDOWS\system32\drivers\etc\host`

On a Unix-based system, add this entry to the `/etc/hosts` file, using the format:

IP_address_of_AD_host AD_host_name

where *AD_host_name* is the host name specified in the referral, for example:

`123.123.123.123 my2003ad.com`

Runtime/Single Sign-On Issues

This section describes various runtime and single sign-on issues that you may encounter when using Oracle Identity Federation:

- 404 Error when Using Oracle SmartMarks
- Incorrect Identity Provider for SAML 1.x or WS-Federation
- Bookmarking a WS-Federation Protected Resource

404 Error when Using Oracle SmartMarks

Problem

The following configuration produces a 404 error when using Oracle SmartMarks:

1. Set up Oracle SmartMarks at source and destination sites; that is, enable Oracle SmartMarks in domain entries and enter the transfer query string in the source domain at the destination site.

2. Issue a SAML 1.x transfer URL. For example:

   ```
   http://ref1.refcompany.com:80/shareid/saml/ObSAMLTransferServ
   ice?DOMAIN=titanium.refcompany.com&TARGET=https://sometarget.
   refcompany.com:2008/test.html&METHOD=POST
   ```

3. Close the browser and open a new browser instance.
4. Attempt to access the protected resource directly:

   ```
   https://sometarget.refcompany.com:2008/test.html
   ```

5. Enter valid credentials (username and password) when redirected to the source site for authentication.

After logging in, you get a `404` error instead of access to the protected resource.

This problem may occur due to the use of an incorrect transfer query string.

Solution

Check that the transfer query string does not include the actual target after the `TARGET=` keyword.

For example, this is an incorrect transfer query string:

```
DOMAIN=platinumob.refcompany.com&METHOD=POST&TARGET=https://some
target.refcompany.com:2008/test.html
```

And this is the correct transfer query string:

```
DOMAIN=titanium.refcompany.com&METHOD=POST&TARGET=
```

Incorrect Identity Provider for SAML 1.x or WS-Federation

Problem

A user changes identity providers (for example, quits Company A and joins Company B) but is redirected to the old identity provider for an SP-initiated SSO with SAML 1.x or WS-Federation.

Solution

The user should clear any ObSAMLDomain cookies set in the browser.

Bookmarking a WS-Federation Protected Resource

Problem

If a user bookmarks a WS-Federation protected resource, and the service provider is using an OracleAS Single Sign-On back-end, the user will receive an error when later trying to access the bookmark.

Solution

Accessing WS-Federation protected resources directly is not supported if the SP is using an OracleAS Single Sign-On back-end. Users should not bookmark and later attempt to access the resource in this scenario.

Oracle Identity Federation Administration Console Issues

This section contains issues related to the Oracle Identity Federation administration console.

Cannot Log in to the Administration Console

Problem

This problem is seen when Oracle Identity Federation server is integrated with Oracle Single Sign-On.

When performing a federated single sign-on using Oracle Identity Federation, where the user authenticates using OracleAS Single Sign-On, it is not possible to access the Oracle Identity Federation administration or monitoring console; access is denied, and authentication fails.

The Oracle Identity Federation consoles are protected by the JAZN module. This problem can be caused by a role conflict between OracleAS Single Sign-On and the JAZN module.

Solution

There are two possible solutions: for an immediate resolution, access the Administration or Monitoring console in a clean browser instance. For a durable solution, configure the administration and monitoring consoles to be protected by OracleAS Single Sign-On.

Take these steps to protect the consoles using OracleAS Single Sign-On:

1. Log on to the Oracle Enterprise Manager 10g Application Server Control Console.
2. At the console, navigate to **OC4J_FED** - > **Applications** - > **fed** or **fedmon** - > **General**.
3. Select "Use JAZN LDAP User Manager" which contains the Oracle Internet Directory location.
4. Click **Apply**.
5. Go to **OC4J_FED** - > **Applications** - > **fed** or **fedmon** - > **Security**.
6. Click **Map Role To Principals**.
7. Select user(s) or group(s) from the LDAP server that will have access to the console.
8. Click **Apply**.
9. Restart the OC4J_FED instance.

A user attempting to access the console will now be redirected to the OracleAS Single Sign-On server for authentication.

B

References

Table B–1 lists the standards documents and protocols referenced in this document.

Table B–1 Identity Federation References

Document	Reference
[SAMLBind]	S. Cantor et al. Bindings for the OASIS Security Assertion Markup Language (SAML) V2.0. OASIS SSTC, March 2005. Document ID saml-bindings-2.0-os. See http://docs.oasis-open.org/security/saml/v2.0/saml-bindings-2.0-os.pdf.
[SAMLCore]	S. Cantor et al. Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0. OASIS SSTC, March 2005. Document ID saml-core-2.0-os, http://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf.
[SAMLErrata]	J. Moreh. Errata for the OASIS Security Assertion Markup Language (SAML) V2.0. OASIS SSTC, May, 2006. Document ID sstc-saml-errata-2.0-draft-nn, http://www.oasis-open.org/committees/security/.
[SAMLExecOvr]	P. Madsen, et al. SAML V2.0 Executive Overview. OASIS SSTC, April, 2005. Document ID sstc-saml-exec-overview-2.0-cd-01, http://www.oasis-open.org/committees/security/.
[SAMLGloss]	J. Hodges et al. Glossary for the OASIS Security Assertion Markup Language (SAML) V2.0. OASIS SSTC, March 2005. Document ID saml-glossary-2.0-os, http://docs.oasis-open.org/security/saml/v2.0/saml-glossary-2.0-os.pdf.
[SAMLMDExtQ]	T. Scavo, et al. SAML Metadata Extension for Query Requesters. OASIS SSTC, March 2006. Document ID sstc-saml-metadata-ext-query-cd-01, http://www.oasis-open.org/committees/security/.
[SAMLMeta]	S. Cantor et al. Metadata for the OASIS Security Assertion Markup Language (SAML) V2.0. OASIS SSTC, March 2005. Document ID saml-metadata-2.0-os, http://docs.oasis-open.org/security/saml/v2.0/saml-metadata-2.0-os.pdf.
[SAMLProf]	S. Cantor et al. Profiles for the OASIS Security Assertion Markup Language (SAML) V2.0. OASIS SSTC, March 2005. Document ID saml-profiles-2.0-os, http://docs.oasis-open.org/security/saml/v2.0/saml-profiles-2.0-os.pdf.
[SAMLSec]	F. Hirsch et al. Security and Privacy Considerations for the OASIS Security Assertion Markup Language (SAML) V2.0. OASIS SSTC, March 2005. Document ID saml-secconsider-2.0-os, http://docs.oasis-open.org/security/saml/v2.0/saml-secconsider-2.0-os.pdf.
[SAMLX509Attr]	R. Randall et al. SAML Attribute Sharing Profile for X.509 Authentication-Based Systems. OASIS SSTC, March 2006. Document ID sstc-saml-x509-authn-attribprofile-cd-02, http://www.oasis-open.org/committees/security/.

Glossary

3DES

See Triple Data Encryption Standard (3DES).

account lockout

A security feature that locks a user account if repeated failed logon attempts occur within a specified amount of time, based on security policy settings. Account lockout occurs in OracleAS Single Sign-On when a user submits an account and password combination from any number of workstations more times than is permitted by Oracle Internet Directory. The default lockout period is 24 hours.

Advanced Encryption Standard (AES)

Advanced Encryption Standard (AES) is a symmetric cryptography algorithm that is intended to replace Data Encryption Standard (DES). AES is a Federal Information Processing Standard (FIPS) for the encryption of commercial and government data.

advanced symmetric replication (ASR)

See Oracle Database Advanced Replication.

AES

See Advanced Encryption Standard (AES).

anonymous authentication

The process by which a directory authenticates a user without requiring a user name and password combination. Each anonymous user then exercises the privileges specified for anonymous users.

API

See application programming interface (API).

application programming interface (API)

A series of software routines and development tools that comprise an interface between a computer application and lower-level services and functions (such as the operating system, device drivers, and other software applications). APIs serve as building blocks for programmers putting together software applications. For example, LDAP-enabled clients access Oracle Internet Directory information through programmatic calls available in the LDAP API.

application service provider

Application Service Providers (ASPs) are third-party entities that manage and distribute software-based services and solutions to customers across a wide area

network from a central data center. In essence, ASPs are a way for companies to outsource some or almost all aspects of their information technology needs.

artifact profile

An authentication mechanism which transmits data using a compact reference to an assertion, called an artifact, instead of sending the full assertion. This profile accomodates browsers which handle a limited number of characters.

ASN.1

Abstract Syntax Notation One (ASN.1) is an International Telecommunication Union (ITU) notation used to define the syntax of information data. ASN.1 is used to describe structured information, typically information that is to be conveyed across some communications medium. It is widely used in the specification of Internet protocols.

assertion

An assertion is a statement used by providers in security domains to exchange information about a subject seeking access to a resource. Identity providers and service providers exchange assertions about identities to make authentication and authorization decisions, and to determine and enforce security policies protecting the resource.

asymmetric algorithm

A cryptographic algorithm that uses different keys for encryption and decryption.

See also: public key cryptography.

asymmetric cryptography

See public key cryptography.

authentication

The process of verifying the identity claimed by an entity based on its credentials. Authentication of a user is generally based on something the user knows or has (for example, a password or a certificate).

Authentication of an electronic message involves the use of some kind of system (such as public key cryptography) to ensure that a file or message which claims to originate from a given individual or company actually does, and a check based on the contents of a message to ensure that it was not modified in transit.

authentication level

An OracleAS Single Sign-On parameter that enables you to specify a particular authentication behavior for an application. You can link this parameter with a specific authentication plugin.

authentication plugin

An implementation of a specific authentication method. OracleAS Single Sign-On has Java plugins for password authentication, digital certificates, Windows native authentication, and third-party access management.

authorization

The process of granting or denying access to a service or network resource. Most security systems are based on a two step process. The first stage is authentication, in which a user proves his or her identity. The second stage is authorization, in which a user is allowed to access various resources based on his or her identity and the defined authorization policy.

authorization policy

Authorization policy describes how access to a protected resource is governed. Policy maps identities and objects to collections of rights according to some system model. For example, a particular authorization policy might state that users can access a sales report only if they belong to the sales group.

basic authentication

An authentication protocol supported by most browsers in which a Web server authenticates an entity with an encoded user name and password passed via data transmissions. Basic authentication is sometimes called plaintext authentication because the base-64 encoding can be decoded by anyone with a freely available decoding utility. Note that encoding is not the same as encryption.

Basic Encoding Rules (BER)

Basic Encoding Rules (BER) are the standard rules for encoding data units set forth in ASN.1. BER is sometimes incorrectly paired with ASN.1, which applies only to the abstract syntax description language, not the encoding technique.

BER

See Basic Encoding Rules (BER).

binding

In networking, binding is the establishment of a logical connection between communicating entities.

In the case of Oracle Internet Directory, binding refers to the process of authenticating to the directory.

The formal set of rules for carrying a SOAP message within or on top of another protocol (underlying protocol) for the purpose of exchange is also called a binding.

CA

See Certificate Authority (CA).

CA certificate

A Certificate Authority (CA) signs all certificates that it issues with its private key. The corresponding Certificate Authority's public key is itself contained within a certificate, called a CA Certificate (also referred to as a root certificate). A browser must contain the CA Certificate in its list of trusted root certificates in order to trust messages signed by the CA's private key.

cache

Generally refers to an amount of quickly accessible memory in your computer. However, on the Web it more commonly refers to where the browser stores downloaded files and graphics on the user's computer.

CBC

See cipher block chaining (CBC).

certificate

A certificate is a specially formatted data structure that associates a public key with the identity of its owner. A certificate is issued by a Certificate Authority (CA). It contains the name, serial number, expiration dates, and public key of a particular entity. The

certificate is digitally signed by the issuing CA so that a recipient can verify that the certificate is real. Most digital certificates conform to the X.509 standard.

Certificate Authority (CA)

A Certificate Authority (CA) is a trusted third party that issues, renews, and revokes digital certificates. The CA essentially vouches for a entity's identity, and may delegate the verification of an applicant to a Registration Authority (RA). Some well known Certificate Authorities (CAs) include Digital Signature Trust, Thawte, and VeriSign.

certificate chain

An ordered list of certificates containing one or more pairs of a user certificate and its associated CA certificate.

certificate revocation list (CRL)

A Certificate Revocation List (CRL) is a list of digital certificates which have been revoked by the Certificate Authority (CA) that issued them.

change logs

A database that records changes made to a directory server.

cipher

See cryptographic algorithm.

cipher block chaining (CBC)

Cipher block chaining (CBC) is a mode of operation for a block cipher. CBC uses what is known as an initialization vector (IV) of a certain length. One of its key characteristics is that it uses a chaining mechanism that causes the decryption of a block of ciphertext to depend on all the preceding ciphertext blocks. As a result, the entire validity of all preceding blocks is contained in the immediately previous ciphertext block.

cipher suite

In Secure Sockets Layer (SSL), a set of authentication, encryption, and data integrity algorithms used for exchanging messages between network nodes. During an SSL handshake, the two nodes negotiate to see which cipher suite they will use when transmitting messages back and forth.

ciphertext

Ciphertext is the result of applying a cryptographic algorithm to readable data (plaintext) in order to render the data unreadable by all entities except those in possession of the appropriate key.

circle of trust

A trust relationship among a set of identity providers and service providers that allows a Principal to use a single federated identity and single sign-on when conducting business transactions with providers within that set.

Businesses federate or affiliate together into circles of trust based on Liberty-enabled technology and on operational agreements that define trust relationships between the businesses.

See also: federation, Liberty Alliance.

claim

A claim is a declaration made by an entity (for example, a name, identity, key, group, and so on).

client SSL certificates

A type of certificate used to identify a client machine to a server through Secure Sockets Layer (SSL) (client authentication).

cluster

A collection of interconnected usable whole computers that is used as a single computing resource. Hardware clusters provide high availability and scalability.

CMP

See certificate management protocol (CMP).

CMS

See Cryptographic Message Syntax (CMS).

code signing certificates

A type of certificate used to identify the entity who signed a Java program, Java Script, or other signed file.

Cold Failover Cluster

Oracle Application Server Cold Failover Clusters are a high availability solution where the Oracle Application Server Infrastructure is typically deployed on a two-node hardware cluster with a shared storage device. Once node is active or "hot," meaning it is running the Infrastructure, while the other node is "cold" and is not running the Infrastructure. When the active node fails, the clusterware switches Infrastructure operations to the previously "cold" node and the Infrastructure is started on that node.

concurrency

The ability to handle multiple requests simultaneously. Threads and processes are examples of concurrency mechanisms.

confidentiality

In cryptography, confidentiality (also known as privacy) is the ability to prevent unauthorized entities from reading data. This is typically achieved through encryption.

connect descriptor

A specially formatted description of the destination for a network connection. A connect descriptor contains destination service and network route information.

The destination service is indicated by using its service name for the Oracle Database or its Oracle System Identifier (SID) for Oracle release 8.0 or version 7 databases. The network route provides, at a minimum, the location of the listener through use of a network address.

contention

Competition for resources.

context prefix

The distinguished name (DN) of the root of a naming context.

CRL

See certificate revocation list (CRL).

CRMF

See certificate request message format (CRMF).

cryptographic algorithm

A cryptographic algorithm is a defined sequence of processes to convert readable data (plaintext) to unreadable data (ciphertext) and vice versa. These conversions require some secret knowledge, normally contained in a key. Examples of cryptographic algorithms include DES, AES, Blowfish, and RSA.

Cryptographic Message Syntax (CMS)

Cryptographic Message Syntax (CMS) is a syntax defined in RFC 3369 for signing, digesting, authenticating, and encrypting digital messages.

cryptography

The process of protecting information by transforming it into an unreadable format. The information is encrypted using a key, which makes the data unreadable, and is then decrypted later when the information needs to be used again. See also public key cryptography and symmetric cryptography.

dads.conf

A configuration file for Oracle HTTP Server that is used to configure a database access descriptor (DAD).

DAS

See Oracle Delegated Administration Services. (DAS).

Data Encryption Standard (DES)

Data Encryption Standard (DES) is a widely used symmetric cryptography algorithm developed in 1974 by IBM. It applies a 56-bit key to each 64-bit block of data. DES and 3DES are typically used as encryption algorithms by S/MIME.

data integrity

The guarantee that the contents of the message received were not altered from the contents of the original message sent.

See also: integrity.

database access descriptor (DAD)

Database connection information for a particular Oracle Application Server component, such as the OracleAS Single Sign-On schema.

decryption

The process of converting the contents of an encrypted message (ciphertext) back into its original readable format (plaintext).

defederation

The act of unlinking a user's account from an identity provider or service provider.

DER

See Distinguished Encoding Rules (DER).

DES

See Data Encryption Standard (DES).

DIB

See directory information base (DIB).

Diffie-Hellman

Diffie-Hellman (DH) is a public key cryptography protocol that allows two parties to establish a shared secret over an unsecure communications channel. First published in 1976, it was the first workable public key cryptographic system.

See also: symmetric algorithm.

digest

See message digest.

digital certificate

See certificate.

digital signature

A digital signature is the result of a two-step process applied to a given block of data. First, a hash function is applied to the data to obtain a result. Second, that result is encrypted using the signer's private key. Digital signatures can be used to ensure integrity, message authentication, and non-repudiation of data. Examples of digital signature algorithms include DSA, RSA, and ECDSA.

Digital Signature Algorithm (DSA)

The Digital Signature Algorithm (DSA) is an asymmetric algorithm that is used as part of the Digital Signature Standard (DSS). It cannot be used for encryption, only for digital signatures. The algorithm produces a pair of large numbers that enable the authentication of the signatory, and consequently, the integrity of the data attached. DSA is used both in generating and verifying digital signatures.

See also: Elliptic Curve Digital Signature Algorithm (ECDSA).

Distinguished Encoding Rules (DER)

Distinguished Encoding Rules (DER) are a set of rules for encoding ASN.1 objects in byte-sequences. DER is a special case of Basic Encoding Rules (BER).

distinguished name (DN)

A X.500 distinguished name (DN) is a unique name for a node in a directory tree. A DN is used to provide a unique name for a person or any other directory entry. A DN is a concatenation of selected attributes from each node in the tree along the path from the root node to the named entry's node. For example, in LDAP notation, the DN for a person named John Smith working at Oracle's US office would be: "cn=John Smith, ou=People, o=Oracle, c=us".

DN

See distinguished name (DN).

Document Type Definition (DTD)

A Document Type Definition (DTD) is a document that specifies constraints on the tags and tag sequences that are valid for a given XML document. DTDs follow the rules of Simple Generalized Markup Language (SGML), the parent language of XML.

domain

A domain is a web site and applications that enable a principal to utilize resources. In federated identity management (FIM), a federated site acts as an identity provider (also known as the source domain), a service provider (or destination domain), or both.

DSA

See Digital Signature Algorithm (DSA) or directory system agent (DSA).

DSE

See directory-specific entry (DSE).

DTD

See Document Type Definition (DTD).

ECC

See Elliptic Curve Cryptography (ECC).

ECDSA

See Elliptic Curve Digital Signature Algorithm (ECDSA).

EJB

See Enterprise Java Bean (EJB).

Elliptic Curve Cryptography (ECC)

Elliptic Curve Cryptography (ECC) is an alternative to the RSA encryption system which is based on the difficulty of solving elliptic curve discrete logarithm problems rather than on factoring large numbers. Developed and marketed by Certicom, ECC is especially suitable for environments, such as wireless devices and PC cards, where computational power is limited and high speed is required. For any given key size (measured in bits) ECC provides more security (is harder to decrypt without the key) than RSA.

Elliptic Curve Digital Signature Algorithm (ECDSA)

The Elliptic Curve Digital Signature Algorithm (ECDSA) is the elliptic curve analog of the Digital Signature Algorithm (DSA) standard. The advantages of ECDSA compared to RSA-like schemes are shorter key lengths and faster signing and decryption. For example, a 160 (210) bit ECC key is expected to give the same security as a 1024 (2048) bit RSA key, and the advantage increases as level of security is raised.

encryption

Encryption is the process of converting plaintext to ciphertext by applying a cryptographic algorithm.

encryption certificate

An encryption certificate is a certificate containing a public key that is used to encrypt electronic messages, files, documents, or data transmission, or to establish or exchange a session key for these same purposes.

end-to-end security

This is a property of message-level security that is established when a message traverses multiple applications within and between business entities and is secure over its full route through and between the business entities.

Enterprise Java Bean (EJB)

Enterprise JavaBeans (EJBs) are a Java API developed by Sun Microsystems that defines a component architecture for multi-tier client/server systems. Because EJB systems are written in Java, they are platform independent. Being object oriented, they can be implemented into existing systems with little or no recompiling and configuring.

Enterprise Manager

See Oracle Enterprise Manager.

failover

The process of failure recognition and recovery. In an Oracle Application Server Cold Failover Cluster (Identity Management), an application running on one cluster node is transparently migrated to another cluster node. During this migration, clients accessing the service on the cluster see a momentary outage and may need to reconnect once the failover is complete.

Federal Information Processing Standards (FIPS)

Federal Information Processing Standards (FIPS) are standards for information processing issued by the US government Department of Commerce's National Institute of Standards and Technology (NIST).

federated identity management (FIM)

The agreements, standards, and technologies that make identity and entitlements portable across autonomous domains. FIM makes it possible for an authenticated user to be recognized and take part in personalized services across multiple domains. It avoids pitfalls of centralized storage of personal information, while allowing users to link identity information between different accounts. Federated identity requires two key components: trust and standards. The trust model of federated identity management is based on circle of trust. The standards are defined by the Liberty Alliance Project.

federation

See identity federation.

filter

A filter is an expression that defines the entries to be returned from a request or search on a directory. Filters are typically expressed as DNs, for example: `cn=susie smith,o=acme,c=us`.

FIM

See federated identity management (FIM).

FIPS

See Federal Information Processing Standards (FIPS).

forced authentication

The act of forcing a user to reauthenticate if he or she has been idle for a preconfigured amount of time. Oracle Application Server Single Sign-On enables you to specify a global user inactivity timeout. This feature is intended for installations that have sensitive applications.

GET

An authentication method whereby login credentials are submitted as part of the login URL.

global administrator

In a hosted environment, one enterprise—for example, an application service provider—makes Oracle components available to multiple other enterprises and stores information for them. In such an environment, a global administrator performs activities that span the entire directory.

global unique identifier (GUID)

An identifier generated by the system and inserted into an entry when the entry is added to the directory. In a multimaster replicated environment, the GUID, not the DN, uniquely identifies an entry. The GUID of an entry cannot be modified by a user.

global user inactivity timeout

An optional feature of Oracle Application Server Single Sign-On that forces users to reauthenticate if they have been idle for a preconfigured amount of time. The global user inactivity timeout is much shorter than the single sign-out session timeout.

globally unique user ID

A numeric string that uniquely identifies a user. A person may change or add user names, passwords, and distinguished names, but her globally unique user ID always remains the same.

grace login

A login occurring within the specified period before password expiration.

guest user

One who is not an anonymous user, and, at the same time, does not have a specific user entry.

GUID

See global unique identifier (GUID).

handshake

A protocol two computers use to initiate a communication session.

hash

A number generated from a string of text with an algorithm. The hash value is substantially smaller than the text itself. Hash numbers are used for security and for faster access to data.

See also: hash function.

hash function

In cryptography, a hash function or one-way hash function is an algorithm that produces a given value when applied to a given block of data. The result of a hash function can be used to ensure the integrity of a given block of data. For a hash function to be considered secure, it must be very difficult, given a known data block and a known result, to produce another data block that produces the same result.

Hashed Message Authentication Code (HMAC)

Hashed Message Authentication Code (HMAC) is a hash function technique used to create a secret hash function output. This strengthens existing hash functions such as MD5 and SHA. It is used in transport layer security (TLS).

HMAC

See Hashed Message Authentication Code (HMAC).

HTTP

The Hyper Text Transfer Protocol (HTTP) is the protocol used between a Web browser and a server to request a document and transfer its contents. The specification is maintained and developed by the World Wide Web Consortium.

HTTP Server

See Oracle HTTP Server.

httpd.conf

The file used to configure Oracle HTTP Server.

iASAdmins

The administrative group responsible for user and group management functions in Oracle Application Server. The OracleAS Single Sign-On administrator is a member of the group iASAdmins.

identity federation

The linking of two or more accounts a Principal may hold with one or more identity providers or service providers within a given circle of trust.

When users federate the otherwise isolated accounts they have with businesses, known as their local identities, they create a relationship between two entities, an association comprising any number of identity providers and service providers.

identity management

The process by which the complete security lifecycle for network entities is managed in an organization. It typically refers to the management of an organization's application users, where steps in the security life cycle include account creation, suspension, privilege modification, and account deletion. The network entities managed may also include devices, processes, applications, or anything else that needs to interact in a networked environment. Entities managed by an identity management process may also include users outside of the organization, for example customers, trading partners, or Web services.

identity management infrastructure database

The database that contains data for OracleAS Single Sign-On and Oracle Internet Directory.

identity provider

One of the three primary roles defined in the identity federation protocols supported by Oracle Identity Federation. (The others are service provider and Principal.) The identity provider is responsible for managing and authenticating a set of identities within a given circle of trust.

A service provider (relying party in SAML), in turn, provides services or goods to a principal based on the identity provider's authentication of a principal's identity.

Identity providers are service providers offering business incentives so that other service providers affiliate with them. An identity provider will typically authenticate and asserts a principal's identity.

IdMBridge

The IdMBridge binds and provides user attributes for assertions, and is responsible for communication with various authoritative sources of data.

import agent

In an Oracle Directory Integration Platform environment, an agent that imports data into Oracle Internet Directory.

import data file

In an Oracle Directory Integration Platform environment, the file containing the data imported by an import agent.

infrastructure tier

The Oracle Application Server components responsible for identity management. These components are OracleAS Single Sign-On, Oracle Delegated Administration Services, and Oracle Internet Directory.

inherit

When an object class has been derived from another class, it also derives, or inherits, many of the characteristics of that other class. Similarly, an attribute subtype inherits the characteristics of its supertype.

integrity

In cryptography, integrity is the ability to detect if data has been modified by entities that are not authorized to modify it.

Internet Directory

See Oracle Internet Directory.

Internet Engineering Task Force (IETF)

The principal body engaged in the development of new Internet standard specifications. It is an international community of network designers, operators, vendors, and researchers concerned with the evolution of the Internet architecture and the smooth operation of the Internet.

Internet Message Access Protocol (IMAP)

A protocol allowing a client to access and manipulate electronic mail messages on a server. It permits manipulation of remote message folders, also called mailboxes, in a way that is functionally equivalent to local mailboxes.

J2EE

See Java 2 Platform, Enterprise Edition (J2EE).

Java 2 Platform, Enterprise Edition (J2EE)

Java 2 Platform, Enterprise Edition (J2EE) is an environment for developing and deploying enterprise applications, defined by Sun Microsystems Inc. The J2EE platform consists of a set of services, application programming interfaces (APIs), and protocols that provide the functionality for developing multitiered, Web-based applications.

Java Server Page (JSP)

JavaServer Pages (JSP), a server-side technology, are an extension to the Java servlet technology that was developed by Sun Microsystems. JSPs have dynamic scripting capability that works in tandem with HTML code, separating the page logic from the static elements (the design and display of the page). Embedded in the HTML page, the Java source code and its extensions help make the HTML more functional, being used in dynamic database queries, for example.

JSP

See Java Server Page (JSP).

key

A key is a data structure that contains some secret knowledge necessary to successfully encrypt or decrypt a given block of data. The larger the key, the harder it is to crack a block of encrypted data. For example, a 256-bit key is more secure than a 128-bit key.

key pair

A public key and its associated private key.

See also: public/private key pair.

latency

The time a client has to wait for a given directory operation to complete. Latency can be defined as wasted time. In networking discussions, latency is defined as the travel time of a packet from source to destination.

LDAP

See Lightweight Directory Access Protocol (LDAP).

LDAP connection cache

To improve throughput, the OracleAS Single Sign-On server caches and then reuses connections to Oracle Internet Directory.

LDAP Data Interchange Format (LDIF)

A common, text-based format for exchanging directory data between systems. The set of standards for formatting an input file for any of the LDAP command-line utilities.

LDIF

See LDAP Data Interchange Format (LDIF).

legacy application

An older application that cannot be modified to delegate authentication to the OracleAS Single Sign-On server. Also known as an external application.

Liberty Alliance

The Liberty Alliance Project is a consortium of companies, non-profits, and non-government organizations around the globe. It is committed to developing an open standard for federated identity management (FIM) and identity-based web services supporting current and emerging network devices.

Lightweight Directory Access Protocol (LDAP)

A set of protocols for accessing information in directories. LDAP supports TCP/IP, which is necessary for any type of Internet access. Its framework of design conventions

supports industry-standard directory products, such as Oracle Internet Directory. Because it is a simpler version of the X.500 standard, LDAP is sometimes called X.500 light.

load balancer

Hardware devices and software that balance connection requests between two or more servers, either due to heavy load or failover. BigIP, Alteon, or Local Director are all popular hardware devices. Oracle Application Server Web Cache is an example of load balancing software.

logical host

In an Oracle Application Server Cold Failover Cluster (Identity Management), one or more disk groups and pairs of host names and IP addresses. It is mapped to a physical host in the cluster. This physical host impersonates the host name and IP address of the logical host.

MAC

See message authentication code (MAC).

man-in-the-middle

A security attack characterized by the third-party, surreptitious interception of a message. The third-party, the *man-in-the-middle*, decrypts the message, re-encrypts it (with or without alteration of the original message), and retransmits it to the originally-intended recipient—all without the knowledge of the legitimate sender and receiver. This type of security attack works only in the absence of authentication.

mapping rules file

In an Oracle Directory Integration Platform environment, the file that specifies mappings between Oracle Internet Directory attributes and those in a connected directory.

master definition site (MDS)

In replication, a master definition site is the Oracle Internet Directory database from which the administrator runs the configuration scripts.

master site

In replication, a master site is any site other than the master definition site (MDS) that participates in LDAP replication.

MD2

Message Digest Two (MD2) is a message digest hash function. The algorithm processes input text and creates a 128-bit message digest which is unique to the message and can be used to verify data integrity. MD2 was developed by Ron Rivest for RSA Security and is intended to be used in systems with limited memory, such as smart cards.

MD4

Message Digest Four (MD4) is similar to MD2 but designed specifically for fast processing in software.

MD5

Message Digest Five (MD5) is a message digest hash function. The algorithm processes input text and creates a 128-bit message digest which is unique to the message and can be used to verify data integrity. MD5 was developed by Ron Rivest after potential

weaknesses were reported in MD4. MD5 is similar to MD4 but slower because more manipulation is made to the original data.

MDS

See master definition site (MDS).

message authentication

The process of verifying that a particular message came from a particular entity.

See also: authentication.

message authentication code (MAC)

The Message Authentication Code (MAC) is a result of a two-step process applied to a given block of data. First, the result of a hash function is obtained. Second, that result is encrypted using a secret key. The MAC can be used to authenticate the source of a given block of data.

message digest

The result of a hash function.

See also: hash.

metadirectory

A directory solution that shares information between all enterprise directories, integrating them into one virtual directory. It centralizes administration, thereby reducing administrative costs. It synchronizes data between directories, thereby ensuring that it is consistent and up-to-date across the enterprise.

middle tier

That portion of a OracleAS Single Sign-On instance that consists of the Oracle HTTP Server and OC4J. The OracleAS Single Sign-On middle tier is situated between the identity management infrastructure database and the client.

mod_osso

A module on the Oracle HTTP Server that enables applications protected by OracleAS Single Sign-On to accept HTTP headers in lieu of a user name and password once the user has logged into the OracleAS Single Sign-On server. The values for these headers are stored in the mod_osso cookie.

mod_osso cookie

User data stored on the HTTP server. The cookie is created when a user authenticates. When the same user requests another application, the Web server uses the information in the mod_osso cookie to log the user in to the application. This feature speeds server response time.

mod_proxy

A module on the Oracle HTTP Server that makes it possible to use mod_osso to enable single sign-on to legacy, or external applications.

MTS

See shared server.

multimaster replication

Also called peer-to-peer or *n*-way replication, a type of replication that enables multiple sites, acting as equals, to manage groups of replicated data. In a multimaster replication environment, each node is both a supplier and a consumer node, and the entire directory is replicated on each node.

naming attribute

The attribute used to compose the RDN of a new user entry created through Oracle Delegated Administration Services or Oracle Internet Directory Java APIs. The default value for this is `cn`.

nickname attribute

The attribute used to uniquely identify a user in the entire directory. The default value for this is `uid`. Applications use this to resolve a simple user name to the complete distinguished name. The user nickname attribute cannot be multi-valued—that is, a given user cannot have multiple nicknames stored under the same attribute name.

non-repudiation

In cryptography, the ability to prove that a given digital signature was produced with a given entity's private key, and that a message was sent untampered at a given point in time.

OASIS

Organization for the Advancement of Structured Information Standards. OASIS is a worldwide not-for-profit consortium that drives the development, convergence and adoption of e-business standards.

object class

In LDAP, object classes are used to group information. Typically an object class models a real-world object such as a person or a server. Each directory entry belongs to one or more object classes. The object class determines the attributes that make up an entry. One object class can be derived from another, thereby inheriting some of the characteristics of the other class.

OC4J

See Oracle Containers for J2EE (OC4J).

OCA

See Oracle Certificate Authority.

OCI

See Oracle Call Interface (OCI).

OCSP

See Online Certificate Status Protocol (OCSP).

OEM

See Oracle Enterprise Manager.

OID

See Oracle Internet Directory.

OID Control Utility

A command-line tool for issuing run-server and stop-server commands. The commands are interpreted and executed by the OID Monitor process.

OID Database Password Utility

The utility used to change the password with which Oracle Internet Directory connects to an Oracle Database.

OID Monitor

The Oracle Internet Directory component that initiates, monitors, and terminates the Oracle Internet Directory Server processes. It also controls the replication server if one is installed, and Oracle Directory Integration Platform Server.

Online Certificate Status Protocol (OCSP)

Online Certificate Status Protocol (OCSP) is one of two common schemes for checking the validity of digital certificates. The other, older method, which OCSP has superseded in some scenarios, is certificate revocation list (CRL). OCSP is specified in RFC 2560.

one-way function

A function that is easy to compute in one direction but quite difficult to reverse compute, that is, to compute in the opposite direction.

one-way hash function

A one-way function that takes a variable sized input and creates a fixed size output.

See also: hash function.

Oracle Application Server Single Sign-On

OracleAS Single Sign-On consists of program logic that enables you to log in securely to applications such as expense reports, mail, and benefits. These applications take two forms: partner applications and external applications. In both cases, you gain access to several applications by authenticating only once.

Oracle Call Interface (OCI)

An application programming interface (API) that enables you to create applications that use the native procedures or function calls of a third-generation language to access an Oracle Database server and control all phases of SQL statement execution.

Oracle Certificate Authority

Oracle Application Server Certificate Authority is a Certificate Authority (CA) for use within your Oracle Application Server environment. OracleAS Certificate Authority uses Oracle Internet Directory as the storage repository for certificates. OracleAS Certificate Authority integration with OracleAS Single Sign-On and Oracle Internet Directory provides seamless certificate provisioning mechanisms for applications relying on them. A user provisioned in Oracle Internet Directory and authenticated in OracleAS Single Sign-On can choose to request a digital certificate from OracleAS Certificate Authority.

Oracle CMS

Oracle CMS implements the IETF Cryptographic Message Syntax (CMS) protocol. CMS defines data protection schemes that allow for secure message envelopes.

Oracle Containers for J2EE (OC4J)

A lightweight, scalable container for Java 2 Platform, Enterprise Edition (J2EE).

Oracle Crypto

Oracle Crypto is a pure Java library that provides core cryptography algorithms.

Oracle Database Advanced Replication

A feature in the Oracle Database that enables database tables to be kept synchronized across two Oracle databases.

Oracle Delegated Administration Services

A set of individual, pre-defined services—called Oracle Delegated Administration Services units—for performing directory operations on behalf of a user. Oracle Internet Directory Self-Service Console makes it easier to develop and deploy administration solutions for both Oracle and third-party applications that use Oracle Internet Directory.

Oracle Directory Integration and Provisioning

A collection of interfaces and services for integrating multiple directories by using Oracle Internet Directory and several associated plug-ins and connectors. A feature of Oracle Internet Directory that enables an enterprise to use an external user repository to authenticate to Oracle products.

Oracle Directory Manager

A Java-based tool with a graphical user interface for administering Oracle Internet Directory.

Oracle Enterprise Manager

A separate Oracle product that combines a graphical console, agents, common services, and tools to provide an integrated and comprehensive systems management platform for managing Oracle products.

Oracle HTTP Server

Software that processes Web transactions that use the Hypertext Transfer Protocol (HTTP). Oracle uses HTTP software developed by the Apache Group.

Oracle Identity Management

An infrastructure enabling deployments to manage centrally and securely all enterprise identities and their access to various applications in the enterprise.

Oracle Internet Directory

A general purpose directory service that enables retrieval of information about dispersed users and network resources. It combines Lightweight Directory Access Protocol (LDAP) Version 3 with the high performance, scalability, robustness, and availability of the Oracle Database.

Oracle Liberty SDK

Oracle Liberty SDK implements the Liberty Alliance Project specifications enabling federated single sign-on between third-party Liberty-compliant applications.

Oracle Net Services

The foundation of the Oracle family of networking products, allowing services and their client applications to reside on different computers and communicate. The main

function of Oracle Net Services is to establish network sessions and transfer data between a client application and a server. Oracle Net Services is located on each computer in the network. Once a network session is established, Oracle Net Services acts as a data courier for the client and the server.

Oracle PKI certificate usages

Defines Oracle application types that a certificate supports.

Oracle PKI SDK

Oracle PKI SDK implements the security protocols that are necessary within public key infrastructure (PKI) implementations.

Oracle SAML

Oracle SAML provides a framework for the exchange of security credentials among disparate systems and applications in an XML-based format as outlined in the OASIS specification for the Security Assertions Markup Language (SAML).

Oracle Security Engine

Oracle Security Engine extends Oracle Crypto by offering X.509 based certificate management functions. Oracle Security Engine is a superset of Oracle Crypto.

Oracle S/MIME

Oracle S/MIME implements the Secure/Multipurpose Internet Mail Extension (S/MIME) specifications from the Internet Engineering Task Force (IETF) for secure e-mail.

Oracle Wallet Manager

A Java-based application that security administrators use to manage public-key security credentials on clients and servers.

See also: *Oracle Advanced Security Administrator's Guide*.

Oracle Web Services Security

Oracle Web Services Security provides a framework for authentication and authorization using existing security technologies as outlined in the OASIS specification for Web Services Security.

Oracle XML Security

Oracle XML Security implements the W3C specifications for XML Encryption and XML Signature.

OracleAS Portal

An OracleAS Single Sign-On partner application that provides a mechanism for integrating files, images, applications, and Web sites. The External Applications portlet provides access to external applications.

OWM

See Oracle Wallet Manager.

partition

A unique, non-overlapping directory naming context that is stored on one directory server.

partner application

An Oracle Application Server application or non-Oracle application that delegates the authentication function to the OracleAS Single Sign-On server. This type of application spares users from reauthenticating by accepting mod_osso headers.

peer-to-peer replication

Also called multimaster replication or *n*-way replication. A type of replication that enables multiple sites, acting as equals, to manage groups of replicated data. In such a replication environment, each node is both a supplier and a consumer node, and the entire directory is replicated on each node.

PKCS#1

The Public Key Cryptography Standards (PKCS) are specifications produced by RSA Laboratories. PKCS#1 provides recommendations for the implementation of public-key cryptography based on the RSA algorithm, covering the following aspects: cryptographic primitives; encryption schemes; signature schemes; ASN.1 syntax for representing keys and for identifying the schemes.

PKCS#5

The Public Key Cryptography Standards (PKCS) are specifications produced by RSA Laboratories. PKCS#5 provides recommendations for the implementation of password-based cryptography.

PKCS#7

The Public Key Cryptography Standards (PKCS) are specifications produced by RSA Laboratories. PKCS #7 describes general syntax for data that may have cryptography applied to it, such as digital signatures and digital envelopes.

PKCS#8

The Public Key Cryptography Standards (PKCS) are specifications produced by RSA Laboratories. PKCS #8 describes syntax for private key information, including a private key for some public key algorithms and a set of attributes. The standard also describes syntax for encrypted private keys.

PKCS#10

The Public Key Cryptography Standards (PKCS) are specifications produced by RSA Laboratories. PKCS #10 describes syntax for a request for certification of a public key, a name, and possibly a set of attributes.

PKCS#12

The Public Key Cryptography Standards (PKCS) are specifications produced by RSA Laboratories. PKCS #12 describes a transfer syntax for personal identity information, including private keys, certificates, miscellaneous secrets, and extensions. Systems (such as browsers or operating systems) that support this standard allow a user to import, export, and exercise a single set of personal identity information—typically in a format called a wallet.

PKI

See public key infrastructure (PKI).

plaintext

Plaintext is readable data prior to a transformation to ciphertext using encryption, or readable data that is the result of a transformation from ciphertext using decryption.

point-to-point replication

Also called fan-out replication is a type of replication in which a supplier replicates directly to a consumer. That consumer can then replicate to one or more other consumers. The replication can be either full or partial.

policy precedence

In Oracle Application Server Certificate Authority (OCA), policies are applied to incoming requests in the order that they are displayed on the main policy page. When the OCA policy processor module parses policies, those that appear toward the top of the policy list are applied to requests first. Those that appear toward the bottom of the list are applied last and take precedence over the others. Only enabled policies are applied to incoming requests.

policy.properties

A multipurpose configuration file for Oracle Application Server Single Sign-On that contains basic parameters required by the single sign-on server. Also used to configure advanced features of OracleAS Single Sign-On, such as multilevel authentication.

POSIX

Portable Operating System Interface for UNIX. A set of programming interface standards governing how to write application source code so that the applications are portable between operating systems. A series of standards being developed by the Internet Engineering Task Force (IETF).

POST

An authentication method whereby login credentials are submitted within the body of the login form.

predicates

In Oracle Application Server Certificate Authority (OCA), a policy predicate is a logical expression that can be applied to a policy to limit how it is applied to incoming certificate requests or revocations. For example, the following predicate expression specifies that the policy in which it appears can have a different effect for requests or revocations from clients with DNs that include "ou=sales,o=acme,c=us":

```
Type=="client" AND DN=="ou=sales,o=acme,c=us"
```

primary node

In an Oracle Application Server Cold Failover Cluster (Identity Management), the cluster node on which the application runs at any given time.

See also: secondary node.

principal

A principal is any entity capable of using a service and capable of acquiring a federated identity.

See also: federated identity management (FIM).

private key

A private key is the secret key in a public/private key pair used in public key cryptography. An entity uses its private key to decrypt data that has been encrypted with its public key. The entity can also use its private key to create digital signatures. The security of data encrypted with the entity's public key as well as signatures created by the private key depends on the private key remaining secret.

private key cryptography

See symmetric cryptography.

provisioned applications

Applications in an environment where user and group information is centralized in Oracle Internet Directory. These applications are typically interested in changes to that information in Oracle Internet Directory.

provisioning

The process of providing users with access to applications and other resources that may be available in an enterprise environment.

provisioning agent

An application or process that translates Oracle-specific provisioning events to external or third-party application-specific events.

proxy server

A server between a client application, such as a Web browser, and a real server. It intercepts all requests to the real server to see if it can fulfil the requests itself. If not, it forwards the request to the real server. In OracleAS Single Sign-On, proxies are used for load balancing and as an extra layer of security.

See also: load balancer.

proxy user

A kind of user typically employed in an environment with a middle tier such as a firewall. In such an environment, the end user authenticates to the middle tier. The middle tier then logs into the directory on the end user's behalf. A proxy user has the privilege to switch identities and, once it has logged into the directory, switches to the end user's identity. It then performs operations on the end user's behalf, using the authorization appropriate to that particular end user.

public key

A public key is the non-secret key in a public/private key pair used in public key cryptography. A public key allows entities to encrypt data that can only then be decrypted with the public key's owner using the corresponding private key. A public key can also be used to verify digital signatures created with the corresponding private key.

public key certificate

See certificate.

public key cryptography

Public key cryptography (also known as asymmetric cryptography) uses two keys, one public and the other private. These keys are called a key pair. The private key must be kept secret, while the public key can be transmitted to any party. The private key and the public key are mathematically related. A message that is signed by a private key can be verified by the corresponding public key. Similarly, a message encrypted by the public key can be decrypted by the private key. This method ensures privacy because only the owner of the private key can decrypt the message.

public key encryption

The process in which the sender of a message encrypts the message with the public key of the recipient. Upon delivery, the message is decrypted by the recipient using the recipient's private key.

public key infrastructure (PKI)

A public key infrastructure (PKI) is a system that manages the issuing, distribution, and authentication of public keys and private keys. A PKI typically comprises the following components:

- A Certificate Authority (CA) that is responsible for generating, issuing, publishing and revoking digital certificates.
- A Registration Authority (RA) that is responsible for verifying the information supplied in requests for certificates made to the CA.
- A directory service where a certificate or certificate revocation list (CRL) gets published by the CA and where they can be retrieved by relying third parties.
- Relying third parties that use the certificates issued by the CA and the public keys contained therein to verify digital signatures and encrypt data.

public/private key pair

A mathematically related set of two numbers where one is called the private key and the other is called the public key. Public keys are typically made widely available, while private keys are available only to their owners. Data encrypted with a public key can only be decrypted with its associated private key and vice versa. Data encrypted with a public key cannot be decrypted with the same public key.

RC2

Rivest Cipher Two (RC2) is a 64-bit block cipher developed by Ronald Rivest for RSA Security, and was designed as a replacement for Data Encryption Standard (DES).

RC4

Rivest Cipher Four (RC4) is a stream cipher developed by Ronald Rivest for RSA Security. RC4 allows variable key lengths up to 1024 bits. RC4 is most commonly used to secure data communications by encrypting traffic between Web sites that use the Secure Sockets Layer (SSL) protocol.

RDN

See relative distinguished name (RDN).

readable data

Data prior to a transformation to ciphertext via encryption or data that is the result of a transformation from ciphertext via decryption.

Registration Authority (RA)

The Registration Authority (RA) is responsible for verifying and enrolling users before a certificate is issued by a Certificate Authority (CA). The RA may assign each applicant a relative distinguished value or name for the new certificate applied. The RA does not sign or issue certificates.

relational database

A structured collection of data that stores data in tables consisting of one or more rows, each containing the same set of columns. Oracle makes it very easy to link the data in multiple tables. This is what makes Oracle a relational database management

system, or RDBMS. It stores data in two or more tables and enables you to define relationships between the tables. The link is based on one or more fields common to both tables.

relative distinguished name (RDN)

The local, most granular level entry name. It has no other qualifying entry names that would serve to uniquely address the entry. In the example, `cn=Smith,o=acme,c=US`, the RDN is `cn=Smith`.

remote master site (RMS)

In a replicated environment, any site, other than the master definition site (MDS), that participates in Oracle Database Advanced Replication.

response time

The time between the submission of a request and the completion of the response.

RFC

The Internet Request For Comments (or RFC) documents are the written definitions of the protocols and policies of the Internet. The Internet Engineering Task Force (IETF) facilitates the discussion, development, and establishment of new standards. A standard is published using the RFC acronym and a reference number. For example, the official standard for e-mail is RFC 822.

root CA

In a hierarchical public key infrastructure (PKI), the root Certificate Authority (CA) is the CA whose public key serves as the most trusted datum for a security domain.

root directory specific entry (DSE)

An entry storing operational information about the directory. The information is stored in a number of attributes.

root DSE

See root directory specific entry (DSE).

root Oracle Context

In the Oracle Identity Management infrastructure, the root Oracle Context is an entry in Oracle Internet Directory containing a pointer to the default identity management realm in the infrastructure. It also contains information on how to locate an identity management realm given a simple name of the realm.

RSA

RSA is a public key cryptography algorithm named after its inventors (Rivest, Shamir, and Adelman). The RSA algorithm is the most commonly used encryption and authentication algorithm and is included as part of the Web browsers from Netscape and Microsoft, and many other products.

RSAES-OAEP

The RSA Encryption Scheme - Optimal Asymmetric Encryption Padding (RSAES-OAEP) is a public key encryption scheme combining the RSA algorithm with the OAEP method. Optimal Asymmetric Encryption Padding (OAEP) is a method for encoding messages developed by Mihir Bellare and Phil Rogaway.

S/MIME

See Secure/Multipurpose Internet Mail Extension (S/MIME).

SAML

See Security Assertions Markup Language (SAML).

SASL

See Simple Authentication and Security Layer (SASL).

scalability

The ability of a system to provide throughput in proportion to, and limited only by, available hardware resources.

schema

The collection of attributes, object classes, and their corresponding matching rules.

secondary node

In an Oracle Application Server Cold Failover Cluster (Identity Management), the cluster node to which an application is moved during a failover.

See also: primary node.

secret key

A secret key is the key used in a symmetric algorithm. Since a secret key is used for both encryption and decryption, it must be shared between parties that are transmitting ciphertext to one another but must be kept secret from all unauthorized entities.

secret key cryptography

See symmetric cryptography.

Secure Hash Algorithm (SHA)

Secure Hash Algorithm (SHA) is a hash function algorithm that produces a 160-bit message digest based upon the input. The algorithm is used in the Digital Signature Standard (DSS). With the introduction of the Advanced Encryption Standard (AES) which offers three key sizes: 128, 192 and 256 bits, there has been a need for a companion hash algorithm with a similar level of security. The newer SHA-256, SHA-284 and SHA-512 hash algorithms comply with these enhanced requirements.

Secure Sockets Layer (SSL)

Secure Sockets Layer (SSL) is a protocol designed by Netscape Communications to enable encrypted, authenticated communications across networks (such as the Internet). SSL uses the public key encryption system from RSA, which also includes the use of a digital certificate. SSL provides three elements of secure communications: confidentiality, authentication, and integrity.

SSL has evolved into Transport Layer Security (TLS). TLS and SSL are not interoperable. However, a message sent with TLS can be handled by a client that handles SSL.

Secure/Multipurpose Internet Mail Extension (S/MIME)

Secure/Multipurpose Internet Mail Extension (S/MIME) is an Internet Engineering Task Force (IETF) standard for securing MIME data through the use of digital signatures and encryption.

Security Assertions Markup Language (SAML)

Security Assertions Markup Language (SAML) is an XML-based framework for exchanging security information over the Internet. SAML enables the exchange of authentication and authorization information between various security services systems that otherwise would not be able to interoperate. The SAML 1.0 specification was adopted by OASIS in 2002.

server certificate

A certificate that attests to the identity of an organization that uses a secure Web server to serve data. A server certificate must be associated with a public/private key pair issued by a mutually trusted Certificate Authority (CA). Server certificates are required for secure communications between a browser and a Web server.

service provider

These are organizations recognized by the members of a circle of trust as the entities that provide Web-based services to users. Service providers enter into partnerships with other service providers and identity providers with the goal of providing their common users with secure single sign-on between all parties of the federation.

service time

The time between the initiation of a request and the completion of the response to the request.

session key

A secret key that is used for the duration of one message or communication session.

SGA

See System Global Area (SGA).

SHA

See Secure Hash Algorithm (SHA).

shared server

A server that is configured to allow many user processes to share very few server processes, so the number of users that can be supported is increased. With shared server configuration, many user processes connect to a dispatcher. The dispatcher directs multiple incoming network session requests to a common queue. An idle shared server process from a shared pool of server processes picks up a request from the queue. This means a small pool of server processes can server a large amount of clients. Contrast with dedicated server.

sibling

An entry that has the same parent as one or more other entries.

Signed Public Key And Challenge (SPKAC)

Signed Public Key And Challenge (SPKAC) is a proprietary protocol used by the Netscape Navigator browser to request certificates.

simple authentication

The process by which the client identifies itself to the server by means of a DN and a password which are not encrypted when sent over the network. In the simple authentication option, the server verifies that the DN and password sent by the client match the DN and password stored in the directory.

Simple Authentication and Security Layer (SASL)

A method for adding authentication support to connection-based protocols. To use this specification, a protocol includes a command for identifying and authenticating a user to a server and for optionally negotiating a security layer for subsequent protocol interactions. The command has a required argument identifying a SASL mechanism.

single key-pair wallet

A PKCS#12-format wallet that contains a single user certificate and its associated private key. The public key is imbedded in the certificate.

single sign-off

The process by which you terminate an OracleAS Single Sign-On session and log out of all active partner applications simultaneously. You can do this by logging out of the application that you are working in.

single sign-on (SSO)

A process or system that enables a user to access multiple computer platforms or application systems after being authenticated only once.

single sign-on SDK

Legacy APIs to enable OracleAS Single Sign-On partner applications for single sign-on. The SDK consists of PL/SQL and Java APIs as well as sample code that demonstrates how these APIs are implemented. This SDK is now deprecated and mod_osso is used instead.

single sign-on server

Program logic that enables users to log in securely to single sign-on applications such as expense reports, mail, and benefits.

SLAPD

Standalone LDAP daemon. An LDAP directory server service that is responsible for most functions of a directory except replication.

SOAP

Simple Object Access Protocol (SOAP) is an XML-based protocol that defines a framework for passing messages between systems over the Internet via HTTP. A SOAP message consists of three parts — an envelope that describes the message and how to process it, a set of encoding rules for expressing instances of application-defined datatypes, and a convention for representing remote procedure calls and responses.

specific administrative area

Administrative areas control:

- Subschema administration
- Access control administration
- Collective attribute administration

A *specific* administrative area controls one of these aspects of administration. A specific administrative area is part of an autonomous administrative area.

SPKAC

See Signed Public Key And Challenge (SPKAC).

sponsor node

In replication, the node that is used to provide initial data to a new node.

SSL

See Secure Sockets Layer (SSL).

stream cipher

Stream ciphers are a type of symmetric algorithm. A stream cipher encrypts in small units, often a bit or a byte at a time, and implements some form of feedback mechanism so that the key is constantly changing. RC4 is an example of a stream cipher.

subACLSubentry

A specific type of subentry that contains access control list (ACL) information.

subclass

An object class derived from another object class. The object class from which it is derived is called its superclass.

subentry

A type of entry containing information applicable to a group of entries in a subtree. The information can be of these types:

- Access control policy points
- Schema rules
- Collective attributes

Subentries are located immediately below the root of an administrative area.

subordinate CA

In a hierarchical public key infrastructure (PKI), the subordinate Certificate Authority (CA) is a CA whose certificate signature key is certified by another CA, and whose activities are constrained by that other CA.

subordinate reference

A knowledge reference pointing downward in the directory information tree (DIT) to a naming context that starts immediately below an entry

subschema DN

The list of directory information tree (DIT) areas having independent schema definitions.

subSchemaSubentry

A specific type of subentry containing schema information.

subtree

A section of a directory hierarchy, which is also called a directory information tree (DIT). The subtree typically starts at a particular directory node and includes all subdirectories and objects below that node in the directory hierarchy.

subtype

An attribute with one or more options, in contrast to that same attribute without the options. For example, a `commonName` (`cn`) attribute with American English as an

option is a subtype of the commonName (cn) attribute without that option. Conversely, the commonName (cn) attribute without an option is the supertype of the same attribute with an option.

success URL

When using Oracle Application Server Single Sign-On, the URL to the routine responsible for establishing the session and session cookies for an application.

super user

A special directory administrator who typically has full access to directory information.

superclass

The object class from which another object class is derived. For example, the object class person is the superclass of the object class organizationalPerson. The latter, namely, organizationalPerson, is a subclass of person and inherits the attributes contained in person.

superior reference

A knowledge reference pointing upward to a directory system agent (DSA) that holds a naming context higher in the directory information tree (DIT) than all the naming contexts held by the referencing DSA.

supertype

An attribute without options, in contrast to the same attribute with one or more options. For example, the commonName (cn) attribute without an option is the supertype of the same attribute with an option. Conversely, a commonName (cn) attribute with American English as an option is a subtype of the commonName (cn) attribute without that option.

supplier

In replication, the server that holds the master copy of the naming context. It supplies updates from the master copy to the consumer server.

symmetric algorithm

A symmetric algorithm is a cryptographic algorithm that uses the same key for encryption and decryption. There are essentially two types of symmetric (or secret key) algorithms — stream ciphers and block ciphers.

symmetric cryptography

Symmetric cryptography (or shared secret cryptography) systems use the same key to encipher and decipher data. The problem with symmetric cryptography is ensuring a secure method by which the sender and recipient can agree on the secret key. If a third party were to intercept the secret key in transit, they could then use it to decipher anything it was used to encipher. Symmetric cryptography is usually faster than asymmetric cryptography, and is often used when large quantities of data need to be exchanged. DES, RC2, and RC4 are examples of symmetric cryptography algorithms.

symmetric key

See secret key.

System Global Area (SGA)

A group of shared memory structures that contain data and control information for one Oracle database instance. If multiple users are concurrently connected to the same

instance, the data in the instance SGA is shared among the users. Consequently, the SGA is sometimes referred to as the "shared global area." The combination of the background processes and memory buffers is called an Oracle instance.

system operational attribute

An attribute holding information that pertains to the operation of the directory itself. Some operational information is specified by the directory to control the server, for example, the time stamp for an entry. Other operational information, such as access information, is defined by administrators and is used by the directory program in its processing.

think time

The time the user is not engaged in actual use of the processor.

third-party access management system

Non-Oracle single sign-on system that can be modified to use OracleAS Single Sign-On to gain access to Oracle Application Server applications.

throughput

The number of requests processed byOracle Internet Directory for each unit of time. This is typically represented as "operations per second."

Time Stamp Protocol (TSP)

Time Stamp Protocol (TSP), as specified in RFC 3161, defines the participating entities, the message formats, and the transport protocol involved in time stamping a digital message. In a TSP system, a trusted third-party Time Stamp Authority (TSA) issues time stamps for messages.

TLS

See Transport Layer Security (TLS).

Transport Layer Security (TLS)

A protocol providing communications privacy over the Internet. The protocol enables client/server applications to communicate in a way that prevents eavesdropping, tampering, or message forgery.

Triple Data Encryption Standard (3DES)

Triple Data Encryption Standard (3DES) is based on the Data Encryption Standard (DES) algorithm developed by IBM in 1974, and was adopted as a national standard in 1977. 3DES uses three 64-bit long keys (overall key length is 192 bits, although actual key length is 56 bits). Data is encrypted with the first key, decrypted with the second key, and finally encrypted again with the third key. This makes 3DES three times slower than standard DES but also three times more secure.

trusted certificate

A third party identity that is qualified with a level of trust. The trust is used when an identity is being validated as the entity it claims to be. Typically, trusted certificates come from a Certificate Authority (CA) you trust to issue user certificates.

trustpoint

See trusted certificate.

TSP

See Time Stamp Protocol (TSP).

Unicode

A type of universal character set, a collection of 64K characters encoded in a 16-bit space. It encodes nearly every character in just about every existing character set standard, covering most written scripts used in the world. It is owned and defined by Unicode Inc. Unicode is canonical encoding which means its value can be passed around in different locales. But it does not guarantee a round-trip conversion between it and every Oracle character set without information loss.

UNIX Crypt

The UNIX encryption algorithm.

URI

Uniform Resource Identifier (URI). A way to identify any point of content on the Web, whether it be a page of text, a video or sound clip, a still or animated image, or a program. The most common form of URI is the Web page address, which is a particular form or subset of URI called a URL.

URL

Uniform Resource Locator (URL). The address of a file accessible on the Internet. The file can be a text file, HTML page, image file, a program, or any other file supported by HTTP. The URL contains the name of the protocol required to access the resource, a domain name that identifies a specific computer on the Internet, and a hierarchical description of the file location on the computer.

URLC token

The OracleAS Single Sign-On code that passes authenticated user information to the partner application. The partner application uses this information to construct the session cookie.

user name mapping module

A OracleAS Single Sign-On Java module that maps a user certificate to the user's nickname. The nickname is then passed to an authentication module, which uses this nickname to retrieve the user's certificate from the directory.

user search base

In the Oracle Internet Directory default directory information tree (DIT), the node in the identity management realm under which all the users are placed.

UTC (Coordinated Universal Time)

The standard time common to every place in the world. Formerly and still widely called Greenwich Mean Time (GMT) and also World Time, UTC nominally reflects the mean solar time along the Earth's prime meridian. UTC is indicated by a z at the end of the value, for example, 200011281010z.

UTF-8

A variable-width 8-bit encoding of Unicode that uses sequences of 1, 2, 3, or 4 bytes for each character. Characters from 0-127 (the 7-bit ASCII characters) are encoded with one byte, characters from 128-2047 require two bytes, characters from 2048-65535 require three bytes, and characters beyond 65535 require four bytes. The Oracle character set name for this is AL32UTF8 (for the Unicode 3.1 standard).

UTF-16

16-bit encoding of Unicode.The Latin-1 characters are the first 256 code points in this standard.

verification

Verification is the process of ensuring that a given digital signature is valid, given the public key that corresponds to the private key purported to create the signature and the data block to which the signature purportedly applies.

virtual host

A single physical Web server machine that is hosting one or more Web sites or domains, or a server that is acting as a proxy to other machines (accepts incoming requests and reroutes them to the appropriate server).

In the case of OracleAS Single Sign-On, virtual hosts are used for load balancing between two or more OracleAS Single Sign-On servers. They also provide an extra layer of security.

virtual host name

In an Oracle Application Server Cold Failover Cluster (Identity Management), the host name corresponding to a particular virtual IP address.

virtual IP address

In an Oracle Application Server Cold Failover Cluster (Identity Management), each physical node has its own physical IP address and physical host name. To present a single system image to the outside world, the cluster uses a dynamic IP address that can be moved to any physical node in the cluster. This is called the virtual IP address.

wait time

The time between the submission of the request and initiation of the response.

wallet

An abstraction used to store and manage security credentials for an individual entity. It implements the storage and retrieval of credentials for use with various cryptographic services. A wallet resource locator (WRL) provides all the necessary information to locate the wallet.

Wallet Manager

See Oracle Wallet Manager.

Web service

A Web service is application or business logic that is accessible using standard Internet protocols, such as HTTP, XML, and SOAP. Web Services combine the best aspects of component-based development and the World Wide Web. Like components, Web Services represent black-box functionality that can be used and reused without regard to how the service is implemented.

Web Services Description Language (WSDL)

Web Services Description Language (WSDL) is the standard format for describing a Web service using XML. A WSDL definition describes how to access a Web service and what operations it will perform.

WSDL

See Web Services Description Language (WSDL).

WS-Federation

Web Services Federation Language (WS-Federation) is a specification developed by Microsoft, IBM, BEA, VeriSign, and RSA Security. It defines mechanisms to allow federation between entities using different or like mechanisms by allowing and brokering trust of identities, attributes, and authentication between participating Web services.

See also: Liberty Alliance.

X.500

X.500 is a standard from the International Telecommunication Union (ITU) that defines how global directories should be structured. X.500 directories are hierarchical with different levels for each category of information, such as country, state, and city.

X.509

X.509 is the most widely used standard for defining digital certificates. A standard from the International Telecommunication Union (ITU), for hierarchical directories with authentication services, used in many public key infrastructure (PKI) implementations.

XML

Extensible Markup Language (XML) is a specification developed by the World Wide Web Consortium (W3C). XML is a pared-down version of Standard Generalized Mark-Up Language (SGML), designed especially for Web documents. XML is a metalanguage (a way to define tag sets) that allows developers to define their own customized markup language for many classes of documents.

XML canonicalization (C14N)

This is a process by which two logically equivalent XML documents can be resolved to the same physical representation. This has significance for digital signatures because a signature can only verify against the same physical representation of the data against which it was originally computed. For more information, see the W3C's XML Canonicalization specification.

Index

A

B

C

D

E

F

H

I

K

L

M

N

O

P

R

S

T

U

W

X